Introduction

Introduction

Hello Cloud Gurus, and welcome to the AWS Certified SysOps Administrator Associate course. I'm Faye Ellis, and I'll be one of your instructors. I've got 20 years' experience as a solutions architect, DevOps engineer, and a Linux Systems Administrator. I also teach the AWS Certified Developer Associate course, and I've taught AWS to millions of students. And I'm Errol, and I'll also be your instructor. I have over 10 years of experience in IT, ranging from data warehouse and Java development, data architecture, cloud engineering, and now my current role, which trains students learning to cloud. So you are in really, really good hands. In this course, you will learn everything you need to know in order to pass the AWS SysOps Administrator Associate exam. You will learn things like CloudWatch, AWS Config, CloudTrail, Event Bridge, Systems Manager, EC2, Elastic Load Balancer, Auto Scaling, and Bastion Hosts. You will also learn RDS, Aurora, Storage Gateway, Athena, S3, Elastic File System, KMS, and CloudFormation. We'll have deep dives on subjects such as security in AWS, advanced networking with AWS, business continuity, and cost optimization as well, and you will learn an awful lot by doing this course. Yes, the Certified SysOps Administrator Associate exam is one of the toughest AWS associate-level exams out there, especially since it now includes a practical element. That being said, once you've completed this course and completed our practical exercises, you will have learned everything that you need to know in order to pass the Certified SysOps Administrator Associate exam. So who should do this course? Well, this isn't a beginner's course and we assume you've either completed the Certified Solutions Architect Associate course and you've passed the exam or you've worked as a SysOps administrator for at least one year on AWS. And if you haven't done either, go do the Solutions Architect Associate course first because we do assume prior knowledge on AWS in order to complete this course. So, why not get started today? Get certified and become a real-life Cloud Guru. Keep being awesome, Cloud Gurus, and we'll see you inside the course!

The Exam Guide

Hello cloud gurus, and welcome to this lecture, which is going to cover the exam guide, so all the different topics that are going to be covered in the exam. And the exam topics are divided into six different domains, and the exam gives different weightings for each of these different domains. So Domain 1 is Monitoring, Logging, and Remediation. Domain 2 is Reliability and Business Continuity. Domain 3 covers Deployment, Provisioning, and Automation. Domain 4 is Security and Compliance. Domain 5 is Networking and Content Delivery. And, Domain 6 is Cost and Performance Optimization. So the exam gives different weightings for each of these different domains, so there is not an equal focus on each domain. For example, the biggest section is Domain 1, which is the Monitoring, Logging, and Remediation section, which is worth 20% of the exam. And, typically, where you see the higher weighting, that's where you will see more questions on those areas than some of the others. So, let's take a look at this in a little bit more detail, and there's actually an official exam guide document, which I'll show you now. And I've included a link to this web page in the Resources section for this lecture, so you'll be able to go through this and read it through in your own time as well, but I'm just going to talk you through the main points. So moving down to Recommended AWS knowledge, and they do recommend that you have a minimum of 1 year hands-on experience with AWS, as well as the following recommended AWS knowledge. So,

experience in deploying, managing, and operating workloads on AWS. Understanding the AWS Well-Architected Framework. Hands-on experience with the Management Console and CLI. And understanding of AWS networking and security services. And, hands-on experience implementing security controls and compliance requirements as well. So this is not a beginners certification, and in this course, we do assume that you have either completed the Certified Solutions Architect - Associate course and passed the exam, or, that you have worked as a SysOps administrator with AWS for at least 1 year. So now, let's take a look at the exam content, and there are now three different types of questions that you are going to see in the exam. So firstly, there's multiple choice, which has one correct response and three incorrect responses. Next, we have multiple response, which has two correct responses out of five options. And then there is the exam lab, and this is where you have a scenario composed of a set of tasks that you must perform in the AWS Management Console or using the AWS CLI. So this is a practical exercise that you must complete, and they will provide you with an AWS account to complete the required tasks. Now, at the beginning of the exam, they will tell you how many multiple-choice questions you will get and how many exam labs you will get. And, you normally get around 50 of the multiple-choice and multiple-response questions, followed by 3 exam labs. And they recommend that you plan at least at 20 minutes to complete each exam lab, and you will need to complete all work on an exam lab before moving on to the next one, and they don't allow you to go back once you have completed an exam lab. Here's a link to the sample questions so that you can see what to expect in the exam. And, be aware that the exam now includes 15 unscored questions that do not affect your score, and these are new questions that they are evaluating for future use. Now, AWS have always included unscored questions, but this is the first time that I've seen them explicitly state the quantity of unscored questions. So moving on to exam results, and it is a pass or fail exam, it is not graded and you are scored out of a maximum score of 1,000, with a minimum passing score of 720. And, on the day, you will find out immediately whether you've passed or failed, but you don't get your score immediately, I think it's usually within around two or three days. They will send you an email with the details of your exact score. Now, for the multiple-choice questions, there is no partial scoring, so you don't get any marks for getting the answer partially correct. However, with the exam labs, they do use partial scoring, so if you can only complete part of the exercise correctly, you will still get some marks. So moving down to the next page, this just drills down into each of the six exam domains, and we will cover all of these topics in the SysOps Administrator - Associate course, including plenty of practical hands-on exercises and tasks, quizzes to test your knowledge, and our exam simulator as well, which will help you to get ready for the exam. So in summary, the exam is 190 minutes in length, so you do get a lot of time to answer the questions. You'll see approximately 50 questions. The questions will be a mix of multiple choice and multiple response, and they will all be scenario-based questions. You'll also get three exam labs, which are the practical tasks that you will need to complete using the AWS console, or you can use the AWS CLI. The pass mark is 720 out of 1,000 points. The cost is 150 US dollars, and when you gain this qualification, it will be valid for 3 years. And then when it comes to the exam labs, these are based on a scenario composed of a set of tasks that you must perform using the AWS console, or you could optionally use the AWS CLI. They are designed to test your practical AWS skills, and you should allow 20 minutes to complete each exam lab. And, just remember that partial credit is awarded for partial completion, so even if you don't know how to completely do the exam lab, it's definitely worth a try because you will get partial credit for the elements that you do correctly. And if you are interested in learning about my own experience with this exam, you can check out my blog post, which I have linked to in the Resources section of the course. And, I've also included a link to the official sample questions as

well so you can take a look at the kind of questions and practical exercises that you can expect to see in this exam. So that is the exam guide, and that's the end of this lecture. If you have any questions, please let me know; otherwise, I'll see you in the next lecture. Thank you.

Introducing the AWS Cloud Sandbox

Hello cloud gurus, and welcome to this lecture, which is going to introduce an awesome new feature of the A Cloud Guru platform, and it's called the AWS Cloud Sandbox. Now, if you are a Personal Plus subscriber or if you are using a subscription provided by your company, then you will have access to this amazing feature. And the AWS Cloud Sandbox allows you to provision and configure AWS resources using a live AWS account that we provide. So you don't need to use your own AWS account, and it allows you to safely play in a real-world environment without incurring any AWS charges, and we will pick up the bill. So how does it work? Well, from the A Cloud Guru website, select Cloud Playground, and this will take you to the AWS Cloud Sandbox. And you will find the Cloud Playground here at the top of your dashboard, or you can also access it from within a course. So select Cloud Playground, start AWS Sandbox, and here are the credentials for your AWS Sandbox. So here is the Username and Password that you can use to log in to the console. Here is a login URL, and you also get an Access Key ID and Secret Access Key to use with the AWS command line interface. And then down here, it's going to tell you how long this sandbox is going to be provided for, so after 4 hours, this sandbox is going to be shut down, and any resources that you created will be deleted. So I'm going to right-click on Open Sandbox, and I'll open in a new incognito window. And if you are using Internet Explorer or Safari, then it's called private browsing mode. So open that up in a new window and sign into the console using the credentials that we've provided. And there you go, you've got your own AWS Cloud Sandbox to safely play, experiment, and explore. And you can use this account to get loads of hands-on experience and complete the practical exercises throughout this course. And if you would like more information about the Cloud Playground, I've included a link to the FAQ in the Resources section for this lecture. And in addition to the AWS Cloud Sandbox, we also have sandboxes for Azure and GCP as well. So that's it for this lecture. I hope you have loads of fun using our Cloud Sandbox. If you have any questions, please let me know. Otherwise, I will see you in the course. Thank you.

Introducing Hands-On Labs

Hello cloud gurus, and welcome to this lecture, which is just a quick introduction to hands-on labs. So what are hands-on labs? Well, the best way to learn how to swim is, of course, to jump in the water. And the best way to learn a new skill is by gaining practical experience, and that's exactly what you get with our hands-on labs. Simply launch the lab, and once the lab is ready, it will create a live AWS environment, including console or AWS CLI credentials that you can use to complete the objectives of the lab. You'll find the lab objectives down here. There's also a diagram, and videos to guide you through the steps should you need them. And there's also a lab guide right here to talk you through the steps as well. And depending on your level of experience, you can either complete all of the objectives by yourself, or you can watch the videos and complete the steps with one of our awesome training architects to guide you each step of the way. And I've included plenty of hands-on labs throughout this course, but if you'd like to find more, then select our Labs icon at the top of the screen, and this will take you to all of the hands-on labs available. And right now for AWS we've got 233, but we are adding more all the time. So that's it for this lecture. I hope you have loads of fun getting practical experience with our awesome hands-on labs. Thank you.

Course Updates

Hello, Cloud Gurus, and welcome to this lesson on course updates and changelogs. My name is Chris Silva, and I'm a content maintenance architect here at A Cloud Guru. Our courses are reviewed periodically to ensure our content remains current, and in this lesson, we're going to go over some of the changes related to this course. This course has been reviewed and updated to meet the current requirements of the AWS Certified SysOps Administrator Associate exam by updating lessons, labs, and quizzes and exams. Along with updating the content, some lessons, labs, or entire sections may have been replaced, removed, or shifted around to line up with this version of the exam, but rest assured that this course will cover everything you need to be successful in your exam. As such, this course has been deemed current as of March 2023. For more Information on the changes related to this course, please see the changelog that is attached to this lesson. That's it for now, and keep being awesome.

Supporting Courses

Supporting Courses

Hello cloud gurus, and welcome to this lecture, which is going to introduce some additional optional courses that you might be interested in. Now the Certified SysOps Administrator is an intermediate exam, and before taking this course, we assume that you have either had one year experience working with AWS, or that you have already passed the Solutions Architect Associate exam. So, if you have already done our Solutions Architect Associate course and passed the exam, then you are good to go. If not, go back and take that course first before moving on to the SysOps Administrator Associate. Now this course is designed to teach you everything that you need to know in order to pass the SysOps Administrator Associate exam. However, if you do feel like you need a helping hand with some of the topics, we have a set of great optional courses that are very relevant to the SysOps Administrator Associate exam and well worth considering if you feel like you have a weak area that you would like to focus on, and these are completely optional. So we've got a great course on Designing Resilient Architectures. We have one focusing on High Availability and Scalability; Logging and Security; Storage, Databases, and Migration; and Network and Computing as well, and all of these are optional. I just wanted to point them out in case you would like to spend extra time on a specific area or revisit some of the topics from the Solution Architect Associate. And in addition to these, we also have an excellent search engine on our website, so if you feel like taking a deeper dive into any of the topics within this course, just head to the Search box and search for the technology that you are interested in. For example, I'm going to search for s3, I'm going to filter by courses, and there we go, we have three additional courses that are dedicated to S3, including a deep dive course as well, and these courses will cover topics that are over and above what you need in the exam, particularly the deep dive. Let's also search for CloudFormation, which comes up quite a lot in the SysOps Administrator Associate course, and we also have three courses on CloudFormation, including Mastering CloudFormation and the CloudFormation Deep Dive, and once again, these courses are going to go over and above what you need in the exam. So that's all for additional courses. If you have any questions, please let me know. Otherwise, I will see you in the course. Thank you.

Deployment, Provisioning, and Automation

Section Introduction

Hello cloud gurus, and welcome to this section of the course, which is going to cover deployment, provisioning, and automation. And we'll begin with deploying EC2 instances and load balancers, creating an Amazon Machine Image, or AMI. We'll then cover provisioning resources using CloudFormation, deployment methods, and then finally, we'll also cover automation using Systems Manager, EventBridge, and AWS Config. So if you're ready to get started, I'll see you in the next lecture. Thank you.

Demo: Deploying an Elastic Compute Cloud (EC2) Instance

Hello Cloud gurus, and welcome to this lecture. Now the key to learning AWS is getting as much hands-on, practical experience as possible, and in this lecture, we're going to be getting our hands dirty deploying an EC2 instance, and we'll configure our instance as a web server. So we'll start off by launching an EC2 instance, and we'll explore the options that are available in the AWS Console. We'll configure our security group to allow HTTP traffic, and we'll be using a Bootstrap script to install httpd and create a simple webpage. And then finally, we can test that everything is working by attempting to access the website, using the public IP address of our EC2 instance. So if you'd like to get started, I'll see you in the AWS Console. So here I am in the Console, and the first thing I'm going to do is make sure that I'm in the Northern Virginia region, and this is the main region where AWS releases all their new services. So if you want to play with the latest and greatest services, then this is the region to select. So now, I'm going to head across to Services, and then you'll find EC2 under the Compute section. So select EC2, and I'm going to scroll down and select Launch instance, and instances are created using Amazon Machine Images, or AMIs, and this is basically the operating system. So, Windows, Red Hat, macOS, or Linux, etc., and there are loads of different ones to choose from. But in this course, we are going to be using the Amazon Linux AMI. So select the latest one, and at the moment, it's the Amazon Linux 2 AMI. And this is where we select our instance type. And we can select an instance type based on the requirements of our application. And as this is just going to be a little web server, we're going to stick with the default, the t2.micro, with 1 virtual CPU and 1 GB of RAM. So hit Next. This is where we can select the number of instances. We can select Spot instance here if we'd like to. Under Network, we're going to stick with the default VPC, and we'll be covering VPCs later on in the course, but for now, just think of it as your own virtual data center in the cloud that is private to you. So we'll stick with the default VPC and the default subnet as well. And we're going to cover subnets later on in the course as well. We'll auto-assign a public IP address. So this is going to create a public IP that we can use later on to connect to our web server. Moving down, we can accept all of the rest of the defaults. And then here, under Shutdown behavior, we can select whether we'd like our system to stop or terminate when we shut our instance down. And we can also protect against accidental termination by selecting this box. So this means that you won't be able to terminate this instance accidentally, and if you do want to terminate the instance, you will have to come back in and remove the termination protection. Moving down to Monitoring. All instances are monitored by default using CloudWatch. And by default, they give you basic monitoring free of charge, and they collect the monitoring metrics at 5-minute intervals. But if you would like more detailed monitoring, you can select it here. And with detailed monitoring, they will collect the data at 1-minute intervals, but they do charge you extra for that. In terms of tenancy, we'll stick with the shared tendency model. And

then if you scroll down to Advanced Details here in this text box under User data, this is where you can add a Bootstrap script, so basically any command that you want to run when the instance first boots up, and we can use a Bootstrap script to configure our operating system. So let's add a really simple Bootstrap script, and there is a link to this script in the resources section of the course. So the first line here just tells the operating system to use the bash interpreter to interpret all of the subsequent commands. Next, we'll run yum update -y to update all of the operating system packages. Then, we'll install Apache, also known as httpd. Next, we'll create a simple webpage, and that's going to be our index.html. Then we'll start Apache, and we'll use this command to set it to start every time this system boots. Okay, so that's our little Bootstrap script, and we want this script to run the very first time our instance boots. So I'm going to copy that script and paste it into my User data text box, hit Next, and this is where we can add our storage. And by default, they give you a single root volume, and it's going to be a General Purpose SSD volume, and that's going to be fine for the majority of applications, but if you are running a low-latency application, or let's say your application needs to have high-performance disks, then you can go for the Provisioned IOPS option instead, which will give you a faster disk. And we'll be covering these EBS volumes later on in the course. So now I'm going to hit Next. I'm going to add a tag, which is a really good way to organize your EC2 instances. So let's add a tag of Name, and I'm going to call my instance MyWebServer. Hit Next, and now it's asking us to configure a security group, and you can think of a security group as similar to a virtual firewall. So this is what we use to allow traffic into and out of our instance, and by default, they give us SSH access on port 22. So this gives us remote login using SSH, but in addition to SSH, we are also going to need to configure another rule. So first of all, I'm going to change the name of my security group so that I know what it's called, and I'll call it MyWebDMZ, then select Add Rule, and we're going to add a rule for HTTP. So select HTTP, the Port Range will be port 80, and the Source is going to be the world because this is going to be a web server, so we'll need to allow access from anywhere. And this is the notation that we use to define that for IPv4, and then this is simply the IPv6 notation, and this just means that this rule will apply to all IP addresses. Then in terms of SSH, we can keep it open to the world or I could choose to set it up to only allow my own IP address. And if this was your production system, then you'd probably want to lock it down to a specific IP address or network. So now, I'm going to hit Review and Launch, and Launch, and now it's asking us about key pairs. So if you wanted to connect to the system using SSH, you can create a new key pair, give it a name, and download your key pair and select Launch Instances, and there is our instance ID. So I'm going to select the instance ID, and you'll be able to see the status in this dashboard, and it might just take a minute or so to finish initializing. So just be patient, and in a few minutes you should be good to go. And you can just refresh this screen to get the latest status. And then once your instance has passed its status checks, then you are good to continue. So there is my EC2 instance. I'm going to select the instance ID, and this will give me a summary of my instance configuration. So now, we should be able to check if we can access our web page. So find your public IP address and select this icon to copy the IP address. I'll open a new tab in my browser and paste my IP address and hit Enter. And if it's all worked, then this is what you should see. And if you're not seeing this, then there's a few things that you can go back and check. So first of all, you can go back and make sure that you've correctly created the index.html, make sure that the httpd service is up and running, and also make sure that the security group allows access for anyone on port 80. So, there we go. We created an EC2 instance, we've installed the Apache web server, and we've created a little website that we can access over the internet using the public IP address. So, on to my exam tips for EC2. Just remember that EC2 is like a virtual machine hosted in AWS instead of in your own data center. You can select the capacity that you need right now, grow

and shrink that capacity whenever you need, you only pay for what you use, and you can be up and running in minutes rather than waiting months. So that's the end of this lecture. And if you have created this EC2 instance in your own account, then do remember to go in and terminate your instance as soon as you've finished to avoid any unnecessary charges. And to do that, just head to the EC2 Dashboard, make sure your instance is selected, come to Instance state, and terminate your instance. So hopefully all of that worked for you. And if you have any questions, please let me know; otherwise, I will see you in the next lecture. Thank you.

Understanding EBS Volumes

Hello cloud gurus, and welcome to this lecture where we're going to take a closer look at EBS volumes. So firstly, we'll look at what EBS volumes are, and we're then going to cover the different types of EBS volume available. And, we'll also consider use cases for each different type of EBS volume. And, for the exam, you'll need to understand the differences between the different types of EBS volume and when you would use them. So EBS stands for Elastic Block Store, and EBS volumes are storage volumes which you can attach to your EC2 instances. So here is our EC2 instance, and here is our EBS volume, attached to our EC2 instance. So they are storage volumes or disks, so just think of the disk in your laptop or in your local computer. And when you first launch an EC2 instance, it has at least one EBS volume attached, and this is where your operating system is going to be installed, so, Windows or Linux. And then you can add more volumes depending on the needs of your application. So, how do you use these EBS volumes? Well, you use them the same way you would any system disk. For example, you could create a filesystem and store some files in there, maybe configuration files or any files you'd like. You could use an EBS volume to run a database, run an operating system, store any type of data that you'd like, and you can install your applications on them as well. So let's take a look at some of the features of EBS, and one of the great things about it is that it's designed for mission-critical production workloads. But what makes it so great for production and mission-critical workloads? Well, one of the great things about it is that it's highly available, and when you provision an EBS volume, it's going to be automatically replicated within a single availability zone, which protects against hardware failures. Now, you don't need to configure this yourself, it's something that happens under the hood, so it's highly available and replicated by default. And in addition to that, EBS volumes are also highly scalable. So you can dynamically increase the capacity and even change the type of volume with 0 downtime and 0 performance impact to your live systems, and I think that's pretty cool. So while we're talking about capacity and volume types, let's run through the different options that are available, and you'll need to understand the different types of EBS volume for the exam. So let's begin with General Purpose SSD, and this is an SSD, so it's a solid-state disk, and it's also known as GP2. And with General Purpose GP2, you get a balance of price and performance, so it's a reasonable price and a reasonable performance, and you get three IOPS, or I/O operations per second per GB, up to a maximum of 16,000 IOPS per volume. And, GP2 volumes, which are smaller than 1 TB, can burst up to 3000 IOPS, and these are great for boot volumes or for development and test, and for applications which are not latency sensitive. And they have also released GP3, which is General Purpose SSD, and it's the latest generation. You get a baseline of 3000 IOPS for any volume size, and the volumes can be between 1 and 16 GB, they deliver up to 16,000 IOPS, and they are currently 20% cheaper than GP2 volumes. And just like GP2, they are great for boot volumes or development and test applications, which are not latency sensitive. But what if you need greater performance than 16,000 IOPS? What if your application is latency sensitive? Well, that's where

provisioned IOPS SSDs, or io1, comes in, and provisioned IOPS, or io1, this is the high performance option, and it's also the most expensive. So with io1, you get up to 64,000 IOPS per volume and you get 50 IOPS per GB, and this is the one to use if you need more than 16,000 IOPS for your application. You know, maybe you're installing an off-the-shelf application, which specifies in the system requirements that you need greater than 16,000 IOPS, then this is the one to choose. Or, if you've developed your own application, and in testing you discovered that you need greater than 16,000 IOPS, then this is the one to go for. So these are designed for I/O intensive applications, so think large databases and latency-sensitive workloads. Now, in addition to io1, they have recently announced io2, which is the latest generation of provisioned IOPS EBS volumes, and this may or may not come up in the exam, but it's just good to know anyway. So io2 is the latest generation for provisioned IOPS, and the main thing to note is that you get higher durability and more IOPS per GB, but it's up to the same maximum of 64,000 IOPS per volume. So it's more durable and more performant as well, but in terms of cost, it's actually the same price as io1, and you might be wondering why does it cost the same if it's so much better? Well, I don't know for sure, but my guess is that io1 will eventually become the legacy option, and io2 will become the standard. So by making them the same price, they're encouraging people to start using io2. So with io2, you get 500 IOPS per GB, and of course, with io1 it's only 50, up to a maximum of 64,000 IOPS per volume. And in terms of durability, io2 is designed for 99.999% durability, whereas all of the other EBS volume types are designed for between 99.8 and 99.9% durability. And, just like with io1, io2 is suitable for I/O-intensive applications, so think large databases and latency sensitive workloads, but also, applications which need higher levels of durability. So if you have an application which needs greater than 99.9% durability for its data, then io2 is the option to go for because this is the only option that's going to give you five nines of durability for your data. You may have heard about io2 Block Express, and this is a relatively new offering from AWS, and it is a SAN, or Storage Area Network, in the cloud, providing the highest performance and sub-millisecond latency. The underlying hardware uses EBS Block Express architecture, which is a special new architecture that they've introduced, and It provides four times the throughput, IOPS, and capacity of regular io2 volumes. It supports up to 64 TB and 256,000 IOPS per volume with 99.999% durability. So it really is SAN-level performance in the cloud. And, io2 Block Express is great for the largest, most critical high-performance applications like SAP HANA, Oracle, Microsoft SQL Server, and IBM DB2, so it's really good for the back-end storage for your really large mission-critical databases. It may or may not come up in the exam, but it is just a good one to be aware of. Now, the next type of EBS volume that you'll need to know about is Throughput Optimized HDD, and this is a hard disk drive, it's an HDD, it's not an SSD, so it's not a solid-state drive, and these are known as st1. And these are great for storing huge amounts of data, so think mountains of data, but you want to access the data frequently. And the performance of Throughput Optimized EBS volumes is measured in megabytes per second per terabyte. So, with these, you get a baseline throughput of 40 MBps per TB. You've also got the ability to burst to up to 250 MBps per TB with a maximum throughput of 500 MBps per volume. So when would we use these kind of EBS volumes? Well these are great for frequently-accessed throughput-intensive workloads, so think workloads like Big Data, data warehouses, ETL, so Extract, Transform, and Load operations, and log processing. So think large amounts of data that you are accessing frequently and you want high performance as well, and it's a really cost effective way to store mountains of data. So if you've got loads of data to store and its frequently accessed, then st1 is the one to choose, and these cannot be a boot volume, so AWS will not allow you to select one of these as a boot volume. And then finally, we have the lowest cost option, and this is COLD HDD, and it's also known as SC1. So SC1 has a baseline throughput of 12

MBps per TB with the ability to burst up to 80 MBps per TB with a maximum throughput of 250 MBps per volume. So it's nowhere near as performant as the throughput optimized st1, but it is a really good choice for colder data, so data which requires fewer scans per day. And it's also good for applications which need the lowest cost and for which performance is not a factor, so if you're running a job, which could happen overnight or across a few days, which involves accessing lots of data and you don't necessarily need a super fast response, then this could be a good option. And also, the COLD HDD SC1 cannot be a boot volume. Now, you might be wondering why are we suddenly talking about IOPS and throughput and what is the difference? Well, IOPS stands for I/O operations per second, and it measures the number of read and write operations per second, and it's a really important metric for applications which are performing quick transactions, so think low-latency applications and transactional workloads. So think about reading and writing to a database and making multiple changes to the data, and each of these changes can be thought of as a transaction. So this is all about the ability to action reads and writes very quickly. And If you do have an application like that, and particularly if you have an application which requires greater than 16,000 IOPS, then you should definitely choose provisioned IOPS SSD as your EBS volume. So what about throughput? Well, throughput measures the number of bits read or written per second, and this is an important metric for large datasets, large I/O sizes, and complex queries. So think about complex queries and large amounts of data, maybe reporting applications, long running jobs involving large amounts of data, maybe a business intelligence application comparing year-on-year financial performance, that kind of thing. So it's all about the ability to deal with large datasets. And if you have an application with those kind of requirements and with a specific requirement around throughput and the number of bits read or written per second, then Throughput Optimized HDD, or st1, that is the one to choose. So let's take a look at my exam tips for Elastic Blocks Store. So just remember, EBS volumes are highly available and scalable storage volumes which you can attach to an EC2 instance. And there are a few different types available, starting off with GP2, which is the General Purpose SSD, and this is suitable for boot disks and general applications, and it's capable of up to 16,000 IOPS per volume. And there's also GP3, which is the latest generation General Purpose SSD. Once again, it's suitable for boot disks and general applications, but with a baseline of 3000 IOPS for all volumes, and it is currently 20% cheaper than GP2. But if you need greater than 16,000 IOPS, then you need to go for io1, which is Provisioned IOPS SSD, and this is suitable for online transaction processing and latency-sensitive applications. You've got up to 64,000 IOPS per volume, and this is a high-performance option and it's also the most expensive. And, in addition to io1, there is also iot2, which is the latest generation for provisioned IOPS. It's also suitable for OLTP and latency-sensitive applications, but the difference with this one is that you get a greater number of IOPS per GB, so you get 500 IOPS per GB instead of the 50 that you get with io1 one, it's up to 64,000 IOPS per volume and you also get the greater durability. So with all the other volume types, it's up to 99.9% durability, but with iot2 you get five nines of durability and they're not going to test you on the percentage durability, so you don't need to remember that, but it's just good to bear in mind that iot2 is the one to go for if you have some specific requirements around data durability. And the price for iot2 is the same as io1. And, there's also io2 Block Express, which is also Provisioned IOPS SSD, and this is best for large mission-critical high-performance applications, so things like SAP HANA, Oracle, Microsoft SQL Server, and IBM DB2, so your mission-critical low-latency applications. It supports volume sizes of up to 64 TB, up to 256,000 IOPS per volume, and this is the AWS SAN in-the-cloud offering, so if you hear anything relating to a SAN in the cloud, then think io2 Block Express. Now, in terms of the Throughput Optimized options, we've got st1, also known as Throughput Optimized HDD, and this is suitable for Big Data, data warehouses,

and ETL. The maximum throughput is measured in MBps per volume, and the max is 500 MBps per volume, and It cannot be a boot volume, only the SSD options can be a boot volume. And then finally, we've got the lowest cost option, which is COLD HDD, also known as sc1, which has a maximum throughput of 250 MBps per volume, and this is suitable for less-frequently-accessed data and applications for which performance is not really an issue and you don't mind waiting a longer time for the results of your queries. And, once again, it cannot be a boot volume, and it is also the lowest cost, so if you have an application which is cost-sensitive, but it's not particularly sensitive to performance, then this could be a good option. And the last thing I wanted to show you is this page in the documentation for EBS volumes. And there's a link to this page in the Resources section of the course. And if we scroll down, we should find a table like this, and this table just compares all the different volume types in one easy place. And it lists the different characteristics and use cases for each different type of EBS volume. So it's got everything there at a glance, and I just wanted to show you that because it's just a different way of looking at the same information that we've been discussing in this lecture. So, that is everything that you should need to know about the different types of EBS volume. If you have any questions, please let me know; otherwise, feel free to move onto the next lecture. Thank you.

What Is a Bastion Host?

Hello, Cloud Gurus, and welcome to this lecture, which is going to cover bastion hosts. First of all, we'll cover what is a bastion host? We'll review an example scenario, and then we'll take a look at my exam tips. So what is a bastion host? Well, it's a public-facing instance which allows you to SSH or RDP to your private instances from an untrusted network. And a bastion host always has a public IP address, and it's in a public subnet connected to the internet or another untrusted network. And generally, a bastion host is security hardened, and it has any unnecessary services removed to reduce the attack surface. And that is just security best practice. So let's take a look at an example scenario. So here is my VPC, and we've got a private subnet with a private instance and a bastion host in a public subnet. We have an internet gateway and our public subnet has a route table with a route to the internet and a security group, which allows incoming SSH traffic. And then our private subnet has its own security group which only allows incoming SSH or RDP connections from the bastion host. So if we want to log into our private instance, we can do so by first connecting to our bastion host and then connecting from our bastion host to the private instance. And that is why a bastion is also called a jump box. So it's allowing you to jump from the public subnet to the private subnet. So onto my exam tips. And just remember a bastion host allows you to connect to your private instances in your VPC from an untrusted network like the internet using SSH or RDP. A bastion host is located in a public subnet and is reachable from the internet. And you will need to configure the security group associated with your private subnet to enable SSH or RDP access from the bastion. So that's it for this lecture. If you have any questions, please let me know; otherwise, I will see you in the next lecture. Thank you.

Exploring Elastic Load Balancer

Hello Cloud Gurus and welcome to this lecture where we are going to explore Elastic Load Balancer. And we'll be taking a look at what is a load balancer, the different types of Elastic Load Balancer available, and my exam tips as well. So, what is a load balancer? Well, a load balancer distributes network traffic across a group of servers. So if you think of a simple website consisting of three web servers behind a load balancer, the user browses to the website and the request hits the

load balancer and then gets routed to one of the web servers. And depending on how you've got it all configured, the load balancer might send the request to the web server, which is the least busy. Or it might use a round-robin algorithm and send requests to each server in turn. And the great thing about this architecture is that if one of the web servers fails, then the load balancer will notice and stop sending requests to this web server until it comes back online. And we can easily increase the capacity when needed. So let's say, for example, our website suddenly gets really popular. Then, we can add some more web servers and register them with the load balancer. Now with AWS Elastic Load Balancer, there are a few different options to choose from. So we've got the application load balancer, which load balances HTTP and HTTPS. There's also the network load balancer, which load balances TCP traffic, and this is the high-performance option. We've also got the classic load balancer, which can handle HTTP and HTTPS and TCP protocols as well, and this is the legacy option. And there's also the gateway load balancer, and this one is a fairly new offering, and it may or may not come up in the exam, but it's good to be aware of at a high level. And this one allows you to load balance workloads for third-party virtual appliances running in AWS, such as virtual appliances purchased on the AWS Marketplace, virtual firewalls from companies like Fortinet, Palo Alto, Juniper, or Cisco, and intrusion detection or intrusion prevention systems from companies like CheckPoint and Trend Micro, etc. So let's take a look at each of the main load balancers in a little bit more detail, beginning with the application load balancer. Now this is used for load balancing HTTP and HTTPS traffic, and it operates at layer 7 of the OSI or networking model, and it is application-aware. And if you haven't heard of the OSI model, it is also known as the 7 Layer Model, and it is a conceptual framework, which describes the functions of a network, beginning with the application layer, which directly serves the end user, right down to the physical layer and everything in between. So we've got layer 7, which is the application layer, and this is everything that the end user sees, and HTTP operates at this layer and so does your web browser as well. We have layer 6, which is the presentation layer. And this layer makes sure that the data is in a usable format, and protocols like encryption and SSH operate at this layer. Layer 5, also known as the session layer, is all about maintaining connections and sessions. Layer 4, also known as the transport layer, is all about transmitting data using protocols like TCP and UDP. And, of course, TCP is one of the main protocols of the internet. Layer 3 is also known as the network layer concerned with logically routing network packets based on IP addresses. Layer 2, the data link layer, is concerned with physically transmitting data based on MAC addresses. And then layer 1, which is the physical layer. This is all about transmitting bits and bytes over the physical devices that make up the network. And in the exam, you won't be tested on this 7-layer OSI model, but just remember that the application load balancer is HTTP-aware, and it is operating at layer 7 for the application layer of the network stack. Now as the application load balancer is application-aware, it supports advanced request routing. So that means that you can route requests to specific web servers based on the contents of the HTTP header. So just imagine a car dealership website, and I want you to think about the kind of services that you might find on this sort of website. So they'll definitely be selling cars, but they might also offer loans or credit agreements and service and repairs. And by using an application load balancer, we can send requests to specific web servers depending on what the user is looking for. So we can send sales inquiries to one set of web servers, loan applications can go to another, and if you're booking a service or repair, that can go to another set of web servers. And the application load balancer can handle all of this for you, and it can use the HTTP request header to determine where to send each request. Moving on to the network load balancer, and this is the high-performance option. So I want you to think of a super fast sports car, and this is used for load balancing TCP traffic when extreme performance is required. And it operates at layer

4 of the OSI model, which is the transport layer. So if we come back to our seven-layer model, it's at this layer, the transport layer, where TCP operates. So it's load balancing based on the TCP protocol. And it is capable of handling millions of requests per second while maintaining ultralow latencies. But just remember, as it is the highest performance, it is also the most expensive option available. Moving on to classic load balancer. And when you think of classic load balancer, I just want you to think of the classic car. So it's not going to be the fastest, and it's not going to have the most exciting features either. And it is the legacy option, but it may still appear in the exam. So classic load balancers, they do support some layer 7-specific features like X-Forwarded-For headers, which we'll cover in the next slide, and sticky sessions as well, which allow you to keep sending requests which originate from the same session to the same web server, making the session sticky. And the classic load balancer also supports layer 4 load balancing for applications, which rely purely on the TCP protocol. So let's circle back to the X-Forwarded-For header, and this is an HTTP header, which allows you to identify the originating IP address of a client connecting through a load balancer. So here is our client, and her IP address is 124.12.3.231. Here is our web server, and our client is connecting to the website through a load balancer. And let's say the load balancer has a private IP address of 10.0.0.23. When the request reaches the web server, the web server will only see this private IP address from the load balancer. So it's only going to see 10.0.0.23, and this might cause problems for your application because you might want to know where all these requests are coming from. For example, you might want to be sure that they're coming from a trusted network or that the request is coming from a location that you are allowed to operate in. So where can you get the IP address? Well, that is where the X-Forwarded-For header comes in, and the originating IP address is going to appear in this X-Forwarded-For HTTP header. And if we're using a load balancer, which supports X-Forwarded-For, then that means that we can identify the originating IP address. So, onto my exam tips for Elastic Load Balancer. First of all, we've got the Application Load Balancer, which load balances HTTP and HTTPS. And this provides intelligent load balancing, which allows you to route requests to a specific web server based on the HTTP request header. So think of the car dealership where they're offering sales and loan agreements and repairs. We've got the Network Load Balancer, which is the high-performance option for TCP traffic only, the Classic Load Balancer, which is the legacy option, and supports both HTTP and HTTPS and TCP as well, the Gateway Load Balancer, which is for third-party virtual appliances. And then finally, if you need to find the IPv4 IP address of your end user, then you can just find it in the X-Forwarded-For HTTP header. So if that is it for this lecture. If you have any questions, please let me know. Otherwise, I will see you in the next lecture. Thank you.

Understanding Elastic Load Balancer Error Messages

Hello Cloud Gurus, and welcome to this lecture, which is all about Elastic Load Balancer error messages. First of all, we'll define server-side and client-side errors. We'll take a look at the HTTP 504: Gateway Timeout error. We'll cover other common error messages that you might see and also exam tips. Now when I'm troubleshooting any issue with a web-based application, I always ask myself, is this problem server side or client side? So is there a problem with your application, or is it a problem with the incoming request? And let's say, for example, you've got an issue on one of your web servers or an issue with your database. Well, that would be a server-side issue. And with server-side issues, you will generally see a 500 error code, and you might see that in your browser or in the error logs of the load balancer. Whereas, if there's a problem with the incoming HTTP request, well, that is a client-side issue. And with client-side issues, we see 400 error codes. And

once again, you are likely to see this error code in the browser or on the load balancer as well. And to help you remember, I always think the number five looks a little bit like the letter S, and S is for server-side errors. So whenever you see a 500 error message, I want you to remember that it is a server-side error and likely to be something wrong in your application. Now one of the most common errors you will see when using the internet is HTTP 504: Gateway Timeout. And in relation to an Elastic Load Balancer, you might also see the message, Gateway Timeout. And this simply means that the target of the load balancer failed to respond, so you will need to check your application. Now under the hood, this means that the Elastic Load Balancer could not establish a connection to its target, so it couldn't connect to the web server, database, or lambda function that was its target. So your application is having issues, and you will need to identify where the application is failing and go in and fix the problem. And I actually experienced a 504 error only recently. I was making a payment online, and the payment took too long to authorize, so I got this Gateway Timeout message. And I don't know what exactly went wrong. But what I think happened is that my bank was too slow to authorize the payment, and I got this Gateway Timeout. So my transaction failed, and I had to try again, so Gateway Timeout is an error that you might see when you're attempting to make a payment online, and the request may time out if the bank system takes too long to respond. So let's take a look at some other common load balancer errors. First of all, we've got 502, and it's sometimes accompanied with the message bad gateway, and this means that the target host is unreachable. And if you see this message, then the first thing you'll want to check is whether traffic is allowed from the load-balancer subnets to the targets on the target port, so check that your security groups are not blocking traffic between the load balancer and the targets. And there's also HTTP 503: Service unavailable. And this means you have no registered targets, so just go in and check that you have correctly registered your targets. But what could possibly go wrong on the client side? Well, you might see HTTP 400: Bad request. And this means that the request is malformed, maybe the header is too long, or there's something missing. There's also HTTP 408: Request timeout. And you might see this error if the client did not send data before an idle timeout period expired. So imagine you're buying tickets for a concert online, you reserve the tickets, but you need to complete the payment, say, within 15 minutes. So the website is waiting for a response from you, and you might see a request timeout if you fail to respond within the given time period. And then finally, there's HTTP 464, and this is an error that you might see if the incoming request protocol is incompatible with the target-group protocol. So you're basically sending in a request using an unsupported protocol. So those are just a few examples of what could go wrong on the client side. But as SysOps Administrators, we are mainly concerned with what is going on server side, so we are more concerned with the 500 errors. So on to my exam tips. And when I'm considering load balancer error messages, I try to determine whether the problem is client side or server side. And just remember, 400 is a client-side error, and 500 is a server-side error. And one of the most common error messages that you are likely to see on the internet is HTTP 504: Gateway Timeout. And this is when the Elastic Load Balancer could not establish a connection to its target, and something has gone wrong in your application, so you will need to identify where the application is failing and fix the problem. Then there's HTTP 502: Bad Gateway. And this means that the target host is unreachable, so double-check that your security groups are allowing traffic from the load balancer to the target hosts. And then finally, we have HTTP 503: Service Unavailable. There are no registered targets, so do check that you have remembered to register your targets. So that's it for this lecture. If you have any questions, please let me know. Otherwise, I will see you in the next lecture. Thank you.

Demo: Deploying an Elastic Load Balancer

Hello, Cloud Gurus, and welcome to this lesson, where we'll be deploying an Elastic Load Balancer. And we'll begin by launching two EC2 instances, each in two different availability zones. And we'll configure our EC2 instances as web servers. Next, we'll create an Elastic Load Balancer, and we're going to register our two EC2 instances as target hosts for the load balancer. And then finally, we're going to test our website, so we'll try and access the website using just the DNS address of our Elastic Load Balancer. So if you'd like to get started, please join me in the AWS Console. So from the console, search for EC2 (Working) and select Launch Instance. The name is web-server-1, operating system is Amazon Linux, Instance type is t3.micro. We're going to proceed without a key pair because we don't need to log into our instance. On the Network settings, select Edit, our Subnet is going to be the subnet located in us-east-1a, so select that one. Scrolling down to security group, I'm going to rename our security group so that it's easier to find later on. Scrolling down to the security group rules, we don't need ssh, so I'm going to change that to HTTP. Protocol is TCP, port range is 80, and source type is Anywhere. Scrolling down to advanced settings, open the drop-down, and scroll down until you come to the User data section. And this is where we're going to add our bootstrap script to configure our instance as a web server. And you'll find a link to this script in the resources for this lesson. And here it is. I'm just going to select the Raw option. So there's our script, and you can just copy everything and paste it into the text box. So paste that into your User data section. So this script just starts off by telling the operating system to use the bash interpreter. It does a yum update -y to update the operating system, installs httpd, creates a simple webpage that just says Hello Cloud Gurus, then starts httpd, and enables httpd to start at boot time. So once you've added your script, we are ready to launch our instance. So hit Launch Instance. That's our first instance launched, so now let's go ahead and launch the second one. This time, in us-east-1b. So I select Instances and Launch Instances. The name is web-server-2, operating system is Amazon Linux, instance type is t3.micro, proceed without a key pair, then edit your Network settings. Under subnet preference, select the subnet that is located in us-east-1b. On the security group, we'll select an existing security group, and I want you to select the one we created in the first step. And there it is, my-web-dmz. Then scroll down to Advance details and open up the drop-down. Come to the User data section, it's down at the bottom, and we're going to paste in our bootstrap script again. Once you've done that, hit Launch Instance. So now I'll select Instances from the top. Here's our two instances, and we'll need to wait a few moments for our instances to finish initializing. But while we're waiting, we can go ahead and create our Elastic Load Balancer. So if you come to the menu on the left-hand side, scroll down to the Load Balancing section and select Load Balances, Create load balancer. And if you remember, there are a few different types of Elastic Load Balancer to choose from. So first of all, we've got the Application Load balancer, which load balances for HTTP and HTTPS, and it also provides advanced routing, which allows you to route requests based on the content of your HTTP header, and it is application aware. Next, there's a Network Load Balancer, which is the ultra-high performance option, and this is the one to use if you need to handle millions of requests per second with ultra-low latency, so this one is great for a low-latency application or a latency-sensitive application. We've also got the Gateway Load Balancer, and this one allows you to load balance for third-party virtual appliances. And then there's also the Classic Load Balancer as well, which is the previous generation, and it supports load balancing for HTTP and HTTPS, and TCP as well. But as we are load balancing HTTP, we're going to use an Application Load Balancer. So go ahead and select that one, and hit Create. Call it my-load-balancer. It's going to be internet-facing using Ipv4. Under Network mapping, it should

have selected your default VPC. And then in the Mappings section, select our two subnets, so we need to select the subnet in us-east-1a and the one in us-east-1b because that's where our two EC2 instances are located. Scrolling down, we'll select the same security group that we created earlier, my-web-dmz, and I'm going to remove the default security group. Our listener is going to listen for HTTP on port 80. And then under target group, this is where we need to create our target group. So select Create target group, and it will open up in a new tab. And the target group is what defines a group of servers that our load balancer is going to load balance traffic for. Target type is going to be EC2 instances, but it could also be an IP address, lambda function, or another application load balancer. Call it my-target-group. Protocol is HTTP on port 80. It's selected our default VPC. Protocol version is going to be HTTP1. Under Health checks, the load balancer is going to periodically send a request to the health-check path that we specify below to test the status of our registered targets. So down here, we'll specify the name of our web page, and its going to be /index.html. And you don't need to write the full path to this file because the Elastic Load Balancer is going to check for the file in the default location, which is var.www.html. And for the health check, the Elastic Load Balancer will simply check that it can reach this file successfully. So now scroll down and hit Next, and this is where we're going to select our targets and register them. Here's our available instances, so we've got our web-server-1 and web-server-2. So select both of those. It's confirming the port is port 80. And then you need to select Include as pending below. Scroll down, and you can review the targets that we selected. They should both be showing as pending because we haven't added them yet, and now select Create target group at the bottom. So that is our target group created. And now we need to go back to our previous tab, the Load Balancer tab, and you might need to just refresh using this button. Once you've done that, you can use the drop-down to select the target group we just created. And there it is. After that, just scroll down. Here's the summary of our configuration, and select Create load balancer. Now it might just take a few minutes to finish creating everything, so select View load balancer, and we can see the status. And once everything is ready, we should be able to test that it's all working. Once your load balancer is showing in an active state, you may need to refresh using this button to refresh the view. You can then select your load balancer. Here's the D0NS name of the load balancer. So copy that, and I'm going to paste it into a new browser tab, and hit Enter. And if everything worked, then this is what you should see. So we are accessing our website, running on EC2 instances, using the DNS name of our Elastic Load Balancer. So that is the basics of Elastic Load Balancer. And I want you to keep your load balancer and your EC2 instances up and running because, in the next lesson, we're going to go straight in and take a look at how we can monitor our load balancer using CloudWatch, and we'll take a look at the different CloudWatch metrics that are available, so don't shut anything down just yet. And for my exam tips, just remember there are a few different types of Elastic Load Balancer to be aware of. So we've got Application Load Balancers, which load balance for HTTP and HTTPS. They allow you to route requests to a specific web server based on the type of requests defined in the HTTP header. We've got Network Load Balancers, which are the high-performance load balancing option for TCP traffic, and these are great for low-latency applications. We've got the Classic Load Balancer. It's the legacy option, but it is still supported, and you may see it mentioned in the exam. And it supports both HTTP and HTTPS, and TCP as well. And then there's Gateway Load Balancers, and these provide load balancing for third-party virtual appliances, like firewalls, intrusion detection, and intrusion-prevention systems. Well, that's it for this lesson. But do remember to keep your Elastic Load Balancer up and running and your EC2 instances as well because we're going to go straight into looking at the CloudWatch metrics for Elastic Load Balancer. So if you'd like to do that, please join me for the next lesson. Thank you.

Demo: Understanding Elastic Load Balancer CloudWatch Metrics

Hello cloud gurus, and welcome to this lesson, where we'll be taking a look at Elastic Load Balancer CloudWatch Metrics. Now, by default, Elastic Load Balancer sends metrics into CloudWatch, and in this lesson, we're going to be using the same Elastic Load Balancer and EC2 instances that we used in the previous lesson, so hopefully you did not terminate your Elastic Load Balancer or your EC2 instances. And if you did, then you will need to go back to the previous lesson and quickly recreate those. Next, we're going to generate some traffic, so we'll generate some traffic to the load balancer by visiting our website a few times, and that is going to cause our Elastic Load Balancer to generate some metrics and send them into CloudWatch. And then finally, we'll review the metrics, so we'll review the CloudWatch metrics to see the information relating to our requests that have been sent to the Elastic Load Balancer. So if you'd like to join me in the console, we'll get started. So in the console, search for ec2 and select EC2. On the left-hand side, this is where you're going to find load balancing, so select Load Balancers. Here's my-load-balancer that I created in the previous lesson, and I'm going to copy the DNS address of my-load-balancer, so copy that. And I'm going to open up a few new browser tabs, and paste in the address of my-load-balancer, just to generate some traffic to my-load-balancer. So now, once you've done that a few times, come back to the AWS console. I'm going to select my-load-balancer, then down here, select Monitoring. And this Monitoring tab in the Elastic Load Balancer is going to show you the CloudWatch metrics that are automatically generated for your Elastic Load Balancer, and all of these metrics are coming from CloudWatch. So we've got the Target Response Time in milliseconds, we've got a Requests count over here, we've also got details of any HTTP status codes, Target connection errors, rejected connections, and TLS Negotiation Errors. Here we've got the Active Connection Count, New Connection Count, Processed Bytes, and Consumed Load Balancer Capacity Units as well. And you can just view all of these from the elastic load balancer console. But if you want to view all of the CloudWatch metrics associated with this load balancer, you'll need to head to CloudWatch. So in the Search box, search for CloudWatch, and select CloudWatch. On the left-hand menu, select All metrics. Then in the Metrics section, select ApplicationELB and select Per AppELB, per AZ, per TG Metrics. And this is where you're going to find the metrics for all of the Application Load Balancers in your account, and I'm going to search for my-load-balancer. So these are all the metrics that are associated with my Application Load Balancer. And if you select one of these metrics down here, I'll select RequestCountPerTarget, it's showing separate metrics for us-east-1a and us-east-1b. So if you select those, then the metric is going to appear up here in my graph. And I'm just going to select the last hour up here so we can really drill down on the metrics. So if we scroll down, we can see the type of metrics that are available. So we've got things like UnhealthyHostCount, RequestCountPerTarget. We've also got HealthyHostCount, RequestCount, TargetResponseTime, HTTP status codes, and they've split everything by availability zone as well, so you can view the metric for either us-east-1a or us-east-1b. And every time you select one of these metrics, it's going to appear in the graph above so that you can easily view it. And if you haven't used CloudWatch very much, then don't worry, we're going to be covering it in a lot more detail in the Monitoring and Remediation section of this course. I just wanted to show you the kind of metrics that you can collect specifically for your Application Load Balancer. So for the exam, just remember, there are lots of different metrics that you can collect with CloudWatch for your Elastic Load Balancer, and the main ones that you should remember are HealthyHostCount, UnhealthyHostCount, RequestCount, TargetResponseTime, and the HTTP status codes as well. And in the next lesson, we are going to take a look at Elastic Load

Balancer access logs, so please do not delete this Elastic Load Balancer or EC2 instances just yet, because we've got one last lesson that's going to use these. So if you're ready to dive into the access logs, I will see you in the next lesson. Thank you

Demo: Working with Elastic Load Balancer Access Logs

Hello cloud gurus, and welcome to this lesson, which is going to cover Elastic Load Balancer Access Logs. Now, access logging is an optional feature of Elastic Load Balancer and it is not enabled by default, so we will need to go in and enable it. And we'll be using the Elastic Load Balancer that we created in our previous lesson. And hopefully you haven't terminated that Elastic Load Balancer yet, and if you have, you will need to go back to the lesson where we created our Elastic Load Balancer and two EC2 instances, and recreate them. So after enabling access logging, we are going to generate some traffic by visiting the website a few times, and that is going to cause Elastic Load Balancer to generate some access logs, which are going to be stored in an S3 bucket. And then after a few minutes, we'll be able to review the access logs. So we'll review the logs to see the information relating to our requests to the load balancer. So if you're ready to get started with Elastic Load Balancer access logs, I'll see you in the AWS console. So from the console, we can actually search for our Elastic Load Balancers up here. So search for load balancers, and you'll find it down here under Features. Make sure that your Elastic Load Balancer that we created earlier is still available, and if it isn't, then go back to the lesson, it's a couple of lessons before this, where we created the load balancer, and you'll need to recreate it along with the EC2 instances. If your load balancer is there, select it. Under Actions, select Edit load balancer attributes. On this page, scroll down until you get to Monitoring, and then it's here that we can enable Access logs. And we're going to use this radio button to enable it. And it's got quite a nice description down here, which says that Access logs deliver detailed logs of all requests made to your Elastic Load Balancer, and you will need to choose an existing S3 location where the logs are going to be stored. And it's just letting us know if we don't specify a prefix, then the access logs will be stored in the root of the S3 bucket. So we do need to create an S3 bucket to store our logs in, and to do that, we can select View, and it's going to open up S3 for us in a new tab. Select Create bucket. I'm going to call it my-elb-access-logs, and then add some random numbers on the end. Then scroll to the bottom and Create bucket. Now once we've created our bucket, we also need to configure a bucket policy to allow the Elastic Load Balancing service to write logs to our S3 bucket. So select your bucket, select Permissions, scroll down to Bucket policy and select Edit, and I've provided the policy that we're going to use in the resources for this lesson. And here it is. I'm going to select the Raw version, copy everything, come back to my S3 screen, and paste in the policy. Now with this policy, the Effect is going be Allow, the Principal is going be the Elastic Load Balancing service in us-east-1, and this is the ARN of that service. The Action is PutObject, so we're allowing the Elastic Load Balancing service to upload a file to our S3 bucket. And then this section here specifies the Resource, and you will need to replace this Resource ARN with the ARN of your bucket, and you will find that up here. So copy your Bucket ARN from up here, then we're going to replace this section with our Bucket ARN, and please remember not to delete these two last characters, so we need the foward slash and the star at the end of our Resource ARN. So when you're finished, it should look like this. So now, scroll down to the bottom of the screen. If you see this message, you can just ignore it, it's just about the access-analyzer, which we're not using in this demo so it won't affect what we're doing here. And then you can just Save changes. After you've edited your bucket policy, you can then come back to the Load Balancer tab, select Browse S3. I'm going to refresh my

view, and there is the S3 bucket I just created. So select your bucket and select Choose. And then once you've selected your bucket, you can go ahead and Save changes. So now we need to generate some traffic to our Elastic Load Balancer. I'm going to copy the DNS name and open a few new browser tabs, and paste the DNS name of my Elastic Load Balancer. And I'm just going to do that a few times to generate some traffic to my Elastic Load Balancer. And then if we head back to the S3 bucket, head to Objects, I'm going to refresh using this button. Here's my AWSLogs folder, so I'm going to go into there, and drill down. The ELB LogTestFile is a good sign because that means that the Elastic Load Balancing service is able to write to our bucket successfully. But it can take up to about 5 or 10 minutes before the logs can be published to the S3 bucket, so now is a good time to take a break, have a cup of tea, and have a short break away from the screen if you feel like it. And then come back in a few minutes, and we should see some logs appearing in our S3 bucket. And after a few minutes, I'm going to just refresh this view. We've got a new folder here called elasticloadbalancing, so click on that, select our region, and then it's storing the logs by date. So keep going down into the folders, and there is my latest log file. So I'm going to select the file, select Download, and it will download the log to your local machine, and it will usually end up in your Downloads directory. And just notice down here under the Server-side encryption settings, you will see the that the log is encrypted by default because this is sensitive information. So now, I'm going to select my log file, and here is my access log. Here's the name of my Application Load Balancer. This IP address here is the IP of my local machine, and then 172.131.21.164 is actually the private IP address of one of my EC2 instances, so one of my web servers. And then over here is the HTTP GET request that was generated by my local machine, so it's a GET request to the Application Load Balancer on port 80. And it even tells you what operating system is running on the machine that generated the request. And then here is the ARN of my target-group, so that's the target-group that the Elastic Load Balancer forwarded the request to. And that is Elastic Load Balancer access logs. So on to my exam tips. Just remember that the Elastic Load Balancer access logs capture information that is relating to the incoming requests to your Elastic Load Balancer. They are disabled by default, and you will need to go in and enable access logs and specify an S3 bucket where the logs are going to be saved. And finally, after the logs are generated, they're going to be encrypted by default and stored in an S3 bucket, and then they get decrypted when you access them. And you can just download them to your local machine and view them there. So that is it for this lesson. If you have any questions, please let me know. Otherwise, I will see you in the next lesson. Thank you.

Understanding Sticky Sessions

Hello Cloud gurus, and welcome to this lecture, which is going to cover sticky sessions. And we'll first review the different load balancing algorithms that are available, we'll cover what are sticky sessions, when to use them, and my exam tips as well. Now by default, an application load balancer routes all requests independently according to the load balancing algorithm. And the default load balancing algorithm is called round robin, and round robin distributes the requests to each target in turn so you get an even spread of requests across all of your targets. And this is great if the requests are similar in complexity and your targets are all the same size. For example, if they are all the same EC2 instance type. Alternatively, you can also configure the least outstanding requests algorithm, and this will send requests to the web server with the least outstanding requests. So let's say that we've got three web servers, and the first one has only got one outstanding request, the second one has two outstanding requests, and the third one has five. Well, the least outstanding request

algorithm will send the next request that comes in to our web server with only one outstanding request. And this is great if the requests vary in complexity or if your targets are not the same size, but what if, after the initial session is established, you want to keep sending requests to the same server? For example, think about applications which cache their session information locally on the web server. Consider a shopping cart, an online form, or even a training website, and I'm sure you'll agree that it's very annoying to find that you've been suddenly logged out or lost your session state and you need to start all over again. The last thing that you want is for your customers to be frustrated and wondering why they've suddenly been logged out or they've lost their session state. And that is where sticky sessions comes in, and they are also known as session affinity. So with sticky sessions enabled, you can actually override the load balancing algorithm and it uses a session cookie, which identifies that the session belongs to the same user. And while the cookie is valid, all requests from the same user are sent to the same target, which holds the session data, so your user does not lose their session. When should we use sticky sessions? Well, they are a great option to use when your application stores session data locally on the web server and you want to send requests that are part of the same session to the same target. So for my exam tips, we use sticky sessions to override the load balancing algorithm. They use cookies to identify a user session and send requests that are part of the same session to the same target. They're a good option to use for applications which cache session information locally on the web server. Consider shopping carts, online forms, or even a training website where we don't want to log out our customers or send them to a different web server when they're halfway through a task. So that's it for this lesson. If you have any questions, please let me know; otherwise, I'll see you in the next lecture. Thank you.

Load Balancing Based On An IP Address

Hello cloud gurus, and welcome to this lesson, which is going to cover how to configure an Elastic Load Balancer to load balance based on IP addresses. And we'll begin by creating two EC2 instances, and these are going be configured as web servers, one in us-east-1a and one in us-east-1b. Next, we'll create an Application Load Balancer, and our EC2 instances are going to be the targets of this load balancer. And when we create our target group, then I want you to notice that it's possible to register targets using either the instance ID or their private IP address. And lastly, we're going to test out our website to make sure that everything works, and we'll try to access the website using the DNS address of our Elastic Load Balancer. So if you're ready to get started, I'll see you in the AWS console. So from the console, search for ec2, Launch instance. The Name is Web server 1, operating system is Amazon Linux, Instance type is t3.micro. We'll proceed without a key pair, edit your network settings, and the Subnet is going be the subnet located in us-east-1a. We will rename our Security group to my-web-dmz, we'll add a new Security group rule allowing HTTP access on port 80, and the Source is going to be Anywhere. And I'm going to remove the rule for SSH, because we don't need that one. Scrolling down to Advanced details at the bottom, find the User data section, and it's right at the end. There it is. And we're going to add a little Bootstrap script, and you'll find a link to this in the resources for this lesson. So first of all, we're going to tell the operating system to use the bash interpreter. Next we're doing a yum update -y to update the operating system with the latest packages. We'll install httpd, create a simple web page called index.html, then we'll start httpd and enable it to start at boot time. So paste that script into the User data section, and go ahead and Launch instance. So that is our first instance launched, let's launch the second one, this time in us-east-1b. So select Instances, Launch instance. (Waiting) The Name is Web server 2. We'll use Amazon Linux, Instance type is t3.micro, Proceed without a key pair, edit

the network settings, and this time we're going to select the subnet that is located in us-east-1b. We'll use the existing security group that we just created, my-web-dmz, scroll down to Advanced settings and User data, paste in your User data script, and Launch instance. So now, while our instances are launching, let's go ahead and create the load balancer. So on the left-hand pane, scroll down, and this is where you'll find Load Balancing. So select Load Balancers and Create load balancer. We'll use an Application Load Balancer, so select Create, call it my-load-balancer. It's going be Internet-facing, using IPv4 addresses. It's already selected our default VPC, and then down here under Mappings, we need to select at least two availability zones that we want our load balancer to operate in, and it's going to be us-east-1a and us-east-1b, so the subnets where we created our instances. Scrolling down, I'm going to change the Security group to my-web-dmz and delete the default one. Our listener is going be HTTP protocol on port 80, and this is where we need to create our target group. So select Create target group, and it's going to open up in a new tab, and here we have a choice. So we can define a target group of instances, IP addresses, a lambda function or even another Application Load Balancer. And we're going to select IP addresses. And the great thing about using IP addresses is that this facilitates routing to multiple IP addresses and network interfaces on the same instance. So if we had multiple IP addresses on the same instance, this is the way that you would configure the load balancing if you wanted to load balance traffic to different IP addresses on that same instance. Down here under Target group name, we'll give it a name. Protocol is HTTP on port 80, IP address type is IPv4. Here's the VPC that's going to be hosting the load balancer. Protocol version is HTTP1. Then down here under Health check, we can configure our Health check path, so this is the path it's going to use to check that the registered targets are in a healthy state. So our path is going be index.html. Then select Next, and this is where we're going to register our targets. The network should be your default VPC, and then down here, this is where we need to specify the IP addresses that we want to use, so we'll need to find the private IP addresses of our EC2 instances. And to do that, I'm going to open up the EC2 console in a new tab. So search for EC2, open the link in a new tab. Here's our EC2 dashboard. Select your running instances, so we've got Web server 1 and Web server 2. I'm going to select the first one and grab the private IP address, come back to our Target groups screen, and paste in my private IP address. We've got another one to add, so select Add IPv4 address. Come back to your EC2 instances screen, select Instances, and we want to get the private IP for Web server 2. So grab that one, back to our Target groups, and paste in the IP address there. Now these two addresses we've added, these are the private IPs, but what do you think will happen if we try to add a public IP address? Well, let's try it. Select Add IPv4 address, come back to our EC2 instance, and this time I'm going to grab the public IP and see what happens. Place that in there, and then select Include as pending below. And It's not going to let me, because it's looking for an IP address that belongs to a private subnet, and it's not going to let you add a public IP address, so let's remove that one. We'll select Include as pending below, and if you scroll down, we can review our targets and Create target group. So now that's our Target group created, and we need to finish off creating our load balancer, so come back to your Load Balancer tab. You might need to refresh using this button so that you can select your target group, and you should be able to find it in the drop-down, and there it is. And then after that, just scroll down to the bottom and Create load balancer. Now it might just take a few minutes to finish creating everything, and then once it's ready, we'll be able to test out that it's all working. So select View load balancer, and as soon as it's finished provisioning, we are good to go. Refresh my screen, and a few minutes later it should be showing a state of Active. So now it's all up and running, I'll select my load balancer. Here's the DNS name, just copy that, and paste into a new browser tab. Hit Enter, and there we go. If it's all worked, you should be able to see your web page. So for the exam, remember

that an Elastic Load Balancer allows you to load balance traffic to targets based on their IP addresses. When we registered our EC2 targets, we can use either the instance ID or the private IP address, and the kind of use cases that this is going to be great for are EC2 instances that have multiple IP addresses, and of course any resources that are accessed using a private IP, for instance, an RDS instance. So that is it for this lesson. Any questions, please let me know. Otherwise, please join me in the next lesson. Thank you.

Discovering EC2 Image Builder

Hello, Cloud Gurus, and welcome to this lecture, which is going to cover EC2 Image Builder. So first of all, we'll take a look at what is EC2 Image Builder? Why is it so cool? How does it work? We'll also take a look at the terminology and exam tips. So what is EC2 Image Builder? Well, it allows you to create virtual machine images, and in AWS, they're known as Amazon Machine Images, and container images as well. It's really simple to use with a graphical interface, and once you have created your image, you can use EC2 Image Builder to test and validate the image, for example, for security compliance and functionality. And you can use AWS-provided tests or you can develop your own custom tests. So why it so cool? Well, the great thing about EC2 Image Builder is that it automates the process of creating and maintaining your images. And when software updates become available, Image Builder can automatically create a new image, run validation tests on the new image, and make it available to the AWS regions of your choice. And it also allows you to share your AMIs with other AWS accounts that you own. So how does it work? Well, the first step is to provide a base operating system image, for example, the Amazon Linux 2 AMI. The next step is to define the software that we want to install, for example, you might want to install .NET, Node.js, or Python, the latest security updates, the latest kernel, or security settings. And then in the third step, this is where Image Builder runs tests on the new image, for example, testing whether the image will boot correctly. And to do this, it will spin up a new EC2 instance using the new image, and it will run tests on this EC2 instance. And then the fourth step is to distribute. So Image Builder is going to distribute the image to the regions of your choice, and by default, it will distribute the image to the region that you are operating in. So let's take a look at some of the EC2 Image Builder terminology, beginning with Image Pipeline. And the Image Pipeline defines the configuration and end-to-end process of building images, including an image recipe, distribution, so which regions you would like to distribute your image to, and test settings as well. So what is the image recipe? Well, Image Builder creates a recipe for each image, which can then be shared, version controlled, and reused. And this will include a source image, for example, the Amazon Linux 2 AMI and build components, so the software that we want to install on our image, for example, Apache Tomcat. And build components simply means the software components that we are including in our image. So onto my exam tips. And just remember, EC2 Image Builder automates the process of creating and maintaining AMI and container images. It's a four-step process, so we begin with selecting a base operating system image like the Amazon Linux 2 AMI. We customize it by adding software, for example, Apache Tomcat. We run tests, for example, testing that we can boot an EC2 instance using the image. And then finally, Image Builder will distribute your image to your chosen regions. And then in terms of the terminology, image pipeline is used to describe the Image Builder settings and process, the image recipe defines the source image, and the build components are the software that you want to include in the image. But the best way to understand EC2 Image Builder is to actually use it. So if you're ready to get your hands dirty with EC2 Image Builder, I will see you in the next lecture. Thank you.

Demo: Creating an AMI Using EC2 Image Builder

Hello Cloud Gurus and welcome to this lesson where we'll create an AMI using EC2 Image Builder, and we'll begin by creating an IAM role. And we'll add the permissions for EC2 Image Builder, and this is going to allow Image Builder to do everything that it needs to run the build and test processes on EC2 instances. Next, we'll create an image pipeline, and this is where we're going to define all of our configuration settings. Then we'll execute our pipeline, and the output of our pipeline will be an AMI. And notice that Image Builder launches temporary build and test EC2 instances to run the build and test workflows. And then finally, once the AMI has been created, we can view it in the Image Builder console and also from the EC2 console under AMIs. So if you're ready to create an AMI using Image Builder, I'll see you in the console. So from the console, first of all, search for IAM. Then, select Roles and Create role. Make sure that your trusted entity type is an AWS service, scroll down to Use case, and select EC2 under Common use cases. Then hit Next, and we need to search for a couple of managed policies. Now the first one is the EC2 instance profile for Image Builder. So I'm just going to search up here for Image Builder. And this is the one that we're looking for, EC2InstanceProfileForImageBuilder. So select that one, and now I need to search for the Amazon SSM managed instant core policy. So clear your filter and then search for SSM. And this is the one that we're looking for, the AmazonSSMManagedInstanceCore policy. And we need this one because Image Builder actually utilizes SSM under the hood. So once you've selected those two policies, you can scroll down to the bottom and hit Next. We'll give our role a name, and I'm going to call it MyImageBuilderRole. Scroll down and you can check the permission policies that we've added and then select Create role. So now we've created our role, we are ready to create our image pipeline. So from the search box, just search for Image Builder, and there it is. Then select Create image pipeline. I'm going to call it My Image Pipeline. Scrolling down, I'm going to deselect this option to collect enhanced metadata, and this option just uses the Systems Manager Inventory to collect information about the images. And it requires some additional Systems Manager configuration, so I'm just going to deselect it so that everything works. Under Build schedule, you can create a schedule to automatically run the pipeline on a schedule that you define. But we're just going to create a pipeline manually, so select Manual. And this just means that the pipeline will run when we initiate it. Once you've done that, go to the bottom and select Next. For our configuration options, we're going to create a new recipe. And just remember, the image recipe is a document that defines the components that are going to be applied to our source image, so what is going to be included in our source image. Scrolling down, we can select our image type, and it's going to be the Amazon Machine Image. But you can also use Image Builder to create Docker images as well. Down here, we'll provide our recipe name, and we need to provide a version number as well using this format that they define. So type 1.0.0, and it must be this format. Otherwise, it's not going to work. Scrolling down to base image, this is where we can select the source image that we'd like to use, and we'll just stick with the defaults here. So make sure that you've selected managed images. We'll be using Amazon Linux 2, so that's going to be our operating system. Scrolling down, the image origin is going be quick start, Amazon-managed. Under the image name, we're going to select the x86 version, so select that one. And we'll use the latest available operating system version. Down here, it says the EC2 Image Builder uses the AWS Systems Manager agent as part of the image build process, and it's installing it automatically if it wasn't already included in the base image. And you can select this box if you would like Image Builder to remove the SSM agent after the pipeline has been executed. Scrolling down to working directory, and this is a temporary working space that Image Builder uses during the build and test workflows. And by default, it just

uses /tmp, so we'll stick with that default. Under Components, this is where we can select the software or scripts that define the custom configuration of our image. And there are loads of different build components that we can select. But for our image, I would like to add the latest security updates. So if we search for update and then scroll down until you find update-linux and select that, this is going to update our operating system with all of the latest security updates. And there's also an update kernel as well. So once you've selected update-linux, you can scroll down until you find test components, and these are the optional tests that we would like to use to verify our AMI once it's been built. And there's a few different test components that they provide for Amazon Linux, but I just want to do a simple boot test. So I'm going to search for simple, and there it is. So this just executes a simple boot test to verify our AMI. Once you've selected that, just scroll down under the Storage section. This is where you can add additional EBS volumes for your AMI, and we'll just stick with the default. Then hit Next. Under infrastructure configuration, this is where we can define the infrastructure that Image Builder will use to run the build and test processes. And if you remember, Image Builder launches EC2 instances in your account to customize the images and run the validation tests. But if we stick with the defaults, then Image Builder is actually going launch quite large EC2 instances, which will not only take longer to configure. They also cost you more money. So let's create our own infrastructure configuration. So select Create a new infrastructure configuration, and this is going to give us control over the type of EC2 instances that it's going to launch. So first of all, we'll give it a name. Under IAM role, we're going to select the role that we created in the beginning. Under AWS infrastructure, this is where we can select the instance type that Image Builder is going to use for the build and test instances. So I'm going to search for micro and select t3.micro. Down here, we can select an SNS topic to receive notifications, but I'm just going to skip that. We can also define which VPC, subnet, and security groups we want to use, but I'm just going to go with the default and launch these in the default VPC. There's also some troubleshooting settings. So you can select whether you want to terminate your instance on failure or keep it alive so that you can troubleshoot and some instance metadata settings as well, but these are beyond the scope for the exam. So for everything else, we're going to stick with the defaults and hit Next. Under distribution settings, this is where you can define which regions we're going to create AMIs for and which regions will our AMI be distributed to. And by default, the AMI will be distributed to the current region, so us-east-1. So you should see, under the region settings, us-east-1. At this point, we can just hit Next, review all of our settings, and if you're happy with all of your options, select Create pipeline. And there we go. It should have successfully created everything. So it's created my image pipeline, my recipe, the infrastructure configuration, and my distribution settings as well. So I'll just close all of those messages down. There's my pipeline. I'm just going to select the pipeline. Select Actions and Run pipeline. And if you select View details up here, this is where you can see the status. Now this is going to take a lot longer than you might be expecting. We really just have to wait an eternity for it to finish. Last time I used Image Builder, it took about 20 minutes or so to complete the pipeline. So, just be prepared. It's not quick. But while this is running, we can go in and check our EC2 instances. So up here, I'm going to search for EC2. Open it up in a new tab. And if we select Instances, you should see here that it's already launched our build instance. So this is the EC2 instance that Image Builder is using to build our new image. And then in a few minutes, if everything's working correctly, we should see a test image being created as well. So it's going to create another EC2 instance that it's going to use to do our simple boot test on our AMI. So just be patient. And in the next few minutes, you should see the that instant appearing. But as I said, it's going to take about 20 minutes to complete. So now is a great time to step away from the screen, have a quick break, and come back to it in a few minutes.

After a few minutes, I'm just going to refresh my screen. And there is my test instance, which is running. My build instance has now been terminated because the build phase is over, and we're into the test phase. And if I select my Image Builder tab, I'm going to refresh the screen, and we can see the status is that it's in the test phase. So probably a few more minutes until that will be completed. A few more minutes later, I'm going to refresh my screen here, and the status has now changed to Distributing, and it's been about 25 minutes since I originally started the build pipeline, so it's still not completed. I'm just going to refresh again. And finally, it's got a status of Available. So if everything has completed successfully, it should show you a status of Available. And it's taken approximately 26 or 27 minutes. So this is our output image, and here's the ARN. So this is the output image. And if we select Images on the left, here's our image name and the version and the ARN of our image. But you can also view this AMI from the EC2 console. So head over to the EC2 console. On the left-hand side, scroll down. Then under Images, you'll find AMI. And there is your AMI. So that is everything that I wanted to show you for EC2 Image Builder. And for the exam, just remember that in the first step we provide a base operating system image, for example the Amazon Linux 2 AMI. Next, we define the software to install, and that could be anything like .NET. Node.js, Python, or the latest security updates like we did in the this lesson. Or you could even add the latest kernel updates or security settings, etc. The third step is that EC2 Image Builder will run tests on the new image, for instance, to check that the image boots correctly. And then the fourth step is to distribute the image to the region of your choice. And by default, it's going to appear in the region that you're operating in. And for us, that was us-east-1. So that is it for this lesson. Any questions, please let me know. Otherwise, please join me in the next lesson. Thank you.

Introducing CloudFormation

Hello, Cloud Gurus, and welcome to this lecture, which is going to introduce CloudFormation. And we'll begin with what is CloudFormation? We'll take a look at some of the benefits, the high-level process of creating resources using CloudFormation. And we'll take a look at the CloudFormation template structure and we'll cover my exam tips as well. So what is CloudFormation? Well it's a service which allows you to manage, configure, and provision your AWS infrastructure as code. Resources are defined using a CloudFormation template and CloudFormation interprets the template and makes the appropriate API calls to create the resources that you've defined. And when you create your CloudFormation template, you can write the template in either YAML or JSON. So let's take a look at some of the benefits of using CloudFormation. Firstly, when you use CloudFormation, your infrastructure is provisioned consistently with fewer mistakes. It's quick and efficient and provisioning using CloudFormation is less time and effort than configuring everything manually. You can implement version control and peer reviews for your CloudFormation templates. It is free to use, but you will be charged for the AWS resources that you create using CloudFormation. It can be used to manage updates to existing CloudFormation stacks, and it can also handle dependencies between different resources to ensure that resources are created in the correct order. And rolling back is really easy. So you can roll back to a previous state and you can also delete the entire CloudFormation stack in one go with one click of a button. So it's really easy to clear up environments that were temporary. For example, if you created an environment that was just being used to test something, then you can really quickly have CloudFormation roll back and delete the whole thing for you as well. So let's take a look at the CloudFormation process. When you start off by creating a CloudFormation template and we use YAML or JSON templates to describe the end state of the infrastructure that we are either provisioning or changing. After

creating the template, you upload it into CloudFormation using S3 and CloudFormation will then read the template and make the appropriate API calls on your behalf. And the resulting set of resources that CloudFormation builds from your template is known as a CloudFormation stack. So let's take a look at an example CloudFormation template. And this one is written in YAML. Now most of these options or parameters are completely optional and it's only the resources section that we'll see in the next slide that is mandatory. So everything else that you can see on this page is optional. And firstly, we have the template format version and this just refers to the supported template format for the CloudFormation template. Moving on, we've got the description and you can just add in a description of your choice. Next is metadata, and that is simply data about data. And once again, it's completely optional and you can just put in a custom field. And for this one, we're just providing some data about the specific instances that this CloudFormation template is going to provision. So it just gives a description of a web server instance. Moving down to parameters, these are input values that you actually input into CloudFormation when you go ahead and launch a stack using this template. And in this case, we are defining an environment type and the allowed values are either prod or test. So we need to provide the environment type when we launch a stack using this template. And then the next parameter is conditions and we can use this to test a condition and then take action based on the outcome of testing that condition. So in this case, we're evaluating the parameter environment type that was declared in the previous parameters section. So we're evaluating that. And we are saying if it equals to prod, then we should create prod resources only. So you can use this section to tell CloudFormation to make decisions based on parameters that you have input at launch time. So moving on to mappings and we can use mappings to set our own user-defined values. For example, we could set the value of an AMI based on a region. So in this case, for eu-west-1, this is the AMI we are going to use and you could define a different AMI for every different region. The next section is called transform. And this is quite important, because you can use the transform section to include snippets of code from outside the main template. And essentially what you are doing is importing that code into the template and you store the code in S3 and then specify the location in S3 for the code that you want to include. And there's also a load of code snippets that AWS provides. And I'm just going to show you a webpage where you can find these and this page is linked in the resources for this lesson. And there's a whole load of different example code snippets that you can use to declare various different resources using CloudFormation. For example, down here there's one that defines an elastic load balancer and they give you the template in both JSON and YAML code. So you can actually create reusable pieces of code, which you can store in S3 and then reference your code within the CloudFormation template. So this just allows you to reuse code and maintain consistency throughout your code base. And you can also use this transform section to specify a Lambda function as well. Moving on to the resources section and this section of the file is the most important. And it's actually the only mandatory section of the template. And this is where you define the AWS resources that you want CloudFormation to deploy. And in this case we'll be deploying an EC2 instance. We're defining the instance type as a t3.micro and we're also specifying which AMI we want to use as well. And then finally, the last section of this example template is the output section. And in this particular template it's going to output the instance ID of the EC2 instance that we are provisioning and you can see the outputs displayed in the AWS console and they can also be used as input into another CloudFormation stack as well. And you don't need to know the template off by heart. It's unlikely that they're going to ask you really detailed questions about it in the exam, but it is important to understand the general anatomy of a CloudFormation template and what all the different sections are used for. And remember the resources section, the most important section of the template, is the

only mandatory section of the CloudFormation template. And the transform section can be used to reference additional code stored in S3 allowing for code reuse. For example, you can specify template snippets or reusable pieces of CloudFormation code and you can also use it to specify a Lambda function as well. So let's take a look at my exam tips for CloudFormation. And just remember, that CloudFormation allows you to manage, configure, and provision AWS Infrastructure as Code. And we can use either YAML or JSON for the CloudFormation template file. And you will need to be aware of the main sections of the CloudFormation template. So we have parameters which allows you to input custom values. And in the example, we used an environment type. So CloudFormation will expect you to state which environment the resources are going to be for, for example, production or test. Conditions are used to make decisions based on a set of parameters. For example, it will allow you to provision resources based on environment if you've used an environment type parameter. The resources section is the only mandatory section and defines all the AWS resources that you are going to create. And that could be EC2 instances, an S3 bucket, elastic load balancer, RDS database, pretty much anything that you can think of within AWS. You can define it within the resources section of your CloudFormation template. The mapping section allows you to create custom mappings for example, mapping a specific AMI to a specific region. And finally, the transform section allows you to reference code located in S3, for example, reusable snippets of CloudFormation code stored in S3. However, the best way to learn CloudFormation is definitely to start using it and get your hands dirty with it. And that is exactly what we'll be doing in the next lecture. So if you have time, please join me in the next lecture. Thank you.

Demo: Provisioning AWS Resources Using CloudFormation

Hello Cloud Gurus and welcome to this lesson where we'll learn how to provision AWS resources using CloudFormation. And I'll begin by creating and downloading an SSH key pair. And we're going to use this later on to log into an EC2 instance. Next, we'll create a CloudFormation stack, and we'll use a provided template to provision an EC2 instance with SSH enabled. And then finally, we'll review our CloudFormation stack. So we'll identify the new instance in the AWS console, and we'll test that we can successfully log into the instance using SSH and providing the name of the key pair that we created in the beginning. So if you're ready to get your hands dirty with CloudFormation, then please join me in the AWS console. Now the first thing I'm going to do is search for EC2. And we are going to create a new key pair. So select Key pairs, make sure that you're working in the Northern Virginia region, and Create key pair. We'll give it a name and make sure you remember the name that you've given it because we're going to use that later. The key pair type is RSA. Down here, you can select the private key file format that you want to use, and I'm going to go for the .pem format because I'll be using OpenSSH to connect to my EC2 instance. But if you normally use PuTTY, then you can create your key in a PPK format as well. So then, come down to the bottom and Create key pair. So that is my key pair created. It's automatically downloaded to my Downloads directory on my local machine. So now, let's head to CloudFormation. Select Create stack, With new resources, and we're going to use a template that's already been prepared. So select Template is ready. And here is the template that I've created for you. And you will find a link to the Git repository in the resources for this lesson. And if you select Raw and then right-click and select Save As, it's going to download this file to your local machine into the Downloads directory. So let's just take a look at this template, and I'll show you exactly what we're creating here. And it's actually a really simple template. All it's going to do is create an

EC2 instance and enable SSH. So first of all, we've got our template format version. Next, we've got our description. Moving down to Parameters and we're defining a parameter called KeyName, and this is going to be the name of the SSH key pair that you want to use when you're logging into your EC2 instance. So it's going to be the SSH key pair that we created in the previous step, and we'll provide this as a parameter to CloudFormation. And down here, we've got this constraint description telling us to provide the name of an existing SSH key pair. Now the next section is Resources, and this is the most important section of the CloudFormation template. And it's actually the only section that is mandatory in a CloudFormation template. And this section defines the resources that we are asking CloudFormation to create. So the first one is my EC2 instance. So this is our EC2 instance. Under Properties, we've got an instance type, which is going to be t3.micro. Here's the image ID, and this is the AMI ID of the AWS Linux 2 AMI available in the Northern Virginia us-east-1 region. And if you remember, AMIs are region-specific. So let's say I try to access this AMI from a different region. If I try to launch this CloudFormation stack in a region like eu-west-1, for example, well it's not going to work because this image ID, this AMI, is not going to be valid for the eu-west-1 region. It's only available in us-east-1. So when using this template, you need to make sure that you are launching instances in the correct region. The next line is KeyName. So this is our KeyName parameter, and it's telling CloudFormation to associate this SSH key pair with our new EC2 instance. Scrolling down, we're adding a tag. So I've defined a tag of Name, and the value is going to be My CF Instance. So that is our first resource. And then the second resource that we're creating is our instance security group. So this is a brand new security group, and it's going to enable SSH on port 22 from an IP address range of 0.0.0.0/0, in other words enabling SSH to this instance from anywhere. And then the last section of the file is Outputs. And in this case, I'm asking CloudFormation to output the instance ID of the new EC2 instance that will be created. So there you go. That's a really, really simple CloudFormation template. And if you haven't already, remember to right-click and select Save As to download it to your local machine. And once you've done that, we should be good to go. Heading back to the AWS console, let's scroll down, and we're going to upload a template file. Select Choose file. Select the CloudFormationTemplate.yml and Open. And before we go any further, just make sure that you are operating in the Northern Virginia region. And once you've uploaded your template, you will notice that an S3 URL appears down here. And this is because CloudFormation has created an S3 bucket to store the template inside. So if you're happy with that, click Next. We'll give our stack a name. Down here, we need to provide the parameter that was defined in our template. So we need to provide the key name of the SSH key pair that we created. If you select the drop-down, you should be able to find your key pair, and there it is. So once you've selected that, hit Next. You can optionally create a tag, and you can optionally choose an IAM role for CloudFormation to use for all the operations it's going to perform, but we won't need to do that. Under Stack failure options, this is where you can specify rollback behavior if something goes wrong. By default, it's set to roll back everything on failure. And that means that if something goes wrong with the provisioning, then it will automatically roll back the whole thing. For instance, let's say there was a problem with your YAML template and CloudFormation cannot successfully complete, then it will automatically roll everything back, even resources that were successfully provisioned. However, the choice is yours. For example, if you wanted to debug a CloudFormation stack that was consistently failing, you could select Preserve successfully provisioned resources and try to understand what is actually going wrong. Under Advanced options, there's a few options that I'd like to draw your attention to. First of all, we've got the stack policy, and a stack policy is a document that defines the update actions that are allowed to be performed on designated specific CloudFormation resources. And this is where you can add a policy to protect

specific critical resources from deletions or unintentional updates, and it's great for protecting mission-critical resources. And we'll be taking a closer look at stack policies later on in the course. Down here, we have rollback configuration. This is where you can specify alarms for CloudFormation to monitor when creating and updating the stack. So if an operation breaches an alarm threshold that you've defined, then you can have CloudFormation roll it all back for you. Under Notification options, you can specify an SNS topic if you'd like to receive notifications. And then finally, there's stack creation options. And under Timeout, you can define the number of minutes before a stack creation times out, and then you've also got termination protection, which is used to prevent the stack from being accidentally deleted, and this is just another option that is great for mission-critical resources. So now we can just go ahead and hit Next. We can review our options, come down to the bottom, and submit. Now it can take a few minutes to go ahead and create everything, and you can see the status of all the tasks it's doing in the Events tab on the right-hand side. And after a few minutes, everything should have completed. But if the status has not updated, you can refresh using this button here. And if everything's been successful, it should say CREATE_COMPLETE. And I'm also going to refresh on this slide as well. So from this Events tab, we can see all the events associated with the CloudFormation tasks. So we can see my instance being created. We can see the security group being created as well. And then if we roll back up and select Resources, we can view the resources that were created on this tab. So here's our security group, and here's our EC2 instance. And if we select the Outputs tab, and if you remember our template had an output of the instance ID and there it is. If we select Parameters, here's the parameter that we entered. So we used a parameter to define the SSH key name that we wanted to use. And then our template is here. So now, let's head over to our EC2 dashboard. Select EC2. Select Instances. And there is the instance that we've just created. Select your instance ID. Scroll down to Details. Here's the SSH key pair that we defined earlier. If you select Security, here's the security group that we created. So it has successfully created everything for us, and that should be everything that we need to go ahead and log into our new instance. So first of all, to do that, I'm going to scroll back up to the top and copy my public IP address. Then, using a terminal window on my local machine, I'm going to try and SSH onto that instance. First of all, I need to change the permissions on my new key pair that I created. So in my Downloads directory, I'm going to run chmod 400 followed by the name of my key pair and hit Enter. Then, run ssh ec2-user@ and then paste in the public IP address -i and the name of my key pair and hit Enter. Answer yes to this question, and there we go. We've SSHed onto our EC2 instance that we created using CloudFormation. So that is CloudFormation. And as you can see, it's a really simple tool to use, and it's a really powerful tool as well for deploying AWS resources. And if we head back to our console, come back to CloudFormation, we can select our stack, and we can actually have CloudFormation delete everything that it created for us using this template, but only because we had termination protection disabled. So I'm going to hit Delete and Delete stack, and that will delete my entire stack. So everything that we created using our template is going to be deleted. So for the exam, just remember that with CloudFormation, we can use either a YAML or JSON template, and the CloudFormation template is used to describe the end state of the infrastructure that you're either provisioning or changing. After creating the template, you can upload it to CloudFormation, and CloudFormation will save it to S3. But you can also save it to S3 yourself and then just provide the location of the file to CloudFormation. When you launch your stack, CloudFormation reads the template and then makes the API calls to AWS on your behalf. And the resulting set of resources is called a CloudFormation stack. So that is it for this lesson. Any questions, please let me know. Otherwise, I'll see you in the next lesson. Thank you.

Troubleshooting CloudFormation

Hello, Cloud Gurus, and welcome to this lecture, which is going to cover troubleshooting CloudFormation. And first of all, we'll take a look at where to find error messages for CloudFormation. What could possibly go wrong? We'll take a look at the UPDATE_ROLLBACK_FAILED error message and we'll consider a failed rollback scenario, followed by my exam tips. So where do you find error messages for CloudFormation? Well the first place to look is the CloudFormation console. And you can view the status of your stack using the CloudFormation console. Each CloudFormation event will include a status and you can check each status for error messages that will help you troubleshoot and work out what went wrong. So let's take a look at some example messages from the CloudFormation console. And you will see in this example that we had an issue creating an EC2 instance and then under status reason, it gives you a reason why CloudFormation failed. And in this example, I used an AMI which did not exist. So either we've specified the AMI name incorrectly or we're using an AMI which doesn't exist in the region that I am trying to launch the EC2 instance. So it's giving me this error saying that the AMI does not exist. And then because CloudFormation has failed to create this resource, including creating the instance security group, which you can see completed successfully but CloudFormation is going to roll everything back due to this failure. So now let's take a look at what could possibly go wrong with your CloudFormation stacks when attempting to launch a CloudFormation stack. And there are a few common scenarios that may cause your CloudFormation operations to fail. First of all, we have identity and access management and IAM can be the cause of many headaches and errors within AWS. And I always think that whenever something goes wrong in AWS, is it to do with insufficient permissions and with CloudFormation, in order to create EC2 instances or any AWS resources, you will need permission to create those resources. So if you're creating EC2 instances, you will need EC2 permission. The next thing that could cause your CloudFormation operation to fail is that you have hit a limit. For example, there is a default limit of 20 EC2 instances per region. And if your CloudFormation template is taking you over that limit, then you will see an error message relating to resource limit exceeded. And then finally the last one that you need to know about is failed rollback. And this is when CloudFormation is attempting to roll back a change. And for some reason it cannot complete the rollback operation. So let's take a look at that in a little bit more detail. When you get a failed rollback, you will see this UPDATE_ROLLBACK_FAILED error message. So what is rollback? Well by default, CloudFormation will roll back to the previous state if a CloudFormation operation fails. For example, if the create stack or update stack operation do not work as planned. And we saw that in my example, where it failed to create the EC2 instance and then it initiates a rollback, but why might it fail in the first place? Well, it can definitely fail if you are trying to roll back something that has been changed outside of CloudFormation. For example, if you made a manual change to your stack or your infrastructure without using CloudFormation. So in this scenario, you would've made a change that CloudFormation doesn't know about. And when you try to roll back, CloudFormation will discover that the stack is in a different state to what it expects and it will throw an error. So what can you do about it? Well, in most cases you will need to fix the error that caused the update rollback to fail before you can continue to roll back your stack. So let's take a look at an example failure scenario. And one common scenario that you might encounter is when you are attempting to roll back changes on a resource which no longer exists. Let's say for example, you have a stack which begins to roll back to an old database instance, but that database was deleted outside of CloudFormation. So what should you do about it? Well, you will need to manually synchronize the resources so that they

match the original stack's template. And then once you've done that, you will be able to continue rolling back the update. So basically you will need to manually recreate the resource using the same name and properties that it had in the original stack and just get it to the same state that CloudFormation is expecting before you try to run the rollback operation a second time. So onto my exam tips, and just remember that you can use the CloudFormation console to view the status of your stack and error messages. And one of the 3 common scenarios that will generate a CloudFormation error are firstly, insufficient permissions. So make sure you have added permissions for the resources that you are trying to create, delete, or modify. Resource limit exceeded is another common error. And if you see this error, you will need to request a limit increase or delete any unnecessary resources and retry. And then finally we have the UPDATE_ROLLBACK_FAILED error. And if you see this error message, you will have to go in and fix the error causing the failure and retry. So basically you'll need to manually recreate any resources with the same name and properties that were used in the original stack and just get everything to how CloudFormation is expecting before you try and run the rollback operation a second time. So that's it for this lecture. If you have any questions, please let me know. Otherwise I'll see you in the next lecture. Thank you.

Demo: CloudFormation Errors

Hello Cloud Gurus and welcome to this lesson where we'll be taking a look at CloudFormation errors. And we'll begin by creating and downloading an SSH key pair, and we're going to use this later when we create our stack. Next, we'll launch a CloudFormation stack, and we'll use the provided template to launch our stack. And then finally, we're going to review the CloudFormation events. And after the stack creation has failed, we'll use the status messages to try to determine what went wrong. So if you are ready to generate some CloudFormation errors, then I will see you in the console. So from the AWS console, first of all, let's head to EC2. Make sure that you're in the Northern Virginia Region and select Key pairs. Let's create a new key pair. I'll call it mycfkp and Create key pair. So now, let's head to CloudFormation. Create stack, With new resources. And we'll use a template that you will find in the Resources section of the course, and here is the template. And if you've completed our previous CloudFormation demo, then this will look very familiar to you. All we're doing is creating an EC2 instance. We're associating an SSH key pair, and we're creating a security group as well, allowing SSH on port 22. So you'll want to download this template, first of all, by selecting the raw version, then right-click and Save As. And this will download to your Downloads directory. Back in CloudFormation, make sure you've selected Template is ready, Upload a template file, Choose file. Select the CloudFormationTemplate_With_Error.yml. Open, scroll down, and hit Next. I'll name my stack my-stack-with-error. We need to provide the name of our SSH key pair. So select your key pair, hit Next, scroll down to the bottom, hit Next. You can review all your options here and then submit. As you know, it does just take a few moments to complete, and you can refresh using this button and this button. And there we go. Our stack has actually failed, and it's rolling back at the moment. So let's take a look at what's gone wrong, starting off from the beginning. So if you scroll down to the bottom of the events, we can see that we successfully launched our stack, we created the InstanceSecurityGroup, and then we started to create the EC2 instance. And then it's here when we're creating the EC2 instance that we've seen the failure. And it's giving me this message saying InvalidAMIIDNotFound. And it's saying that the image ID does not exist. So there's something wrong with the AMI ID that I've specified in the template. So what can we do to work out what

went wrong? Well, I'm going to head over to Google, and I'm going search for this AMIIDNotFound message. And this one looks pretty promising. It's on docs.aws.amazon.com. I'll select that, and this is giving us all of the error codes for the EC2 AMI. I'm going to search for InvalidAMIIDNotFound, and there we go. The specified AMI does not exist. Check the AMI ID and ensure that you specify the AWS region in which the AMI is located if it's not in the default region. And this error may also occur if you've specified an incorrect kernel ID when launching the instance. And that actually describes the error really, really well. So let's head back to our CloudFormation template. Here's my AMI ID. I'm going to copy the AMI ID that I'm using. Come back to the AWS console. Search for EC2. I'll open that in a new tab. And then, on the left-hand menu, scroll down until you get to Images, and then select AMI Catalog. I'm going to search for the AMI that we're trying to use. So paste it in the search box and hit Enter. And it's telling me no results were found for that AMI. And we want to be looking in the Quickstart AMIs section. So it's not finding that AMI ID. But what if I change region? And I'm going to change from the Northern Virginia region, us-east-1, to us-east-2. I'll search in here for my AMI ID, and there we go. That is our problem. I've used an AMI ID that's from the wrong region. It's a valid AMI, but only for us-east-2, and I cannot use it in us-east-1. So basically, we have the wrong AMI. So for my CloudFormation error exam tips, just remember that you can check the CloudFormation console for error messages when you're troubleshooting, and it gave us a really helpful status reason and a really useful error code that we could use to understand what went wrong in our CloudFormation stack. So that's it for this lesson. If you have any questions, please let me know. Otherwise, please join me in the next lesson. Thank you.

Introducing CloudFormation StackSets

Hello Cloud Gurus and welcome to this lecture, which is going to introduce CloudFormation StackSets. And we'll start off by looking at what are CloudFormation StackSets? We'll take a look at the cross-account roles, which underpin the use of StackSets. We'll review some Identity and Access Management Policy examples. We'll take a look at Resource Access Manager and CloudFormation StackSet exam tips. So what are CloudFormation StackSets? Well, they allow you to create, delete, and update your CloudFormation stacks across multiple AWS accounts and Regions using a single operation. So we can use a StackSet to launch a CloudFormation stack in Account A and in Account B and in different AWS Regions as well, using a single CloudFormation operation. And in order to launch a CloudFormation stack in another AWS account, we will need to use cross-account roles. So you will need to configure the appropriate Identity and Access Management permissions, using cross-account roles. So in your administrator account, and this is the account where you are launching the StackSet from, you will need to create an Identity and Access Management role, and in our example, our role is named AWSCloudFormationStackSetAdministrationRole. And then in your target accounts, and these are the accounts where you want to create the CloudFormation resources, you will need to create a service role, and in this example, our service role is called AWSCloudFormationStackSetExecutionRole. And the StackSetAdministrationRole from the administrator account needs to be allowed to assume the StackSetExecutionRole in the target accounts in order to provision resources in those target accounts. So let's take a look at the Identity and Access Management policies that you would need to attach to these roles. So in the administrator account, you will need to create a policy like this, which allows the administrator account to call sts:AssumeRole and assume the StackSetExecutionRole. And in the administrator account, you will also need to create a trust policy for the role. And this will allow CloudFormation

in the administrator account to assume the StackSetExecutionRole. And then in the target account, the StackSetExecution service role needs a policy like this, which gives permissions for specific CloudFormation actions on specific AWS resource types, and it's in this policy that you define the actions that you are permitting in the target account, and you scope it for only the type of resources that you want to create in the target account using StackSets. And then you will also need a trust policy for this role, allowing the administrator account to call STS:AssumeRole and assume the StackSetExecutionRole in the target account, and don't worry, you don't need to memorize all of these policies for the exam. You just need to understand how it all hangs together at a high level. So once you've created all these resources across different accounts, how are you going to access them? Well, that is where Resource Access Manager comes in. And Resource Access Manager is a service which allows you to share resources across different AWS accounts. For example, EC2 instances, S3 buckets, and EC2 Image Builder images. And you can share AWS resources with any other AWS account of your choice. And the way it works is that you create a resource share and then send an invitation to the account that you would like to share your resources with. And after accepting the invitation, the account will then have access to the shared resources. And once again, you only need to understand this at a very high level. So onto my exam tips for CloudFormation StackSets. And just remember that StackSets allow you to create, delete, and update CloudFormation stacks across multiple AWS accounts and Regions using a single operation. And in order to use StackSets, you will need to configure some cross-account roles, and in our example, in our administrator account, we had the AWSCloudFormationStackSetAdministrationRole, which was allowed to assume the AWSCloudFormationStackSetExecutionRole, and it was this role which had permission to provision resources in the target accounts. And then finally, once you've created all these resources in different accounts, you can configure access to them using Resource Access Manager. And this is a service, which allows you to share resources with other accounts, for example, EC2 instances, S3 buckets, and EC2 Image Builder images. So that's it for this lecture. If you have any questions, please let me know. Otherwise, I will see you in the next lecture. Thank you.

CloudFormation Best Practices

Hello, Cloud Gurus and welcome to this lecture where we are going to take a look at CloudFormation best practices. We'll start off with best practices and why we need them. We'll then cover stack policies. And we'll be taking a look at an example stack policy, followed by my exam tips. Now, as you know CloudFormation is an extremely powerful tool. And of course, "with great power comes great responsibility. " And you may be thinking that is a quote from Spiderman, but it was originally coined by Voltaire who was an 18th century French writer and philosopher who was instrumental in instigating the French Revolution. But what he meant by that phrase it actually had nothing to do with CloudFormation. What he actually meant was that no authority should be immune to challenge by a reason. And that applies to CloudFormation as well. And there are some simple best practices which are worth adhering to when you make changes in your environment using CloudFormation. And it's definitely worth being aware of these best practices, because you will be tested on them in the exam. So let's start off with identity and access management, and it is best practice to control access to CloudFormation using identity and access management. You should also be aware of the service limits in your account. And if you hit the service limit, then CloudFormation will fail to create your stack. For example, there is a limit of 20 EC2 instances per region, and let's say your CloudFormation template is trying to create 25 instances in a single region. Then you are going to hit the limit and CloudFormation will throw an error and it will fail to

create your stack. You also want to avoid manual updates when you are using CloudFormation because manual updates will create a mismatch between your stack template and the current state of the stack. And that's going to cause errors of course, when you try to update or delete the stack and CloudFormation will fail because there will be a mismatch between what it's expecting based on the template and the actual state of the stack. And you should use CloudTrail to log all CloudFormation API calls, so that you have an audit trail of all the changes and whoever made them. And then finally you should specify a stack policy whenever you create a stack that has critical resources. And that is a best practice from AWS. And a stack policy is a JSON document, which describes what update actions can be performed on designated resources. So if you have mission critical resources and you want to protect them against accidental change or deletion by people using CloudFormation, then you can do that using a stack policy. And that just helps to protect critical stack resources from unintentional updates and mistakes caused by human error. So we are going to take a look at an example stack policy right now. And it's actually really simple. And if you take a look at the first statement in the policy, this allows all update actions to be made by any user with access to CloudFormation on any resource in the stack. But then the second statement denies any update actions by any user. And then it specifies the logical resource ID, which is the ID of our production database. So this policy basically prevents anyone from making updates to the production database using CloudFormation. And when you want to set up the stack policy, it's really simple. When you're creating the stack in the console, you can either write it in directly or you can upload it as a text file. So moving on to our exam tips. First of all, control access to CloudFormation using identity and access management. Be aware of your service limits, because if you hit a limit then CloudFormation will fail to create your stack. And remember that for EC2, of course, there is a limit to the number of EC2 instances that you can create in a single region. And at the moment the limit is 20. So if you create a CloudFormation template, which is going to take you over that limit of 20 instances per region, then CloudFormation will fail. It's best practice to avoid manual updates when you are using CloudFormation. So don't go in and manually make changes to your databases or your EC2 instances, because that can cause errors later on when you try to update or delete those resources using CloudFormation. Of course, use CloudTrail to track all the changes made in CloudFormation, along with who made them and when. And finally, if you have mission critical resources, then use a stack policy to protect your critical resources from either unintentional updates or mistakes caused by human error. So that's it for this lecture. If you have any questions, please let me know. Otherwise I will see you in the next lecture. Thank you.

Exploring Blue/Green Deployments

Hello Cloud Gurus, and welcome to this lecture where we're going to take a look at blue/green deployments. And we'll begin with what is a blue/green deployment. And we'll take a look at how it can be used to reduce deployment risk, when to use a blue/green deployment strategy, and my exam tips as well. So, what is a blue/green deployment? Well with a blue/green deployment, you have two identical environments, one blue and one green. And blue is the current version of your application, whereas green is the new version of your application. And all live customer traffic will go to the blue environment, whereas our test traffic will go to the new version to test our new environment and our new application version before we send our customer traffic to this new version of our application. And after testing is successfully completed on the green environment, then live traffic can be directed to the green environment and the blue environment eventually becomes deprecated. And later on, if something goes wrong with the new version that wasn't picked up in testing, then

you can easily roll back to the original version of your application. And this is a great way to reduce deployment risk because we deployed the new version of our application to a completely new environment. And if something goes wrong, then we can redirect customer traffic to the previous version, as long as we haven't terminated that environment. So, when would you use a blue/green deployment strategy? Well, it's a great one to use if you are introducing a new version of your application. So, you're deploying a new version of an application that you are already running in a production environment. It's a great one to use if you need to roll back your changes really quickly, and if you want a really simple and fast rollback process. And it's a great way to reduce deployment risk and avoid outages. So, on to my exam tips. And just remember, a blue/green deployment strategy is a low risk strategy. And when we say blue/green, blue is the current version of the application and green is the new version. After testing is successfully complete, live customer traffic can be directed to the new version. And eventually, when you're happy with the new version, the current version can be deprecated. And finally, rollback is fast and easy. And if something goes wrong after the new version is being used in production, simply redirect all customer traffic back to the original environment. So, that's it for this lecture. If you have any questions, please let me know. Otherwise, I will see you in the next lecture. Thank you.

Understanding Rolling Deployments

Hello, Cloud Gurus, and welcome to this lecture, which is going to cover rolling deployments. And we'll begin with, what is a rolling deployment? We'll take a look at a rolling deployment CloudFormation stack example, the disadvantages of using a rolling deployment, when to use a rolling deployment strategy, and my exam tips as well. So, what is a rolling deployment? Well, a rolling deployment allows you to make changes to your application or your environment in batches, and to deploy your changes to only a portion of your environment, instead of using an all or nothing approach. So, let's take a look at a CloudFormation stack example where we're deploying changes in an existing environment. So, it's an existing CloudFormation stack. So, here is our stack, and it consists of an auto-scaling group of 4 web servers, and our web servers are running a very old version of Apache, and we would like to deploy the latest version, and we can use CloudFormation to deploy our change in batches using a rolling deployment strategy. So, here is our stack, and with a rolling deployment, we can set a minimum number of servers that we want to keep in service, for example, 2. So, we'll keep these 2 servers in service as normal. And then, we can set a maximum batch size, for example, 2. And these 2 servers will be our Batch 1. When we begin the rolling deployment, our Batch 1 servers will be updated to the latest version. And once that has successfully completed, our Batch 2 servers will be updated. Now, all that sounds pretty good, right? However, there is a trade-off to using a rolling deployment strategy, and there are some disadvantages that you will need to be aware of. So, first of all, during the deployment, once you've made your changes on the first batch of servers, you will have old and new versions of your application running side by side. And depending on your application, that may or may not cause issues for your application or for your customers. And secondly, if a deployment fails, then you will need to redeploy the previous version to get back to your original configuration. It won't be instant like it would be if you were using the blue-green deployment method. So, when would you use a rolling deployment strategy? Well, it's a great one to use if you are looking for a cost-effective strategy, because you are only maintaining a single environment, unlike a blue-green deployment. And if you don't mind operating in a mixed environment with some of your servers running a new version of your application, and others running an old version, and others running an older version,

then it is a great cost-effective strategy. So, onto my rolling deployment exam tips. And just remember: a rolling deployment strategy allows you to deploy new application versions and other changes in batches. It's a cost-effective approach because it doesn't involve provisioning a second environment, and you can set the batch size and the minimum number of servers to keep in service. However, there is a trade-off, and the trade off is added complexity. So, for a while, you will be running a mixed environment, with some of your servers running the new version of your application, and some still running the original version. And also, rolling back is going to involve a redeployment, unlike with the blue-green approach, which allows you to simply redirect customer traffic back to the original environment. So, that's it for rolling deployments. If you have any questions, please let me know. Otherwise, I will see you in the next lecture. Thank you.

When to Use Canary Deployments

Hello, Cloud Gurus, and welcome to this lecture, which is going to cover canary deployments. And we'll begin with what is a canary deployment? The benefits of a canary deployment and when to use one, and my exam tips as well. So what is a canary deployment? Well, imagine that you have 10 web servers behind an elastic load balancer. With a canary deployment you can deploy the new version of your application to a small number of web servers. And the remaining web servers will stay on the original version. And then you can direct only a small proportion of your customer traffic to the new version. And let's say 10% of your customer traffic is going to the new version, while 90% of your customer traffic is going to the original version. So if any problems arise with the new deployment then it's only going to impact 10% of your customers. And this allows you to test the new version in production before fully committing. And if there is an issue then you can quickly redirect all traffic back to the original version while you fix the problem. However, if there are no issues with the new version then you can deploy the new version to the remaining servers and direct all traffic to your new version. And just like the canary in the coal mine, this is a great approach to use when you want to deploy changes to a mission critical production environment in a controlled way, and get an early indication that something is wrong. And if you haven't heard about the canary in the coal mine, it's an old practice where coal miners would bring a poor canary down in the mine tunnels with them while they were working. And if any dangerous gases like carbon monoxide built up within the mine, then the canary would react. So maybe it would pass out or start making noises, and this would alert the miners so that they could evacuate immediately. And if you are an animal lover like me, then don't be alarmed because they would also take a tank of oxygen to revive the little canary, if it passed out. So with a canary deployment, you can test out your new deployment in production by directing a small percentage of traffic towards your application version. And then if something is wrong, your customers will react, and hopefully they won't pass out like the canary, but they will start making noises and letting you know that something is wrong. So a canary deployment provides an early indication that something is wrong in your application. And this is a great one to use if you want to reduce the risk of deploying new application versions or changes when working with a production environment. And finally rolling back is very easy. You simply redirect all your customers back to the original version. So onto my exam tips and just remember the canary in the coal mine. So a canary deployment is an early warning system that can indicate that something is wrong in your application. You deploy the new version to a small number of servers, direct a small proportion of customer traffic to the new version, for example, 10%. And this allows you to test your application with a small proportion of real customers before you roll out to everybody. And this enables canary testing, which allows you to test your application with a

small proportion of real customers before you roll out to everybody. So that's it for this lecture. And if you have any questions, please let me know. Otherwise, I'll see you in the next lecture. Thank you.

Automating Tasks Using AWS Systems Manager

Hello, Cloud Gurus, and welcome to this lecture, which is going to cover automating tasks using AWS Systems Manager, and it's also known as SSM. So we'll begin with looking at what is Systems Manager? And then we'll take a look at a couple of amazing features of Systems Manager, and the first one is Run Command, and the second one is Patch Manager. And then finally, we will cover my exam tips. So what is Systems Manager? Well, it's a management tool, which gives you visibility and control over your AWS infrastructure, and it provides an inventory of your EC2 instances, allowing you to organize and group resources together in a way that makes sense to you. So either by application or environment, or even by business unit or project, and even including on-premises systems as well. And it allows you to automate common operational tasks on multiple systems simultaneously, for example, patching, installing applications, and running scripts, etc. So Systems Manager makes it really easy to manage large fleets of EC2 instances, and it allows you to perform common operational tasks on groups of instances simultaneously without having to log into each one. So let's take a look at what you can do with Systems Manager Run Command. I want you to imagine that you're a SysOps Administrator, and you are supporting hundreds of servers, and your manager asks you to run a command on each one to check the network configuration of each server. Well, this is something that you can do with Systems Manager Run Command, and you can use Run Command to run a command on all of these systems simultaneously without having to log into a single one. So Run Command is a great tool, and it allows you to run operational tasks across multiple EC2 instances. For instance, you can run predefined commands and scripts on one or more EC2 instances. You can stop, restart, terminate, or resize instances, attach or detach EBS volumes, or install individual patches and packages. We then have another great feature of Systems Manager, and it's called Patch Manager. So imagine you are a SysOps Administrator supporting hundreds of servers, and your manager asks you to update the operating system patches on each of these servers. Well, imagine patching all of these systems manually. Not only is it going to be time-consuming, it is also boring and labor intensive, and there are much more exciting things to do as a SysOps Administrator. So that is where Patch Manager comes in. And you can use Patch Manager to patch all of these systems without having to login to a single one. So onto my exam tips. And just remember that Systems Manager gives you visibility and control over your AWS infrastructure at scale. It provides inventory and automation. So it allows you to organize your inventory and logically group resources together in a way that makes sense to you, and this enables you to automate boring tasks like patching and running scripts and running commands on multiple instances at once. And one of the really great features of Systems Manager is Run Command. And this allows you to perform operational tasks on groups of instances simultaneously without logging into each one. And we then have Patch Manager, which allows you to patch multiple EC2 instances simultaneously. So that's it for this lecture. And if you're ready to get your hands dirty with Systems Manager, then I will see you in the next lecture. Thank you.

Demo: Implementing Automated Patching Using AWS Systems Manager

Hello Cloud Gurus and welcome to this lesson where we'll take a look at automating patching using Systems Manager. And we'll start off by creating an Identity and Access Management role, and we'll

attach a managed policy to our role. And the managed policy we're going to attach is called AmazonSSMManageInstanceCore. Next, we'll launch an EC2 instance, and we're going to attach the role that we just created to our instance as an instance profile. We'll then open the Systems Manager console, and we should be able to locate our instance in the Fleet Manager section. And then, we're going to use Patch Manager to scan for any missing patches and install any missing patches on our EC2 instance. Now we don't currently support the use of Patch Manager in the cloud playground or the AWS cloud sandbox. So please use your own AWS free-tier account to complete the steps in this demo. So if you'd like to join me in the AWS console, let's get to it. So from the console, and remember you need to do this in your own AWS account, search for IAM. Select Roles and Create role. Make sure the AWS service is selected. Common use case is EC2. Select Next, and we're going to search for SSM. And this is the one that we're looking for, AmazonSSMManagedInstanceCore. So select that, scroll down to the bottom, and hit Next. Name it my-ssm-role. Scroll down to the bottom and Create role. Next, let's create our EC2 instance. So search for EC2, Launch instance. Call it SSM-Demo. We'll use Amazon Linux. Instance type is t3.micro. We'll proceed without a key pair. Scroll down to Advanced details. And under IAM instance profile, select my-ssm role. The rest of the defaults are fine, so you can go ahead and launch instance. Select your instance ID. And in a few moments, it should have initialized. After a few moments, we can refresh. And there, our instance is ready. So now, let's head to Systems Manager. On the left-hand menu, scroll down and select Fleet Manager. And if our system has correctly registered with Fleet Manager, then it should appear down here. And there it is, SSM Demo. So this instance has successfully registered with Systems Manager, and I do have another instance here. It's actually in a stopped state, and it is not registered with Systems Manager. So now, on the left-hand menu, I'm just going to select this icon. Then scroll down until we get to Patch Manager and select that. First of all, let's view the predefined patch baselines. So select that, and this is where you will find a whole load of predefined baselines. And these are basically a baseline of all the latest security patches that have a severity level of critical or important, as well as any patches relating to bug fixes. And they are all organized by operating system. So let's filter based on Amazon Linux 2. If you select the search box, select Operating system, select Operating system =, and we'll select Amazon Linux 2 from down here And there we go. There is our baseline for Amazon Linux 2. And you'll notice on the right-hand side, this is actually set as a default baseline. Now at the top of the screen, you'll also see that there is a dashboard, so select that. And this dashboard will basically give you a patch compliance summary, and you'll be able to see which instances are being managed by Systems Manager. You've got to a compliance summary, which gives you the status for managed nodes that have previously reported patching data. You've got noncompliance counts and compliance reports, and it's showing here that we've got one host, which has never reported. And then finally down here, we've got the patch operations history. And I've been playing around with Patch Manager this afternoon, so I've got quite a bit in my patch operations history. And then down here, we have any recurring patching tasks. Now if you scroll back up to the bottom and select Compliance reporting, this will give us a report based on instance ID, and it will tell us which of our systems are in compliance for reporting, and this one has never reported. If you select Patches, this is where you can select specific patches for your operating system. So if we select Amazon Linux 2, we can go in and search all the different kinds of patches that we're looking for. For instance, if we were looking for Python-related patches, you can select Name and search for Python, and it will select all of the Python-related patches. So now, let's come back to our patch baseline. We can select our operating system = Amazon Linux 2. Here's our patch baseline. Select your patch baseline and then select Patch now. And this is where we can configure

our patching job. And we can either scan for any missing patches, or we can scan and then install the missing patches. So I'm going to select Scan and install. We can specify if we want to reboot if needed or do not boot or even schedule a reboot for another time. And we can decide if we want to patch all instances or only the target instances that I specify. By default, it creates a patching operations log that it's going to store in S3. But I'm going to select do not store logs. Then under Advanced operations, this is where you can configure lifecycle hooks. And these refer to SSM documents or scripts that you can run at certain points during the patching operation, and this allows you to orchestrate for complex patching scenarios, for instance if you needed to perform specific actions during the patching process. So now, all we need to do is select Patch now. And if we scroll up to the top, we can follow the status as the patches are executed, and it will take a few minutes to complete. And there we go. We've got a status of Success. And if we scroll down, we should see the that we have a 100% succeeded. So now, let's head back to Patch Manager and come to our dashboard, and we should see that in our compliance summary, we are 100% compliant. We've got zero nodes with missing patches or failed patches. And if we scroll down to the patch operations history, we can see the last time that we ran the patch baseline. So that is Patch Manager. And if you are working in your own AWS account, then do remember to head back to EC2, locate your instance, select it, come to Instance state, and terminate your instance to avoid any unnecessary charges. Onto my exam tips. And just remember that you can use AWS Systems Manager and the Patch Manager functionality to patch multiple EC2 instances, and you can patch those instances simultaneously. So just imagine you're a sysops administrator and you're looking after hundreds of servers and you need to apply security patches to all of these instances. Well, you can do that using Patch Manager in one single operation. And you can also use Patch Manager to review a dashboard showing which of your systems are in compliance and which are out of compliance. So that's it for this lesson. And if you are working in your own AWS account, remember to delete your EC2 instance. If you have any questions, please let me know. Otherwise, I'll see you in the next lesson. Thank you.

Demo: Using AWS Systems Manager EC2 Run Command

Hello Cloud Gurus and welcome to this lesson where we'll be getting our hands dirty using AWS Systems Manager EC2 run command. And we'll begin by creating an Identity and Access Management role, and we'll attach the AWS managed policy named AmazonSSMManagedInstanceCore. Next, we'll launch two EC2 instances, and we're going to attach the role that we just created as the instance profile. And you don't need to remember it all for the exam, but this instance profile is going to allow our EC2 instances to communicate with Systems Manager. Next, we'll open up the Systems Manager console, and we'll use EC2 run command to run a simple command on our instances. So if you'd like to get your hands dirty with run command, please join me in the AWS console. So from the console, search for IAM. Select Roles and Create role. Trusted entity type is AWS service. Under Use case, select EC2 and Next. Now we're going to search for the Amazon SSMManagedInstanceCore policy. I'm going search for SSM, hit Enter, and this is the one we want, AmazonSSMManagedInstanceCore. So select that, scroll down to the bottom, and hit Next. We'll give our role a name, and I'm going to call it my-rc-role. Then scroll to the bottom and Create role. So next, we'll launch our EC2 instances and attach this role to our instances. So search for EC2. Launch instance. The name will be Run Command Demo Instances. We'll use Amazon Linux. The instance type will be t3.micro. We can proceed without a key pair because even though we're going to be running a command on our

instances, we won't be using SSH to do that. So we don't need a key pair. And in the network settings, we can remove the security group rule that allows SSH. Scroll down to Advanced settings and expand the advanced settings. And under IAM instance profile, we're going to select the role that we created earlier. So now I'm going to close Advanced details. And then, on the Summary section on the right, change the number of instances to 2 and Launch instance. And as soon as our instances have initialized, we will be ready to continue. I'm going to select Instances, and I'm just going to wait for them to finish initializing. And if your screen hasn't refreshed, just refresh using this button. And after your status checks have passed, you are good to continue. So now, let's head to Systems Manager. And if you remember, Systems Manager is actually a suite of lots of different management tools that give you operational oversights and allow you to take action on AWS resources. So if you take a look at the left-hand menu and then scroll down under Node Management, this is where you'll find run command. So select Run Command. Select Run command. And under Command document, this is where you can select the kind of command that you want to run. Now Systems Manager manages all the different commands as these command documents, and these are simply predefined actions that you can use to perform an operational task on your AWS resources. You can filter by platform by selecting the search box, select Platform type, and select Linux, and these are all the different kinds of command documents that are available for Linux. So you can apply an Ansible playbook, apply a Chef recipe, configure an AWS package, or configure Docker, etc., and there are loads of others as well. But we are going to run our own command, and you can also run your own shell scripts as well. So in the search window, I want you to search for shell and hit Enter, and this is the one that I'm looking for, AWS-RunShellScript. So select that and scroll down. Under Document version, we'll stick with the default. And then down here under Command parameters, this is where you can specify either a shell script or a single command that you want to run on your instances. And we're going to just run a simple command to display our network interface settings on our instance. So I type ifconfig -a, and you might also know this command as ifconfig, however you like to say it. Then, scroll down, and you can optionally set the path to the working directory on your instance. We'll just leave that as empty. You can also configure an execution timeout. The default is 1 hour, so let's stick with that. Scrolling down to Targets, this is where you can specify the instances that you want to run the command on, and we're going to select our instances manually. Down here are the instances that are available and registered with Systems Manager, and it should automatically have found your instances. And if you don't see your instances down here, it's usually because you forgot to add the instance profile. So make sure when you launch your instances, you add the instance profile. Now we want to run our command on all of our instances, so select all of them using this checkbox. Scrolling down, we can optionally add a comment and specify a timeout. Rate control allows you to specify the number or percentage of targets that you want to execute the command on simultaneously. Under Output options, we can write the command output to an S3 bucket. I'm just going to disable that though. We can add a CloudWatch alarm as well to trigger while the command is running. And we can also enable SNS notifications, so we can receive a notification about the command status. And then down here under AWS command line interface command, it tells you that you can perform the same actions on this page using the AWS CLI. And they've really helpfully provided you with the CLI command that you would use to do that. Pretty cool, right? So you can run the same command using the command line interface instead of using the console. So if you're happy with all of that, click Run. And this is going to run our command on the two instances that we selected, and it should only take a few moments. I'm going to refresh my screen by using this button, and there we go. It's been successful. And if we scroll down and select one of our targets, just select the first one, we can view

the output of our command. And there is our command output, and you can even download it to your local machine for analysis, or you can copy it as well using this button. And if any errors occurred, they're going appear down here. So that is run command, and I hope you can see that it's a really easy way to run the same command on multiple EC2 instances at the same time without having to log into each one individually. And for the exam, just remember that you can use run command to securely run a command on multiple EC2 instances simultaneously without logging into each one. And you will not need to configure SSH, RDP, security groups, or use a bastion host. So that is it for this lesson. If you have any questions, please let me know. Otherwise, please join me in the next lesson. Thank you.

Understanding OpsWorks

Hello, Cloud Gurus, and welcome to this lecture which is going to cover AWS OpsWorks. And we'll begin with covering what is OpsWorks? What is configuration management? What to do if you are already familiar with Chef. What to do if you are already using Puppet, OpsWorks Stacks, and my exam tips as well. So what is OpsWorks? Well it is a configuration management service allowing you to centrally manage the operating system and/or application configurations on EC2 and on-premises systems as well. And there are 3 different offerings available. So you've got OpsWorks for Puppet, OpsWorks for Chef, and OpsWorks Stacks as well. And if you haven't heard of Puppet or Chef, these are 2 of the most popular and widely used configuration management tools out there. And because they allow you to automate the configuration of your systems, they make it really, really easy to maintain configuration consistency across your estate. And you may have heard this practice referred to as configuration of code. So what is configuration management? Well, it allows you to automate the configuration of your systems using code and the kind of things you might want to automate are your operating system configuration, your application configuration, and package or software installation as well. But what do you do if you are already familiar with Chef? Then use AWS OpsWorks for Chef Automate. And if you are already using Puppet, then use AWS OpsWorks for Puppet Enterprise. And the advantage of using OpsWorks is that you don't need to build, manage, or maintain your own configuration management system. And this allows you to focus instead on managing your core business systems. But what about OpsWorks Stacks? Well OpsWorks Stacks allow you to model your application as a stack made up of different layers. So think about an application which consists of a load balancing layer, an application server layer, and a database layer. With OpsWorks Stacks, each layer can be configured using Chef recipes and these Chef recipes define how to configure the EC2 instances and other resources that make up your application. For example, you can automate the installation of Java, deploy and configure your application, install MySQL, or run a Bash or PowerShell script, etc. And you can manage everything using OpsWorks Stacks. So on to my exam tips. Well just remember that OpsWorks is an automated configuration management service and it is compatible with Puppet and Chef. So if you hear anything in the exam relating to Puppet or Chef, then I want you to think about OpsWorks. And when you think about configuration management, this is all about configuration as code. And the kind of things that you can manage with OpsWorks are your operating system configuration, application settings, and packages and software installations as well. OpsWorks is a managed service, and it provides you with managed instances of Puppet or Chef. And then there's also OpsWorks Stacks, which allows you to model your application as a stack consisting of multiple layers. For example, a database layer, web server or application layer, and a load balancing layer as well. And you can configure these layers using Chef recipes and OpsWorks Stacks will handle the

configuration of everything for you. So that's it for this lecture. If you have any questions, please let me know. Otherwise I will see you in the next lecture. Thank you.

Section Review: Deployment, Provisioning, and Automation Summary - Part 1

Hello Cloud Gurus and welcome to this lesson, which is part 1 of the deployment, provisioning, and automation summary beginning with EC2. And just remember that EC2 is like a virtual machine hosted in AWS instead of in your own data center. You can select the capacity that you need right now, grow and shrink whenever you need, pay only for what you use, and wait minutes to deploy your infrastructure, not months. Elastic Block Store provides highly available and scalable storage volumes that you can attach to your EC2 instances. And there's a few different types of EBS volumes to be aware of, beginning with solid state disks or SSD volumes. So, first of all, we've got gp2, which is a general purpose SSD volume suitable for boot disks and general applications, and you get up to 16,000 IOPS per volume. There's also gp3, which is the latest generation. You get a baseline of 3,000 IOPS minimum for all volumes up to 16,000 IOPS per volume. And at the moment, gp3 is 20% cheaper than gp2 because they're trying to get everybody to move to the latest generation. But if your application requires greater than 16,000 IOPS per volume, then you will need to move to a provisioned IOPS volume. And with provisioned IOPS, you've got a choice between io1 and io2. And io1 is going to become the legacy option eventually, but it is still available. You may still see it in the exam. So this is suitable for OLTP and latency-sensitive applications. And with io1, you're getting 64,000 IOPS per volume. And of course, provisioned IOPS SSDs are the high performance and most expensive options. We've also got io2, which is the latest generation. It's also suitable for OLTP and latency-sensitive applications. But the difference with io2 is that you get 10 times more IOPS, up to 64,000 IOPS per volume, and that is the latest generation. Now we've also got io2 Block Express. This one is suitable for the largest, most critical, high-performance applications like SAP HANA, Oracle, SQL Server, IBM DB2, and you get a massive 256,000 IOPS per volume. And this combines the performance of a SAN with the convenience of being in the cloud. Now there's also HDD volumes as well, and there are two different options with HDD. So firstly, we've got st1, which is also known as Throughput Optimized HDD, and these are suitable for big data, data warehouses, ETL workloads, and have a maximum throughput of 500 MB/s per second per volume. There's also sc1, which is cold HDD. This one has a lower throughput, so the maximum throughput is 250 MB/s per volume. And this is suitable for less frequently accessed data, so think cold data, and it also cannot be a boot volume. A bastion host enables you to connect to private instances in your VPC from an untrusted network using SSH or RDP. A bastion is in a public subnet and is reachable from the internet, and you will need to remember to configure the security group associated with the private subnet to enable SSH or RDP access from the bastion. Onto elastic load balancer, and there are a few different types of elastic load balancer to be aware of, and you will need to understand the differences. So first of all, we have Application Load Balancers, which load balance for HTTP and HTTPS, and you get intelligent load balancing. So you can route requests to a specific web server based on the type of request. And just have a think about my example of a car dealership website where they're offering sales and loan agreements and service or repairs, and you can use an application load balancer to route requests to a specific server based on the HTTP header. Network load balancers provide high-performance load balancing for TCP traffic, and these are great for low-latency applications. There's also classic load balancer, the legacy option, supporting HTTP and HTTPS and TCP as well. And even though it's the legacy option, it may still appear in the exam. Then finally, we have gateway load balancers,

which provide load balancing for third-party virtual appliances like firewalls and intrusion detection and prevention systems. And if you need to find the IPv4 address of your end user, then look for the X-Forwarded-For header. And of course, if access logging is enabled, then we can also check the load balancer access logs as we saw in the demo. Onto load balancer error messages. And if you see any questions in the exam relating to HTTP codes and HTTP error messages, then I want you to ask yourself, is this a client-side error or a server-side error? And just remember that a 400 error is a client-side error, and a 500 error is a server-side error. And my way of remembering it is that the number 5 looks a little bit like the letter s. Well, it does if I take my glasses off anyway. And as a sysops administrator, we are more concerned with a 500 error because that's the errors on the server side that we actually have control over. And there's a few common error messages that you should be aware of. So firstly, we've got the HTTP 504 Gateway Timeout. The elastic load balancer could not establish a connection to the target. So, something is wrong with your application, and you will need to identify it where it's failing. Next, we have HTTP 502, Bad Gateway. And this means that the target host is unreachable. So you'll need to check, for example, that your security groups are allowing traffic between the load balancer and the target hosts. And then finally, we've got HTTP 503 Service Unavailable, no registered targets. So go in and check that you've correctly registered your targets and that they have successfully registered. Onto elastic load balancer access logs. And just remember, the elastic load balancer access logs capture information relating to your incoming requests to the elastic load balancer. They're disabled by default, and you'll need to go in and enable them. And the access logs are stored in S3. So they're encrypted and stored in an S3 bucket and then decrypted when you go to access them. We use sticky sessions to override the load balancing algorithm, and sticky sessions use cookies to identify a user session and then send requests that are part of the same session to the same target. And these are great for applications which cache session information locally on the web server. So consider shopping carts, online forms, or even a training website where we don't want to log out our customers halfway through a task. Onto elastic load balancer CloudWatch metrics. And the elastic load balancer is going send metrics into CloudWatch by default. And by default, you get metrics relating to healthy host count, unhealthy host count, request count, target response time, and HTTP status codes as well. And that's just some of the CloudWatch metrics, but these are the important ones to remember for the exam. Onto EC2 Image Builder. And EC2 Image Builder automates the process of creating and maintaining Amazon machine images and container images and is a four-step process. So you begin by selecting a base operating system image. You then customize it by adding the software you'd like to install. You test the AMI and then distribute it to your chosen region. And EC2 Image Builder handles the entire process for you, but there is some terminology that you will need to be aware of. So first of all, we have image pipeline, and that defines the settings and the process that you're configuring within Image Builder. The image recipe is the source image and build components. And if you remember, build components actually refers to the software or packages that you would like to include in your AMI. So that is it for this lesson. If you have any questions, please let me know. And if you're ready for part 2, then please join me in the next lesson. Thank you.

Section Review: Deployment, Provisioning, and Automation Summary - Part 2

Hello Cloud Gurus, and welcome to this lecture, which is part 2 of the Deployment, Provisioning, and Automation Summary, beginning with CloudFormation. And just remember that CloudFormation allows you to manage, configure, and provision AWS infrastructure as YAML or JSON code. And you will need to remember the different sections of the CloudFormation template.

So we've got parameters, which allows you to input custom values, for example, input the name of an SSH key. Conditions allows you to provision resources based on environment. The resources section is mandatory and describes the AWS resources that CloudFormation will create. Mappings allows you to create custom mappings, for example, mapping a region to an AMI that you would like to use in that region. And then we also have the transform section, which allows you to reference code located in S3, for example, Lambda code to provision a Lambda function or reusable snippets of CloudFormation code. CloudFormation StackSets allow you to create, delete, and update CloudFormation stacks across multiple AWS accounts and regions using a single operation. And to do this, you will need to set up some cross-account rules if you're working with multiple accounts. So for the administrator account, you can use the AWSCloudFormationStackSetAdministrationRole, which is allowed to assume the AWSCloudFormationStackSetExecutionRole to provision resources in the target accounts. And then once you've created resources in these other accounts, you can use a Resource Access Manager to share the resources between your accounts. For example, you could share resources like EC2 instances, S3 buckets, and EC2 Image Builder images. When it comes to so troubleshooting CloudFormation, you can use the CloudFormation console to view the status of your stack and any error messages. And common error messages include insufficient permissions. And if you see an error like that, you'll need to go in and add permissions for the resources that you are trying to create, delete, or modify. If you see the resource limit exceeded error, then request a limit increase, or you could also delete any unnecessary resources and retry. And if you see Update_Rollback_Failed, then you will need to fix the error causing the failure and retry. For example, if you deleted or changed a resource manually, which you then try to roll back or delete using CloudFormation, then CloudFormation will throw an error because the resource is not in the state it's expecting. So you will need to get your stack back to the state that CloudFormation is expecting in order to continue. Now, if you are troubleshooting CloudFormation, then the first place to check is the CloudFormation console. And under Status Reason, it will give you some pretty good information that is going to give you a clue as to what went wrong. And in this example, I used an incorrect AMI, which did not exist in the region that I was operating in. Onto CloudFormation best practices. You should be controlling access to CloudFormation using Identity and Access Management. Be aware of your service limits because if you hit a limit, then CloudFormation will fail to create your stack. Avoid manual updates because they're going to cause errors when you try to update or delete the stack. Use CloudTrail to track all changes in CloudFormation, along with who made them and when. And use a stack policy to protect critical stack resources from unintentional updates and mistakes caused by human error. On to blue/green deployments. And a blue/green deployment strategy is a low risk strategy. So blue is the current version of your application while green is the new version. And after testing is successfully complete, live traffic can be directed to the new version and the old version will become deprecated. Rolling back is fast and easy because if something goes wrong after the new version is being used in production, then simply redirect customer traffic back to the original environment. A rolling deployment allows you to deploy new application versions and other changes in batches. It's very cost-effective because you only need one environment and you can set the batch size and the minimum number of servers to keep in service in your environment. But the trade-off is that it comes with some added complexity because for a while, you will be operating with a mixed environment with some instances running your old version and some running the new version. And in addition to that, rolling back will involve a redeployment. When it comes to canary deployments, remember the canary in the coal mine. And just like the canary in the coal mine, you get an early

warning system that can indicate that something is wrong in your application. And with a canary deployment, you deploy the new version to a small number of servers, direct a small proportion of customer traffic to the new service, for example, 10%. And this allows you to test your application with a small proportion of real customers before you roll out to everybody. And a canary deployment is a great way to reduce risk of deploying a new application version when you are working with a production environment, for example, if you're installing new code or new packages. And to roll back, simply direct 100% of users back to the original version. Onto OpsWorks. And just remember, OpsWorks is an automated configuration management service compatible with Puppet or Chef. And if you see anything in the exam relating to Puppet or Chef, then I want you to think of OpsWorks. And it allows you to use a configuration-as-code approach to managing the configuration of your operating systems, application settings, and packages as well. You get managed instances of Puppet or Chef, and there is also OpsWorks stacks, which allows you to model your application as a stack consisting of multiple layers, for example, a database layer, a web server layer, application layer, and load balancer layer, et cetera. Onto Systems Manager Run Command. And imagine you are a SysOps Administrator supporting hundreds of servers and your manager asks you to run a command to check the network configuration on each server. Well, this is something that you can really easily do using Run Command. So using Run Command, you can run a command of your choice on a group of servers simultaneously without having to log in to each one individually. And there's also Systems Manager, which we can use to patch multiple EC2 instances simultaneously. So imagine you've got hundreds of servers to patch. It's going to be pretty boring and laborious if you have to patch each one of these manually. So you can just go ahead and use Patch Manager to patch a group of servers simultaneously. So that's it for this lecture. If you have any questions, please let me know. Otherwise, we will see you in the next section of the course. Thank you.

Monitoring, Logging, and Remediation

Section Introduction

Hello, Cloud Gurus, and welcome to this section of the course, which is going to cover monitoring, logging, and remediation, and we'll begin by looking at CloudWatch. We'll also cover CloudTrail, EventBridge, Config, Systems Manager, and Health Dashboards as well, so if you are ready to get started with monitoring, logging, and remediation with AWS, I'll see you in the next lecture. Thank you.

Introduction to CloudWatch

Hello, Cloud Gurus, and welcome to this lecture, which is going to introduce CloudWatch. And we'll begin with: What is CloudWatch? What can CloudWatch monitor? We'll take a look at the CloudWatch agent, CloudWatch and EC2, metric frequency, and my exam tips. So what is CloudWatch? Well, it's a monitoring service, which allows you to monitor the health and performance of your AWS resources, as well as the applications that you run on AWS, and it can be used to monitor applications that are running in your own data center as well. So what can CloudWatch monitor? Well, in terms of compute, you can monitor EC2 instances, auto scaling groups, elastic load balancers, route 53, using the route 53 health check, and Lambda functions, as well. When it comes to storage and content delivery, we can monitor EBS volumes, storage gateway, which we'll come to later on in the storage section of the course, and CloudFront. For

databases and analytics, we can monitor DynamoDB tables, ElastiCache nodes, RDS instances, Redshift, and Elastic Map Reduce clusters. And some of the other things we can monitor, are things like SNS topics, SQS queues, API gateway, and estimated AWS charges. And one of the cool things about CloudWatch is that it provides this CloudWatch agent, which allows you to define your own custom metrics. And you can also install the CloudWatch agent on systems located in your own data center. And in combination with CloudWatch logs, you can monitor operating system and application logs as well. Now when it comes to CloudWatch and EC2, all EC2 instances send key health and performance metrics into CloudWatch. And the default EC2 host-level metrics are things like CPU utilization, network number of packets received and sent by the instance, disk read and write operations, and status checks, as well. So, whether your EC2 instance has passed its status check. And all of these are host-level metrics. The metrics are stored indefinitely, and you can retrieve data from any EC2 or elastic load balancer instance, even after it's been terminated. Now, when it comes to gathering operating system-level metrics, by default, EC2 does not send any operating system-level metrics into CloudWatch. And if you would like to start collecting operating system-level metrics, then you will need to install the CloudWatch agent on your instance. So by installing the CloudWatch agent on your EC2 instances, you can start collecting operating system-level metrics and send them into CloudWatch. And when I say operating system-level metrics, what I mean is the kind of metrics that are only available from within the operating system of your instance. So things like memory usage on your instance, processes running inside your instance, the amount of free disk space on the instance, and CPU idle time within your EC2 instance, et cetera. Now, by default, EC2 sends metric data to CloudWatch in 5-minute intervals. However, for an additional charge, you can enable detailed monitoring, which is going to send metrics at 1-minute intervals. And for custom metrics, the default is 1-minute intervals, but you can configure high-resolution metrics that will send metrics at 1-second intervals instead. So onto my CloudWatch exam tips. And just remember that CloudWatch is all about monitoring the performance and health of your systems. And the default host-level metrics include metrics around CPU utilization, network, disk, and status check, and anything outside of that will require the CloudWatch agent. So you will need to use the CloudWatch agent for operating system-level metrics, for example, memory usage on your instance, processes running on your instance, and CPU idle time. So that's it for this lecture. If you have any questions, please let me know. Otherwise, I'll see you in the next lecture. Thank you.

Demo: Creating CloudWatch Dashboards

Hello Cloud Gurus and welcome to this lesson. And in this lesson, we'll be creating our own custom CloudWatch dashboard. We'll begin by launching an EC2 instance, and we're going to generate some activity on our instance to create some CloudWatch metrics. Next, we'll create a custom CloudWatch dashboard, and we'll explore the metrics that are available for our EC2 instance. And then finally, we're going to add some widgets to our dashboard to display the metrics of our choice. And we'll be adding the CPUUtilization widget, NetworkPacketsIn, and NetworkPacketsOut as well. And we'll save our dashboard so that we can come back and check our metrics for our instance at any time. So if you'd like to get your hands dirty with CloudWatch dashboards, please join me in the AWS console. So in the console, we're working in the Northern Virginia region. Search for EC2. Launch instance. Call it CloudWatch Demo. We'll use Amazon Linux. Instance type is going to be t3.micro. And we'll proceed without a key pair. And then go ahead and launch instance. Select your instance ID. And as soon as our instance has finished initializing, we are good to go. I'm going to

refresh my view, and now my instance has passed its status checks. So now, I'm going to select the instance and select Connect and Connect. And we're using EC2 Enstance Connect to connect to our instance. So now we're on the instance, I just want to generate some CPU activity on this instance so that we can see some metrics appearing in CloudWatch. So type while true then semicolon space do space echo semicolon space done and then hit Enter. And this is going to create an infinite loop, which is going to result in high CPU load on our system, and we'll be able to see the effect of this command in CloudWatch. So once you've typed that command, I'm going to copy my instance ID from down here. Then, search for CloudWatch. I'm going to open it in a new tab, and we are going to create our own custom CloudWatch dashboard. So select Dashboards, Create dashboard. I'm going to call it my-production-systems and Create dashboard. And this is where we can start adding widgets to our dashboard. So I'm going to add a line graph, and this is a great one if you want to compare metrics over time. So select Line graph, and we'll be creating this widget based on CloudWatch metrics. So select Metrics. And right now our graph is empty, so we need to select the metrics that we want to add into the graph. And if you scroll down and select EC2, then select Per-Instance Metrics. And now, we can search based on our instance ID. So if you copied your instance ID, you can just paste it in here and hit Enter, and these are all of the default metrics provided by CloudWatch for our EC2 instance. So we've got metrics relating to CPU usage, EBS volumes, so relating to the disks attached to our system. There's also some network-related metrics. And then down here, there are the status checks as well. But I'm interested in CPU utilization, so I'm going to select that one. And then, if you scroll up, you'll see that that's been added to our graph. So now, select Create widget, and that's added our widget to our dashboard. Now I also want to add widgets for the NetworkPacketsIn and PacketsOut as well. So let's quickly add both of those as well. Select the plus sign on the top right-hand side. Select Line, Metrics. Scroll down. Select EC2, Per-InstanceMetrics. We'll search for our instance ID again and select NetworkPacketsIn. Create widget. And I'm going to do the same again and select NetworkPacketsOut. Select Per-Instance Metrics, search for our instance, and there we go, NetworkPacketsOut and Create widget. So that is our dashboard created. And if you want to come back to the dashboard again, you can just select Save dashboard, and then your custom dashboard is going to appear in this dashboards view. So you can just select your dashboard and come back to view the metrics. And if you're not seeing any metrics appearing in your dashboard, then just be patient and wait a few minutes because it does just take a few minutes for the metrics to be delivered to CloudWatch. And by default, EC2 is sending metrics into CloudWatch every 5 minutes. Now you can even add widgets for different regions if you are permitted to do so. So from your dashboard, select the plus sign. Select a widget type. I'm going to select Metrics. And then down here under Metrics, you can just select the region that you want to add the widget for. But if you are working in our AWS sandbox, then you will not have permission to add any metrics for a different region other than us-east-1 because we only let you work in us-east-1. But if you were working in your own AWS account, then you would absolutely be able to add metrics for another region, and it's really easy to do that. So that is CloudWatch dashboards. And for the exam, remember that CloudWatch dashboards allow you to customize your view within CloudWatch so you can create a dashboard which monitors the metrics that are meaningful to you. For instance, you can monitor all of your production systems in one single dashboard. A dashboard can be multi-region, so you can display metrics from any region. And you just select the region that you want when you're adding the widget. And always remember to save your dashboard when you've finished because CloudWatch doesn't automatically save it for you. So once you've configured it as you want it, just remember to hit the Save button. And that is

all for this lesson. If you have any questions, please let me know. Otherwise, I will see you in the next one. Thank you.

Exploring CloudWatch Logs

Hello Cloud Gurus and welcome to this lecture where we're going to be taking a look at CloudWatch Logs. So we'll begin with an overview of CloudWatch Logs. We'll take a look at some CloudWatch Logs terminology, retention settings. We'll look at a CloudWatch log scenario and cover my exam tips as well. So what is CloudWatch Logs? Well, CloudWatch Logs provides centralized logging for application logs, for example Apache logs, system logs, for example operating system logs running on EC2, like var log messages if you're running a Linux system and logs for AWS services like CloudTrail or Route 53. And with CloudWatch Logs, you can view, search, and filter the logs so you can search based on error codes and messages within the log, for example a 404 status code in an Apache log. And you can also configure CloudWatch Logs to send you notifications. So you can receive a notification whenever the rate of errors exceeds a threshold that you specify. So we can use CloudWatch Logs to monitor our log files, and we can monitor and troubleshoot our applications using the existing system or application log files. And you can customize CloudWatch Logs for your own application. So you can monitor your logs in near real time for specific phrases or error codes or patterns that make sense for your application. And to do this, you will need the CloudWatch agent to be installed on your EC2 instance. So let's take a look at some important CloudWatch Logs terminology, beginning with log events. And a log event consists of an event message and a timestamp. We've also got log stream, which is a sequence of log events. For example, think of an Apache log from a specific host sending a sequence of log events into CloudWatch, and every log stream must belong to a log group. So using a log group, this allows you to group streams together and centrally manage their retention, monitoring, and access control settings. And there is no limit on the number of log streams that can be added to a log group. Moving on to retention settings. Now by default, CloudWatch Logs are kept indefinitely, but you can set your own retention period for anything between 1 day and 10 years, and expired log events will be automatically deleted. And when you configure retention settings, they can be applied to an entire log group, so you don't need to go in and set the retention settings for every single EC2 instance. So let's take a look at a CloudWatch log scenario. Just imagine you have a couple of EC2 instances running Apache, and they are sending log events from the Apache logs into CloudWatch, a sequence of log events from the same origin, so from Apache on a single EC2 instance. That sequence of log events makes up a log stream, and we can centrally manage multiple log streams as a single log group. So this means we can centrally manage the retention, monitoring, and access control settings of multiple log streams as one log group. So, onto my exam tips. Just remember that log events consist of an event message and a timestamp, for example, events coming from an Apache log. A log stream is a sequence of log events coming from the same source, for example an Apache log coming from a specific EC2 instance. And each log stream must belong to a log group. And a log group is simply a group of log streams, which share the same retention, monitoring, and access control settings, for example multiple web servers running Apache and sending log events into CloudWatch. So that's it for this lecture. If you have any questions, please let me know. Otherwise, I'll see you in the next lecture. Thank you.

Demo: Collecting Metrics and Logs Using the CloudWatch Agent

Hello Cloud Gurus and welcome to this lesson where we'll learn all about collecting metrics and logs using the CloudWatch agent. And we'll begin by launching an EC2 instance, and we're going to attach an IAM role with permission to send CloudWatch agent metrics into CloudWatch. And we'll be using the CloudWatch agent server policy, which is an AWS managed policy that is going to give us those permissions. Next, we're going to install the CloudWatch agent on our EC2 instance, and we'll configure the agent to send operating system metrics and logs into CloudWatch. And then finally, we're going to view the metrics in CloudWatch, and we should be able to see the EC2 default metrics, as well as metrics and logs that are sent by the CloudWatch agent. So if you're ready to get your hands dirty with CloudWatch, then please join me in the AWS console. As usual, we'll do everything in the Northern Virginia region. And first of all, I'm going to search for IAM. On the left-hand side, select Roles, Create role. Trusted entity type is an AWS service. Use case is EC2. Hit Next. Under Permission policies, we'll search for the CloudWatchAgentServer policy. And there it is. And this policy is going to give the CloudWatch agent on our EC2 instance permission to send metrics into CloudWatch. So hit Next. The role name is going be cloudwatch-agent-role. Scroll down to the bottom and create role. So now we are ready to create our EC2 instance. Come to EC2, Launch instance. We'll call it CloudWatch Demo. Use Amazon Linux. Instance type will be t3.micro. We can proceed without a key pair. Then scroll down to Advanced details. And then under IAM instance profile, this is where you need to select the role that we just created. Then scroll down until you get to the User data section. And this is where we're going to add a Bootstrap script. Here is our script, just a couple of lines. And the first line just tells the operating system to use the Bash interpreter, and then we're going to update the operating system with the latest packages using yum update -y. And you will find this file with all the commands linked in the resources section of the course. So come back to your AWS console, paste in the Bootstrap commands, and Launch instance. Select your instance ID, and we'll wait a few moments to allow it to finish initializing. And it may just take a little longer than usual because we need to wait for it to finish the yum update. When your instance is ready, select it using the checkbox by the side and select Connect and Connect. Now that we're logged into our instance, we're just going to run a few simple commands to configure the CloudWatch agent. And there are a few different commands that you'll need to run, and you'll find everything linked in the resources for this lesson. So the first thing we'll need to do is install the CloudWatch agent. Once that's completed, I'll clear my screen. The next command we need to run is this one, which helps us to configure the CloudWatch agent. And when we run this command, it's going to ask us a whole bunch of questions that we will need to answer within this configuration wizard. But the main thing to remember is that we need to say no when it asks us if we want to monitor collectd because collectd is not installed on our system. And the log file that we want to monitor is var/log/messages. So just keep that in mind when you're running the configuration wizard, but we're going to go through it all together anyway. So just copy that command, come back to your instance, and paste in that command and hit Enter. And this is where we need to start answering all of the questions. So first of all, it's asking which operating system are we running, and we'll accept the default, which is Linux, so hit Enter. We're using EC2, so accept the default and hit Enter. We'll run our agent as root. So accept the default. Hit Enter. Do we want to turn on the StatsD daemon? Well, yes we do. So we'll accept the default choice and hit Enter. And StatsD is just a daemon that collects metrics on your system. We accept the default port, so hit Enter, and the collection interval as well, so hit Enter, and the aggregation interval as well. Now for this one, it's asking us if we want to monitor metrics for collectd, and it must be installed.

Otherwise, the agent will fail to start. We don't have it on our system. So for this one, I'm going to type number 2 for no and hit Enter. Do we want to monitor any host metrics? Yes, we do. So accept the default and hit Enter. Do we want CPU metrics per core? Yes, we do. Hit Enter. Do we want to add EC2 dimensions like ImageId, InstanceId, etc.? Yes, we do. So hit Enter. Do we want to aggregate them? Yes, we do. Hit Enter. Do we want high-resolution metrics? Well, let's just go for the default and hit Enter. Which default metrics config do you want? Let's go for the standard config. So select number 2 and hit Enter. Are we satisfied with the above config? Yes we are, so hit Enter. Do we have an existing CloudWatch agent? Well, we don't. So let's go with the default choice of number 2, which is no, and hit Enter. Do we want to monitor log files? Yes we do, so hit Enter. And at this point, we need to enter the path to the log file that we want the CloudWatch agent to monitor, and we're going to use it to monitor the operating system log, which is located in var/log/messages. So just type var/log/messages and hit Enter. For the group name, let's accept the default. So, hit Enter. For the log stream name, we accept the default as well. For log group retention, accept the default for this one as well. And now it's asking if we want to specify additional log files to monitor. And if we were running an application on this EC2 instance, then this is where we could specify any additional log files, so application logs that we wanted CloudWatch to monitor. So we could add an application log file in here. As we don't have any application running on here, I'm just going to type 2 for no and hit Enter. Do we want to store the config in SSM parameter store? No, we don't. So, select number 2 and hit Enter. And that is our CloudWatch agent configured. So now, if we change directory to /opt/aws/amazon-cloudwatch-agent/bin and then type ls, you should have this config.json file. And this is our CloudWatch agent configuration file that we just created using the wizard. If we open up that file, here's the path to our messages file. And then down here are the metrics that we're going to collect. So we've got our CPU metrics here. Here's our disk metrics, memory metrics, metrics associated with StatsD. And we've got some swap metrics as well. So now, we are ready to start our CloudWatch agent. I'm going to come back to my commands, and this is the command that we're going to use to start the CloudWatch agent. So copy that. I'll clear my screen, paste, and hit Enter. And if it was successful, this is what your screen should look like. So first of all, it's going to validate your configuration. And then, it's going to start the CloudWatch agent. Now, at the moment, we don't have anything running on our system. So I want to generate some CPU activity so that we've got something interesting in our metrics that we can then take a look at in CloudWatch. So we're going to install a utility called stress, and we can then use that to generate some CPU load. And there's just a couple of commands that we need to run in order to do that. And first of all, we need to add the extra packages for enterprise Linux repository, and it's called epel. So just copy this command. Back on our instance, I'll clear my screen and paste the command in there. Then, in the next command, we're just going to install the stress utility. So copy the command and paste. And then finally, we can run stress --cpu and the number 1. And this is just going to create some CPU load on our system. So just run that command and leave it running. And then after a few minutes, we should start to see some metrics appearing in CloudWatch. So let's head back to our AWS console, search for CloudWatch, select Metrics on the left-hand side, and select All metrics. If we scroll down here, this is where we'll find the default EC2 metrics. So select EC2. Select Per-Instance Metrics. And in here, we can search for the metrics based on our instance ID. So I'm going to find my instance ID, there it is, copy my instance ID, come back to CloudWatch, and search based on the instance ID. And these are all the default metrics that you get for your EC2 instance. But what about the CloudWatch agent? Well, if we select All over here, under Custom namespaces, this is where you'll find the metrics that are being sent by the CloudWatch agent on our

EC2 instance. And if we select one of these custom namespaces, we can take a look at the metrics provided. And these are only metrics that you can get from within the operating system itself. So it's things like disk usage, memory, and swap. You've also got some CPU stats as well. And we can select some of these metrics. And when we select them, they will get added to the graph above. And there we go. We're starting to see some metrics come through there. But it does just take a while for the metrics to be populated. So that is the CloudWatch agent metrics. But what about the logs? Well, if you head over to Logs on the left-hand side, select Log groups, this is where we are going to find our logs Select Messages. Scroll down to Log streams and select our log stream. It will take you straight into the log, and these are all the messages that are appearing in var/log/messages. And you'll notice here that we can even see when the CloudWatch agent was started on our EC2 instance. And down here, we can see where we installed stress on our system. So these are all the operating system messages that have appeared in the var/log/messages file. So for the exam, just remember that the CloudWatch agent allows you to send operating system level metrics and logs into CloudWatch. And when we talk about operating system metrics, it's things like CPU, memory, swap, and disk usage. And when it comes to logs, you can send operating system logs like var/log/messages, and you can also define any application log files that you like. For instance, if you were running Apache on your system, then you might like to send your access logs into CloudWatch. So you could specify your var/log/httpd/access_log and have that sent into CloudWatch as well. So that's it for this lesson. If you have any questions, please let me know. Otherwise, I will see you in the next one. Thank you.

Demo: Creating CloudWatch Metric Filters

Hello Cloud Gurus and welcome to this lesson where we'll be taking a look at CloudWatch metric filters. We'll begin by creating a very basic lambda function. And whenever you create a lambda function, a new CloudWatch log group is automatically created. And this is going to allow our function to send logs into CloudWatch. Next, we'll test that our function is working. And when we test the function, this is going to generate some log events that will appear in CloudWatch. We'll then take a look at the options that are available with CloudWatch metric filters, and we're going to explore how to monitor for specific terms or values appearing in our log files. And then finally, we'll create our own metric filter. And we're going to create a filter that's going to cause CloudWatch to alert us if any errors appear in our log files. So if you'd like to join me in the console, we'll get our hands dirty with CloudWatch metric filters. So from the console, search for lambda. Create function. We'll author from scratch. Call it my-lambda-function. Accept all the rest of the defaults. Scroll down to the bottom and Create function. When your function is created, scroll down to Code source. Double-click on index.mjs, and there is our code. I'm just going to update this message here and deploy. Next, we'll run a test. So select Test, Configure test event. The event name is te1. Scroll down to the bottom and save. Then, select Test. And I'm just going to test it a few times because every time that we test, it's going send logs into CloudWatch. So now, we can search for CloudWatch. I'll open it in a new tab. Using the left-hand menu, select Logs and Log groups. And this is the log group that was automatically created when we created our function. And this log group is used to collect logs from our function. So select your log group, scroll down, and you'll see that you've already got some logs. So if we click on this log stream, these are the log events for our function. So this is what lambda is automatically logging for us. So first of all, we've got the time when the request started, when it ended, and a report as well. And in the report, it's actually telling us the duration. And in the report, it's telling us the duration that the lambda function

ran for, how many milliseconds we've been billed for, the amount of memory allocated to the function, and the maximum amount of memory used. So now let's create our metric filter. And if we click on our log group up here and select Actions, Create metric filter, and it says here we can use metric filters to monitor events in a log group as they are sent to CloudWatch Logs. So you can monitor and count specific terms or extract values from log events and then associate the results with a metric. And once you've done that, you can also get CloudWatch to alert you and trigger an alarm. So down here under Filter pattern, this is where we can specify the terms or the patterns that you want your metric filter to match from your log events. For example, we can match log events that contain terms like ERROR, [INFO], or Warning. We can also match based on an HTTP response code or even specific JSON log events. And you can define your own patterns as well. These are just a few examples that it's giving you. And you can also filter based on any text you like. Let's say you wanted to filter based on a specific word, and you wanted to be alerted if that word ever appears in your log file. Well, you can use a filter pattern to do that, and I'm going to search for the word duration because I know that that definitely appears in my log file. Down here, we can actually test out our filter to make sure that it's working as expected. We can select the log data that we want to test with. So select the log data that's been produced today. Then select Test pattern, and then scroll down to see the results. And if we take a look at our test results, you can see that it selected the entries which include the word duration. So I hope you can see that this is a really easy way to search for custom words in your log files. So now, let's create a metric filter that's going to trigger if any errors appear in our log file. Come back to the top, and we'll change our filter pattern to error. Scroll down, and we can test by running a test pattern. But as we've currently got no errors in our log files, it's not going to find anything. So don't worry about that. So we'll hit Next. Scroll up to the top. The filter name is my-lambda-filter. Our filter pattern is error. We'll create a new metric namespace, and a namespace just allows you to group similar metrics together. Our metric name is lambda-log-errors, and the metric value is a value that's going to be published to the metric name when the filter pattern match occurs, and I'm going to set my metric value to the number 1. And my default value is going to be 0. Down here, we don't need to set a unit. But for other types of metric, it might make sense to set a unit. So now hit Next and Create metric filter. And it is as simple as that. So any time my lambda function receives an error, this metric is going to be triggered. So now, let's go and create some errors. We'll head back to our lambda function, down to our code source, open up your code source, and I'm just going to remove the very last line of our code. And that's definitely going to create an error. So now deploy and test. And we can see straight away, we have got to a couple of errors. So I'm going to run the test a few more times to generate some more log events. So I just run it half a dozen times to generate the more log events. And then after you've done that, head back to your CloudWatch tab. Back to our metric filter. We'll select our metric, lambda-log-errors. Select our metric down here. Check this box to have our metrics appear in the graph. I'm going to set our timeline to 1 hour. And you might need to refresh your screen using this button to get the metrics to appear. And don't worry if you don't see anything immediately. Just be patient and CloudWatch should pick it up in a few minutes. And there we go. CloudWatch has picked up that I have received some errors in my lambda log. And that is metric filters. So for the exam, it's really important that you understand how to configure metric filters. And hopefully, all of that was pretty straightforward. And metric filters allow you to filter for specific phrases that might be appearing in your logs, for instance warnings, errors, and HTTP status codes. You can create a metric filter at the log group level, and you can create metric filters for any log group in your account. And finally, metric filters are a really great way to receive CloudWatch alerts for specific errors, warnings, or messages that might appear in your log files. So

that is it for this lesson. If you have any questions, please let me know. Otherwise, I'll see you in the next lesson. Thank you.

Demo: Exploring CloudWatch Logs Insights

Hello Cloud Gurus and welcome to this lesson where we'll take a look at CloudWatch Logs Insights. Now, CloudWatch Logs Insights allows you to perform interactive queries and analysis on your data that is stored in CloudWatch Logs, and it actually uses its own bespoke query language. But don't worry. You don't need to learn this query language for the exam, and they give you an awful lot of example queries. So it's really easy to get started. We can use CloudWatch Logs Insights to query and filter our logs directly. And we can also use it to generate visualizations, for instance bar graphs, line graphs, or pie charts. And you can also do stacked area graphs as well. And all this is possible from within the CloudWatch Logs Insights console. But the best way to understand it is to actually go ahead and use it, and that's exactly what we'll be doing in this lesson. So first of all, we're going to create a basic lambda function. And if you remember, a new log group is automatically created whenever you create a new lambda function, and that is going to allow our function to send logs into CloudWatch. Next, we'll test that our function is working correctly. And when we test the function, that is going to generate some log events. Then, we'll explore how we can use CloudWatch Logs Insights to analyze our log data. And then finally, we're going to review some of the sample queries that are available. So if you'd like to join me in the console, we'll get started. So from the console, search for lambda. Create function. We're going to author from scratch. Call it cw-logs-insights-function. Scroll down and Create function. When your function is ready, scroll down to the function code, double-click, and I'm going to Update the message just to send a new message. Then deploy. We're going to test our function to create some log events. So select Configure test event. Call it te1. Scroll down to the bottom and save. And then we're going to run a few tests. So click on the Test button a few times just to generate some log events for CloudWatch. Next, we'll head to CloudWatch. Select Logs and Logs Insights. Select the log group at the top that is associated with our lambda function. And they've already provided this sample query that we can go ahead and run. And this is just going to select the latest 20 log events from our log. So you can go ahead and select Run query, then scroll down, and there is the last 20 log events. Now sometimes it does take a few minutes for CloudWatch to pick up your logs. So just be patient, and eventually it will catch up. And if you don't see anything appearing immediately, just wait a few minutes and try again. But just remember, up here, make sure you've got the correct time frame selected, and I always make sure it's set to about 1 hour. So now, let's take a look at some of the sample queries that they give you. On the right-hand side of your screen, select Queries, and there are loads of different sample queries that we can take a look at. So select Lambda, and down here, we can use this query to Identify the most expensive lambda requests. Here's the code, and if we want to apply the query, just hit Apply. The query will appear over here. And hit Run query. And there we go. Down here, it's listed the lambda requests in order of the most expensive. Under Sample queries, close down the Lambda section and open up VPC flow logs. And this query allows you to run a query to find out which IP addresses are using a specific protocol like UDP. Here's the code. And if you want to apply that one, you can just hit Apply, and it will appear over here for you to run. I'm going to close that down and select Common queries. And here, under Common queries, we can query for the 25 most recently added log events. And if we want to apply that, we can just hit Apply, and Run query. And there we go. It's all going appear down here. Now we can also get CloudWatch Logs Insights to visualize our data as well using this tab. So I'll click on the

Visualization tab. However, visualization is not available on every single type of query. But to help you out, they give you an example of one to get started, and here it is. So I'm just going to select that, copy it, paste it up here into my query text box, and this query just shows the distribution of log events over the last 30 seconds, But I don't think that's a very good time frame for us, so I'm going to change it to the last 4 hours. So this is going to show the distribution of log events per hour over the last 4 hours. So now hit Run query, and there is our visualization. I'm just going to close down that screen so that it appears a bit better on the screen here. Now it's currently created a line graph, but we can also create a stacked area graph, a bar chart, or a pie chart. So, that is CloudWatch Logs Insights. And for the exam, just remember that we can use CloudWatch Logs Insights to interactively query and analyze our data that is stored in CloudWatch Logs. We can query the logs directly, and we can also generate visualizations like bar graphs, line graphs, pie charts, and stacked areas as well. And they also give you loads of cool example queries. For instance, you can display the 25 most recently added log events, search your VPC flow logs to find out which IP addresses are using a specific protocol, or even find the most expensive lambda requests. So that is it for this lesson. If you have any questions, please let me know. Otherwise, I will see you in the next lesson. Thank you.

Receiving Notifications with CloudWatch

Hello, Cloud Gurus, and welcome to this lecture, where we going to take a look at how you can receive notifications from CloudWatch. And we'll begin with taking a look at CloudWatch Alarms. We'll then cover SNS notifications, serve as quotas or limits, health events, and my exam tips as well. Now with CloudWatch Alarms, you can create an alarm to monitor any CloudWatch metric in your account. For example, this could include EC2 CPU utilization or Elastic Load Balancer latency or even the charges on your AWS bill. You can set appropriate thresholds to trigger the alarms and actions that you want to be taken if an alarm state is reached. For example, you can create an alarm, which sends an SNS notification or executes an Auto Scaling policy if CPU utilization exceeds 90% on your EC2 instance for more than 5 minutes. So let's take a look at an example using SNS notifications and CloudWatch integrates with SNS to send email notifications to notify you of events within your AWS account. So just imagine you've got an EC2 instance and it's running at 99% CPU utilization. We can configure a CloudWatch Alarm, which is going to trigger if our instance is running at greater than 90% CPU utilization for over 5 minutes. So our alarm gets triggered, which in turn is going to trigger an SNS notification, which will send an email to our IT support team. But what about service quotas? Well, CloudWatch can also be used to monitor your service quotas or your limits and notify you if you are about to reach the limit. And it's actually really easy to configure. You just use the service quotas console to configure a CloudWatch Alarm to send us an SNS notification. So imagine we want to be notified if we reach 90% of the quota value for on-demand EC2 instances. This is going to generate a CloudWatch Alarm, which in turn will send an SNS notification, and send an email to our it support team. Now, we can also get CloudWatch to notify us of health events, and there's an API called AWS Health, which is actually the same API that is used by the personal health dashboard, which we're going to be using later on in the course. And the personal health dashboard just gives you a personalized view of the health status of all of the AWS services that you are using. So we can use AWS health to send health events to EventBridge, triggering a CloudWatch Alarm, which can then go on to trigger an action that we define. For example, imagine AWS Health is sending us notifications about all of the EC2 related health events. The events are sent into EventBridge, and if you've ever used CloudWatch

events before, well CloudWatch events is now called EventBridge and it actually uses the same technology. So the AWS Health events are sent into EventBridge, which triggers an alarm in CloudWatch Alarms. And CloudWatch Alarms can trigger an SNS notification to send an email to our support team. Or you might want to trigger a Lambda function to take some action on your behalf, or send a message to an SQS queue. Moving on to my exam tips, and just remember that you can create an alarm to monitor any CloudWatch metric in your account and alert you if a threshold is reached. For example, you can create an alarm, which is triggered if CPU utilization exceeds 90% on your EC2 instance for more than 5 minutes. And CloudWatch is really well integrated with SNS. And it's really easy to configure it to send an SNS notification to send an email to your IT support team. When it comes to service quotas or limits, CloudWatch can be used to monitor your service quotas or limits and notify you if you are about to reach the limit. An SNS notification can be used to email your IT support team. And CloudWatch Alarms can also alert you in the dashboard. And a great use case is to monitor your service quota or service limit for EC2 instances. For example, you may want to receive a notification to tell you that you've reached 90% of the quota value for on-demand EC2 instances. And when it comes to health events, the AWS Health API provides information about changes in the health of AWS resources. And it can be used to send health events into EventBridge. And if you've ever used CloudWatch events before, CloudWatch events is now EventBridge, and we'll be using EventBridge later on in the course. An EventBridge can trigger a CloudWatch Alarm to trigger an action of your choice, like send an SNS notification, send a message to SQS, or trigger a Lambda function. And AWS Health is the same API use by the personal health dashboard, which we will also be looking at later on in this section of the course. So that's it for notifications with CloudWatch. If you have any questions, please let me know. Otherwise I'll see you in the next lecture. Thank you.

Demo: Creating CloudWatch Alarms

Hello Cloud Gurus and welcome to this lecture where we're going to be creating a CloudWatch alarm. And we'll begin by launching an EC2 instance, and we'll enable detailed monitoring. So our instance is going to send metrics to CloudWatch at 1-minute intervals. Next, we'll create a CloudWatch alarm, and the alarm is going to trigger if CPU utilization on our instance exceeds 90% for 1 minute or longer. Next, we'll configure an email alert. So we'll configure CloudWatch to alert us by sending an email using SNS if the alarm is triggered. And then finally, we're going to max out our CPU. So we'll run a command to stress our CPU, and this should cause the CloudWatch alarm to trigger. And if our alert is working, then we will also receive an email. So if you're ready to get started, I'll see you in the AWS console. So here I am in the console, and I'll be doing everything in the Northern Virginia region. So first of all, head to Services and select EC2 under Compute. Scroll down and select Launch instance. We'll select the Amazon Linux 2 AMI. T2.micro is fine, so hit Next, and we'll accept all the defaults apart from the CloudWatch monitoring settings. So we'll enable CloudWatch detailed monitoring. And with detailed monitoring, our EC2 instance will send metrics into CloudWatch at 1-minute intervals instead of 5-minute intervals. And do be aware that additional charges apply when you enable detailed monitoring. But if you don't want to enable able detailed monitoring, don't worry. You can still do the lab. You will just need to wait a few more minutes for the CloudWatch alarm to trigger. So once you've enabled that, we can just go ahead and select Review and Launch and Launch. And we'll proceed without a key pair and Launch instance. So now select your instance ID. And while our instance is still initializing, we can go ahead and create our CloudWatch alarm. But first of all, I want to copy my instance ID to my clipboard

because we'll be using this later when we create our alarm. So now, let's head to Services, and you'll find CloudWatch under Management & Governance. So open that link in a new tab, and we are going to configure our CloudWatch alarm. So select Alarms from the left-hand menu. Create alarm. We'll select the metric that we want to base our alarm on. Scroll down and select EC2. Select Per-Instance-Metrics. And we're going to search for our instance ID. And you might just need to hit Enter in the search box a couple of times because our instance is still initializing. And once it finds your instance ID, scroll down until you find CPUUtilization. And that is the metric that we're looking for, do select that and select Metric. Scroll down. Under Statistic, select Average. Under Period, we're going to change that to 1 minute because we want our alarm to be triggered if our CPU is exceeding the threshold for 1 minute. So scroll down. The threshold type is going to be a static value, and the value is going to be whenever CPU utilization is greater or equal to, and we define the threshold down here. And I'm going to define the threshold as 90%. There's some additional configuration here. So we can define the number of data points within the evaluation period that must be breaching to cause our alarm, to go into the alarm state, and I'm just going to keep it to 1 out of 1. And you can also define how we treat missing data, and I'm going to keep that to the default. So hit Next. And then on the next screen, we can configure the actions that we want CloudWatch to take. So we can define the alarm state that's going to trigger this action, and it's going to be when the metric or expression is outside of our defined threshold and our alarm is in the alarm state. Down here, this is where we can configure an SNS topic to send us an email notification. And this is actually integrated within the CloudWatch console, so it makes it really easy to set up. So I'm going to select Create new topic, and it's going to create a new topic for us. Down here, we can add our email address so that we receive the email notification. So in here, just type your email address and then select Create topic. In this section, you can add an auto scaling action. So you can configure an auto scaling group to trigger based on this alarm, and you can also add an EC2 action as well. So you could stop the instance, terminate, or reboot, for example. And you can also add a Systems Manager action. So you can create an ops item or an incident within Systems Manager to notify to your support team that something has gone wrong, but that is really out of scope for the SysOps Administrator exam. So if you're happy with those settings, hit Next. And we'll give our alarm a name. Hit Next and scroll down to the bottom and Create alarm. So that is our alarm created. And before we try and trigger our alarm, we'll need to go into our email and accept the SNS subscription. Otherwise, we won't receive an email. So if you head to your email, you should have received an email like this, letting you know that you've chosen to subscribe to the topic, Default_CloudWatch_Alarms_Topic. And to confirm this subscription, you will need to click this link. So click the link, and you'll get a notification like this, which lets you know your subscription is confirmed. So now that we've confirmed our SNS subscription, we are ready to have some fun and max out the CPU on our instance. So come back to the EC2 console. I'm just going to refresh my dashboard. Our system should be up and running with our two status checks passed. I'll select my instance and hit Connect. And we'll use EC2 Instance Connect. So select Connect, and now we're just going to use a little script, which is going to max out our CPU. So type while true colon do echo colon done and hit Enter. And this is just going to put our system into an infinite loop, and it's going to max out our CPU. So now, let's head back to the CloudWatch console, and we'll need to wait a minute or so for the metrics to appear in CloudWatch. And hopefully, in a few moments, we should see our alarm change from the OK state to an alarm. So just be patient, have a cup of tea or coffee, and in a few minutes, we should be good to go. And there we go. Our alarm has been triggered. The state has changed to In alarm. And if it seems to be taking a little too long for you, you can just hit this Refresh button, and it will update the status. So now, the moment of truth.

We can go and check our email and see if SNS has sent us a notification, letting us know that the CloudWatch alarm has been triggered. So if you head to your email, you should see an email like this, letting you know that your CPU utilization is over 90%. And if you open the email, it will give you all the details of your alarm, including the name and description of the alarm, the metric that is being monitored, and your instance ID as well. And if you haven't received an email, it could be because you didn't accept the SNS subscription, so do make sure you accept the subscription. So, onto my exam tips. And just remember that CloudWatch alarms allows you to set an alarm to notify you when a threshold has been hit. You can create an alarm for any metric that you choose, for example CPUUtilization. And you just saw how easy it is to set up email notifications. So as well as showing you an alarm in the console, CloudWatch can also send you email notifications using an SNS topic. So that's it for this lecture. Once you've finished exploring CloudWatch alarms, please remember to delete your EC2 instance if you're using your own AWS account. If you have any questions, please let me know. Otherwise, I will see you in the next lecture. Thank you.

Introduction to CloudTrail

Hello, Cloud Gurus, and welcome to this lecture which is going to introduce Cloudtrail. And we'll begin with what is CloudTrail? We'll take a look at a Cloudtrail example. What gets logged in CloudTrail? We'll review some CloudTrail use cases. We'll take a look at keeping logs longer than 90 days. We'll explore what we mean by near real-time in relation to CloudTrail logging. And we'll take a look at my exam tips, as well. So what is Cloudtrail? Well, Cloudtrail is a service which records user activity in your AWS account. It is enabled by default when you first create your AWS account, and it records events related to creation, modification, or deletion of AWS resources. For instance, the creation of identity and access management users, modification of settings on your S3 buckets, and deletion of EC2 instances, for example. So let's take a look at a CloudTrail example, and CloudTrail logs the API calls that are made in your AWS account and the majority of AWS services are supported. So whether you're using the AWS console or the command line interface, everything that you do is going to be logged in CloudTrail. For example, you could create an RDS database, change your S3 settings, or restart an EC2 instance, and it will all be logged in Cloudtrail. However, let's say you ran an SSH or RDP session from your local machine to an EC2 instance. Well, CloudTrail is not going to log that because you would be communicating directly with the operating system of your instance, and you're not using the console or AWS CLI. Now, when we say CloudTrail supports the majority of services, it really supports almost all AWS services. And if you head to the AWS documentation for CloudTrail-- I've included a link to this website in the resources for this course-- you can review all the different AWS services that are supported for CloudTrail. And if we head over here, you can actually see the unsupported services. And it's really only a handful of services that are not supported by CloudTrail the moment. And you might be wondering: What does CloudTrail log? Well, here's an example. And don't worry, you won't need to decipher something like this in the exam. I just want to show you an example and highlight the main points. So first of all, it logs who took the action, the date and time, what they did, and where they did it, and in this example, the action was to create a key pair and this was done in us-east-1. It also logs the source IP that the request came from, any request parameters, and in this case, a key name was provided, and the response from AWS. And as it was a request to create an SSH key pair, the response includes the key material of the key pair. So you really get a lot of information in the CloudTrail logs. So let's take a look at some use cases for CloudTrail. Well, CloudTrail is great for after-the-fact investigation of incidents in your AWS account. It's also a great tool to use for near

real-time security analysis of user activity. And it can also be used to help you meet your industry regulatory compliance and audit requirements. Now by default, CloudTrail keeps 90 days worth of event history, but if you need longer than 90 days then you can create your own trail. And when you create a trail in the console, the logs are saved indefinitely to an S3 bucket, but do remember, that you will be charged for storing files in S3. Now CloudTrail is secure by default and when the logs are stored in S3, they are encrypted using server side encryption. And when you create a trail, log integrity validation is also enabled by default. And this means that the logs are digitally signed so that you can detect if a log was ever changed or deleted by somebody trying to cover their tracks. And by default, a trail created in the console will apply to all regions. Now CloudTrail provides near real-time logging, but what does that actually mean? Well, after making an API call, it can take up to 15 minutes for the call to appear in CloudTrail. And it can then take a further 5 minutes for CloudTrail to publish the logs to S3. So it's publishing logs approximately every 5 minutes. So it's not instant. And it can take up to 15 or 20 minutes for an event to appear in the CloudTrail log. So onto my exam tips. And just remember that CloudTrail logs all of the API calls that are made within your AWS account, and that could be using the AWS CLI or using the console as well. And it provides an audit trail of activity on your account. Recording the who, when, what, and where source IP of the request, request parameters, and the response from AWS. It's great for incident investigation, near real-time security analysis of user activity, and compliance. You get 90 days history by default, but if you need greater than 90 days history, then create your own trail and that will store your CloudTrail logs indefinitely in an S3 bucket. So that's it for this lecture. If you have any questions, please let me know. Otherwise, I'll see you in the next lecture. Thank you.

Demo: Working with CloudTrail

Hello Cloud Gurus and welcome to this lecture where we'll get some hands-on experience working with CloudTrail. And we'll begin by reviewing the past 90 days' event history using the CloudTrail console. Next, we'll create a trail to store encrypted CloudTrail logs indefinitely in a new S3 bucket. And after a few minutes, we should be able to see the CloudTrail logs appear in S3. So if you're ready to get your hands dirty with CloudTrail, please join me in the AWS console. So here I am in the console, and I'll head to Services. And you'll find CloudTrail under Management & Governance. So select CloudTrail. We'll select our event history on the left. And here is our event history. So we can view all the API calls over the last 90 days, and you can filter by resource name, resource type, or username. So I'm going filter by my username. And you can even select an event and download it as CSV or JSON. But these are only the events from the last 90 days. So what if you need to keep your events for longer than 90 days? Well, that is where you will need to create a trail. So head over to Trails on the left-hand side. We'll select Create trail. We'll give our trail a name. Scroll down. Under Storage location, we're going to create a new S3 bucket to store all of our logs for our trail. And its already provided a name for our trail log bucket. And you'll notice that server-side encryption is enabled by default, and you will need to provide either an existing KMS encryption key or create a new one, and we are going to create a new one. So we'll create a new KMS alias, and this is just an alias that it's going to use to reference our new KMS key. Scrolling down, you'll notice that log file validation is also already enabled, and this allows you to verify that log files were not modified or deleted after CloudTrail delivered them. Scrolling down, you can optionally configure CloudWatch Logs to monitor your trail logs and notify you if a specific activity occurs, but we're not going to do that right now. So scroll down to the bottom and click Next. Under log events, we are going to capture the management operations performed on our AWS resources, but

you can also log resource operations on or within a particular resource or use insight events, which allows you to identify unusual activity or behavior in your account. Scrolling down, we can select the activities that we want to log, and I'm just going to keep to the default of read and write. Hit Next. We can review our settings. And notice that, by default, it's creating a multi-regional trail. So it will log events for all AWS regions, and that is the recommended best practice So if you're happy with that, scroll down to the bottom and Create trail. So that is our trail created. So now, let's take a look at our logs. And here's our S3 bucket where the logs are going to be saved to, so select that. This is the Digest folder where CloudTrail will store a digest file once every hour, and this is used for the log file integrity validation. But it's this CloudTrail folder where the logs themselves are stored. And it may just take a few minutes to populate. And after a few moments, you should see a folder appear. So if you select that and the logs are organized by date, so select the folder for today's date. And here are our logs. So if we select one of our logs and select Open, and it's all in JSON format, so it can be pretty difficult to read. But at the top here is my user identity. Here's the event time, the event source, which is the security token service from AWS. We've got to a source IP address down here, a user agent, which is the AWS SDK. Down here, we've got some request parameters, and it's passing in a roleArn as a request parameter. And then down here where it says eventName, it's telling us that this event was an AssumeRole event. So, somebody has assumed a role in our AWS account. And then here is the region, and it's us-east-1. So I hope you can see that CloudTrail stores an awful lot of information about the API calls and activity that is happening in your AWS account. And once you've finished exploring, if you are working in your own AWS account, remember to head back to CloudTrail, and you'll want to go in and delete your trail. And remember to also delete your S3 bucket as well so that you don't get charged. So, onto my exam tips. And just remember that CloudTrail is enabled by default, and it includes 90 days of event history. But if you want to store greater, you'll need to create your own trail in order to store the data for more than 90 days. And by default, when you create a trail using the console, the data will be encrypted and stored in S3. So that's it for this lecture. If you have any questions, please let me know. Otherwise, I'll see you in the next lecture. Thank you.

AWS Config 101

Hello, Cloud Gurus, and welcome to this lecture, which is going to cover AWS Config. And we'll begin with what is AWS Config. We'll take a look at an example scenario, we'll review the AWS Config terminology and my exam tips as well. So what is AWS Config? Well, Config is all about configuration monitoring and it continuously monitors the configuration of your AWS resources in your account, and it evaluates the configurations against a desired state that you define. It can send notifications so it sends events into EventBridge and SNS if a resource deviates from the desired state. For example, a non-compliant resource can trigger an SNS notification. And one of the great things about AWS Config is that it can perform automatic remediation so it can automatically remediate non-compliant resources by triggering an action that you define. So here is the AWS Config dashboard from my account. And it actually discovers the AWS resources within my account and builds out a resource inventory that you can see here on the left-hand side. And the dashboard also shows compliance and non-compliance based on the rules that I have defined, so here it's identifying 3 non-compliant resources, which are not compliant with my rules, and then the non-compliant rules appear down here. So you can see the state of compliance at a glance. Now, AWS Config also stores your resource change history, and the change history for your resources is stored in an S3 bucket, and that means it's great for compliance and security governance. And

Config is also integrated with loads of different AWS services, so it's integrated with identity and access management, EC2, Elastic Block Store, Elastic Load Balancer, CloudFormation, CloudFront, CloudTrail, KMS, RDS, S3, Security Groups, SNS, and VPC to name just a few. So now, let's take a look at an example of what we can do with Config. Just imagine that your company requires that EC2 instances in your VPC must not have public IP addresses. AWS Config evaluates all EC2 instances and discovers that one of the instances is non-compliant. And the great thing about config is that it can go in and perform an automatic remediation action on your behalf. For example, it can go ahead and stop the instance. Let's take a look at some of the AWS Config terminology that you will need to be aware of. So, first of all, we have rules, and a Config rule represents the desired configuration for a specific resource. And AWS provides over 180 managed rules for predefined common best practices, and you can also create your own. And here are a few examples of the kind of rules that they provide. So there's a managed rule to check that S3 buckets have public read prohibited. There's one to check that your EC2 instances are all of a desired instance type. You can check that CloudTrail encryption is enabled and it hasn't been disabled, and that EBS volumes are configured to be encrypted by default. And these are just a few examples. And they also provide something called conformance packs, and a conformance pack is a set of rules and remediation actions that can be deployed and managed as one. And they give you loads of different sample templates that you can use. For example, operational best practices for S3, for EC2, and for identity and access management. There's also best practices for PCI DSS and for the AWS Well-Architected Framework Security Pillar. So these ready-made sets of rules to help you to use config to evaluate your own environment against these best practices. And just remember that AWS Config is all about configuration monitoring, and it continuously monitors the configuration of your AWS resources for compliance with a desired state that you define. It provides a dashboard with an inventory, and shows compliance and non-compliance at a glance. Config uses rules to define the desired state of your resource configuration. For example, there's a managed rule that will check if your S3 buckets have public read prohibited. There's one to check that EC2 instances are of the desired instance type. You can check that CloudTrail encryption is enabled and it hasn't been disabled, and that EBS volumes are configured to be encrypted by default. And these are just a few examples. And a conformance pack is just a set of rules managed as one, and they provide loads of different conformance packs. For example, operational best practices for S3, for EC2, or for identity and access management. And finally, with AWS Config, you can perform automatic remediation so you can remediate non-compliant resources by triggering an action that you define. For example, you can stop or terminate a non-compliant instance. So if that's it for this lecture. If you have any questions, please let me know. Otherwise, I will see you in the next lecture. Thank you.

Demo: Using AWS Config

Hello Cloud Gurus and welcome to this lecture where we're going to be using AWS Config. And we'll begin by creating an S3 bucket, and we're going to accept all of the default settings. Next, we'll create an AWS Config rule. So we'll create a rule in Config, which checks that server-side encryption is enabled on all S3 buckets. And we'll use a Config-managed rule named s3-bucket-server-side-encryption-enabled. And if everything is working, after the rule is evaluated, then Config should report that the S3 bucket we created is non-compliant. So if you'd like to join me in the AWS console, we'll get started. So here I am in the console, and we're going to head to Services, and you'll find S3 under the Storage section. So select S3, select Create bucket, and we're going to create a bucket in the Northern Virginia region, and we need to give our bucket a globally

unique name. And then I'm going to add my name on the end. And then to make sure it's completely unique, I'll just add some random numbers on the end. So now, we'll accept all the rest of the defaults scrolling down, including the default encryption settings. So we are not going to enable server-side encryption on our bucket. So scroll down to the bottom and select Create bucket. So that is our bucket created, and there it is. And now we are ready to create our Config rule. So come to Services, and you'll find Config under Management & Governance. So select Config. You should be in the us-east-1 Northern Virginia region. And if you haven't used AWS Config before, you will need to set it up for the first time, and we'll just use this one-click setup. So select that, scroll down, and if you remember, Config tracks the changes to the configuration of your AWS resources, and it records the configuration history for each resource. And this configuration history is stored in an S3 bucket, so it actually creates an S3 bucket for us when it is first initializing Config. So if you're happy with that, just click Confirm. And it can sometimes just take a few moments to get everything ready. And there is our Config dashboard. And at the moment, we don't have any rules configured. So come over to Rules on the left-hand side, select Add rule, and we're going to add an AWS-managed rule. And if you remember, we use rules to define the desired configuration settings of our AWS resources. And if you scroll down here, you'll see that there are loads of different AWS-managed rules. Now we are going to configure a rule, which checks if all of our S3 buckets have server-side encryption enabled, and we can search for that rule in this search box. So I'm just going to search for server-side encryption, and it's identified the rule I'm looking for. So we want the s3-bucket-server-side-encryption-enabled rule, and this is going to check that our S3 bucket either has default encryption enabled or that it has a bucket policy explicitly denying put requests without server-side encryption. So select the rule. Click Next. On this page, you can customize any of the fields, and it's already created a unique name for our rule and added a description. Scrolling down, we can define the scope of changes and choose when evaluations will occur. And I'm just going to stick to the default. So the rule is going to be triggered when any resource that matches the specified type or the type plus identifier is created, changed, or deleted. Scrolling down, we can define the resource category, and it's already selected S3 bucket. So we'll stick with that. We don't need to define any parameters. So just come down to the bottom and click Next. We can review everything on this page. And if you're happy, just click Add rule. So that is our rule created. And as we already created our S3 bucket, we'll need to manually re-evaluate the rule. So select the rule, come to Actions, and select Re-evaluate. And it does just take a minute or so to evaluate the rule, so be patient. And when it's completed, you'll see the result in the Compliance field on the right. And if you're impatient like me, then you can refresh your browser. And there we go. After refreshing my browser, we can see that I've got two noncompliant resources. So let's head over to my dashboard. And here, on the dashboard, I can see my resource inventory. So these are all the resources in my account that Config is aware of. Over here, under Compliance status, it's showing that I've got one noncompliant rule, and I've actually got two noncompliant resources. So let me select the noncompliant resources and see what it's found. And sometimes with Config, you do get this error saying it's currently experiencing unusually high traffic. And this is something that I've noticed before with Config, so just be patient. And sometimes you will need to try your request a couple of times before Config responds. And there we go. The second time I tried, it's showing me that I've got two S3 buckets that are noncompliant. And you might be surprised to see the that the first one on the list is actually the S3 bucket that Config created itself. And I don't know about you, but I think it's pretty funny that Config has identified its own bucket as noncompliant. So when it created this bucket, it didn't configure server-side encryption by default, and that is just one of the settings within Config. And then, it should be no surprise that the bucket that I created in the beginning has

also been identified as noncompliant because we didn't enable server-side encryption. So, onto my exam tips for AWS Config. And just remember that Config continuously monitors the configuration of your AWS resources, and it's monitoring your resources for compliance with a desired state that you define. We use rules to define the desired configuration state of our resources, and Config also provides a dashboard inventory. So it provides an inventory, and it also shows compliance and noncompliance with our rules. So that's it for AWS Config. And if you are working in your own AWS account, then you will need to disable something in Config to avoid getting an unexpected bill. So if you are in your own AWS account, head to Settings, select Edit, and just disable recording because it's this configuration recorder which is going to cost you money. And then you'll also want to go back to S3 and delete your S3 buckets as well. So that's it for this lecture. If you have any questions, please let me know. Otherwise, I'll see you in the next lecture. Thank you.

Remediation Using AWS Systems Manager and AWS Config

Hello Cloud Gurus and welcome to this lecture, which is going to cover remediation using AWS Systems Manager and AWS Config. And we'll begin with a recap of Systems Manager. We'll take a look at some of the Systems Manager predefined actions. We'll explore the relationship between Config and Systems Manager. We'll take a look at an example scenario and consider some of the other useful remediation actions as well. And we'll finish off with my exam tips. Now if you remember, Systems Manager is a management tool, which gives you visibility and control over your EC2 instances, and it allows you to perform common operational tasks on groups of instances simultaneously without having to log into each one. So you can use it to automate loads of different boring operational tasks, for example patching and installing applications, running scripts, stopping and starting EC2 instances, and running AWS API calls on multiple instances simultaneously. And with Systems Manager, you get loads of different predefined actions that can be used to perform operational tasks. For example, for EC2, you can resize an instance, release an elastic IP address, stop or terminate an instance. When it comes to EBS volumes, you can attach an EBS volume, delete an unused volume, enable encryption by default, or even modify an EBS volume type. For security groups, you might want to delete an unused security group, disable incoming SSH, or disable public access for the security group. And Systems Manager can do loads of other cool things as well like configure CloudTrail logging if its been disabled, create a snapshot of your RDS database, and even publish an SNS notification. And these are just a few examples of the kind of operational tasks that can be automated using Systems Manager, and any of these predefined actions can be leveraged by AWS Config and used to remediate noncompliant resources. Now if you remember, Config is a service that monitors the configuration of resources in your AWS account, and it's monitoring the configuration for compliance with rules that you configure. And Config can also remediate noncompliant resources. So as well as identifying them as noncompliant, it can go in and take an action on your behalf to remediate the situation. And how does it do that? Well, it utilizes Systems Manager to perform an automated remediation action that you define. So imagine that the company that you work at only supports EC2 instances that are running Amazon Linux 2, and you have been asked to terminate any EC2 instances that are running a different operating system. Somebody has launched an EC2 instance, and it's running Red Hat. Very naughty. But luckily, you've configured a role in AWS Config, which will check that all EC2 instances have been launched using the Amazon Linux 2 AMI. So Config will mark the offending instance as noncompliant, and we can also configure an automatic remediation action within Config. And under the hood, of course, we are using Systems Manager to perform the remediation. And we can use a

predefined action to terminate a noncompliant instance, like AWS-TerminateEC2Instance. And that will go in and terminate our noncompliant instance. So let's take a look at some of the other useful remediation actions that we can use, and all of these automatic remediation actions can easily be configured from inside the AWS Config console. But of course, under the hood, it's all being done by Systems Manager. So we can publish an SNS notification if a resource becomes noncompliant. We could delete unused resources, for example unused EBS volumes, unused elastic IP addresses, or security groups, etc. We can enable encryption on an S3 bucket, or we could disable public access for a security group. And these are just a few examples of remediation actions that we can use in response to findings in AWS Config. So for my exam tips, the main thing to remember is that AWS Config integrates really well with Systems Manager. Noncompliant resources can be automatically remediated using actions defined in Systems Manager. For example, we could stop or terminate noncompliant EC2 instances, publish an SNS notification to send us an email about a noncompliant instance, delete unused resources, enable S3 encryption, or disable public access for a security group. So that's it for this lecture. If you have any questions, please let me know. Otherwise, I'll see you in the next lecture. Thank you.

Demo: Configuring Automatic Remediation Using AWS Systems Manager and AWS Config

Hello Cloud Gurus and welcome to this lesson where we'll be configuring automatic remediation using Systems Manager and AWS Config. We'll begin by launching two EC2 instances. One will be Amazon Linux, and the other one will be Red Hat. Next, we'll create an AWS Config rule, and we're going to create a rule that checks that all EC2 instances are running Amazon Linux. After the rule is evaluated, Config should report that we have one instance that is compliant and one that is noncompliant. After that, we're going to configure automatic remediation. So we'll configure AWS Config to automatically stop the noncompliant instance. And under the hood, it is using AWS Systems Manager. Now for everything to work, we need to configure an Identity and Access Management role, and that role is going to be called MyAutomationRole. We'll associate a managed policy named AmazonSSMAutomationRole, and this policy contains the permissions that are needed to stop an EC2 instance. We'll also attach what is known as an inline policy, and this is going to allow the role to be passed to another service. And don't worry too much about the details here because we don't need to know them for the exam. And then finally, we're going to create a trust policy, which will allow Systems Manager and EC2 to assume our role. And this simply enables EC2 to register with Systems Manager, and it allows Systems Manager to stop the EC2 instance. And don't worry, you will not need to learn any of this for the exam. So we are going to configure it using a script to save time. So if you are ready to get started with automatic remediation using Systems Manager and AWS Config, then I will see you in the console. So from the console, search for EC2 and Launch instance. Call it SSM Amazon Linux. We'll use the Amazon Linux AMI. Instance type is t3.micro. Proceed without a key pair and Launch instance. So that's our first instance, and now we need to launch the second one. So come to Instances, Launch instance, call it SSM RHEL for Red Hat Linux. Scroll down and under operating system, select Red Hat, so that's Red Hat Enterprise Linux. Instance type is t3.micro, proceed without a key pair, and Launch instance. So now, select Instances. There's our two instances, and they're still initializing. But before we go any further, I'm going to make a note of the Amazon Linux 2 AMI that was used to launch our first instance. So select your Amazon Linux instance, scroll down to the instance details, and here's your AMI ID. I'm going to copy that to my clipboard, and I'm going to paste it into a text

editor so that I've got it to hand when I come to create my AWS Config rule. Next, we're going to configure our IAM role. And to make it easy, we're going to do it using a script. And you will find the URL for this script in the resources for this lesson, and here it is. So let's quickly go through the script before we run it. Now first of all, I've just got some echo statements just commenting on what we're doing, but this is the first real command. So first of all, we're creating the Identity and Access Management role and configuring a trust policy, allowing EC2 and Systems Manager to assume the role, and that's this first command. In the next command, we attach the AmazonSSMAutomationRole AWS-managed policy. In the next command, we add the inline policy, which will allow the role to be passed to another service. And then right at the end, the script is going to output the ARN of the role we just created so that we can use it. But don't worry. You don't need to understand or remember any of this for the exam. I'm just explaining it in case you're interested. So how are we going to run this script? Well, the easiest way is to use the AWS Cloud Shell. So from the AWS console, click on the Cloud Shell symbol up here. I'm going to close down the welcome screen. And when your cloud shell is ready, just type this command curl -o, then the name of our script, and then the URL to our role file in GitHub. And this is the URL that you are going to find in the resources for this lesson. So I'm going to copy that URL, paste it in here. This is what your command should look like. And hit Enter. Now type ls, and there is our file. So that command has just copied the file from our GitHub repository, and it's saved it as roleconfig.sh. But before we run our script, we need to change the permissions. So type chmod u+x to give execute permissions and the name of our script and hit Enter. So now we're ready to run the script. Type ./ the name of our script and hit Enter. And then, if everything has been successful, at the end of the output of the script, you will find the ARN of the role that we just created, and there it is. So I just want to copy that ARN, and I'm going to paste it into the same text editor where I've saved my AMI. And there we go. We are now good to continue using AWS Config. So back in the AWS console, search for Config. And if you haven't used Config before in your account, then you will need to select the 1-click setup. So select that and confirm, and this is just setting up Config for the very first time. Once it's completed, it will take you to this dashboard, and you want to select Rules on the left-hand side under Dashboard. Select Add rule, and we're going to add an AWS-managed rule. Under Rules, we're going to search for a rule, and I want you to search for AMI. So type AMI, and this is the one we're looking for, approved-amis-by-id. So it checks whether running instances are using specified AMIs, so select that rule. And in this case, we're going to specify the Amazon Linux AMI because that is the AMI that we want to use. So now I'll hit Next. On this screen, we can customize any of the fields, and it's already given as a name for our rule. Scroll down. Under Scope of changes, we'll keep the scope of changes to the default. So when any resource that matches the specified type is created, changed, or deleted, then it's going to trigger this rule. Under Resource category, it's going be EC2 instance. And under Parameters, this is where we want to specify the AMI ID that we're looking for. So in here, I'm going to paste my AMI ID. So Config is going check that all instances are using this AMI. So now we can just scroll down and hit Next. We can review everything and then Add rule. So that is our rule added. And if we want Config to evaluate the rule for an existing resource, we can just select our rule, come to Actions, and select Re-evaluate. And the results of the evaluation will appear down here under Detective compliance. And sometimes the screen doesn't refresh really quickly, so just use the Refresh button over here and it will refresh. And there we go. It is showing us that we've got one noncompliant resource. So let's take a look at our dashboard. Select Dashboard on the left, and it's showing us here that we've got one noncompliant rule. And on my screen, the number of resources that are noncompliant hasn't quite caught up yet, so it's still saying 0 noncompliant resources. So let's select our rule. Here's the noncompliant rule,

and we've got one noncompliant resource. I'm going to select my rule and then select Actions, and then this is where we can manage our remediation. So select Manage remediation, and we can configure Config to automatically remediate our instance that is noncompliant. Under Remediation action, we're going to select an automatic remediation method. If a resource is still noncompliant after the automatic remediation, you can set the rule to try again or just keep it to the defaults. Down here, this is where you can select the remediation action, and it's reminding here that the execution of the remediation action is done using Systems Manager Automation. If you select the drop-down, you can take a look at all the different remediation actions that are available. But I'm going to search for EC2, and these are all the actions that are associated with an EC2 instance. So you can do things like restart it, start it, stop it, terminate it, etc., but we are going to use Config to stop any EC2 instance that was created using the wrong AMI. So select StopEC2Instance. Scrolling down to Resource ID parameter, that's going be InstanceId, and that's going to populate this field down here. And then under AutomationAssumeRole, this is where we need to provide the ARN of the role that we created in the beginning. So hopefully you've made a note of it. Here's mine. And paste the ARN in there. Save changes, and that has updated our remediation action. So now, if we scroll down to the bottom, select our noncompliant resource and select Remediate, just be patient because it does take a few moments, and we can follow the status down here. If it feels like it's taking too long, just use the Refresh button, and there we go. Action executed successfully. So now, let's head over to our EC2 dashboard and see if our instance has been stopped properly. So come to EC2, select Instances, and you should find that the Red Hat Enterprise Linux instance is the one that stopped, and your Amazon Linux instance is still running. So that is automatic remediation with Config. And for my exam tips. Just remember that we can use AWS Config to continuously monitor the configuration of our AWS resources. Rules are evaluated by Config to determine noncompliance. For example, you can create rules to check for overly permissive security groups, EBS volumes that are not encrypted, or like we did just now, instances created using the wrong AMI. And we can configure automatic remediation because Config integrates really well with Systems Manager to remediate the resources that were found to be noncompliant. For example, we can stop or terminate an EC2 instance that has been labeled as noncompliant. So that is it for this lesson. Any questions, please let me know. Otherwise, I will see you in the next lesson. Thank you.

What Is EventBridge?

Hello, Cloud Gurus, and welcome to this lecture, which is going to cover EventBridge. And we'll begin witt what is EventBridge. We'll take a look at scheduled events and it might all sound very similar to CloudWatch events. We'll take a look at some EventBridge use cases and my exam tips as well. So what is EventBridge? Well, EventBridge is all about event-driven architectures and an event is a change in state. So imagine services like AWS Config, CloudTrail, and CloudWatch, all generating events relating to various changes in state in your AWS account. And these events can be sent to EventBridge and within EventBridge, we can configure rules, which match the events and route them to the correct target, and targets define an action that will be taken on an associated AWS service. For example, shutting down an EC2 instance that has been marked as non-compliant by AWS Config or triggering a Lambda function to take some action on your behalf in response to an event or sending an SNS notification to notify you of an event in CloudTrail or CloudWatch. Now EventBridge can also be used to schedule events. So we can create EventBridge rules, which run on a schedule that we define. For example, once an hour or once a day or using a cron expression we can set a rule to run at the same time on a specified day, week, or month. Say for example, you

wanted to reboot a particular instance every month at the same time. You can do that using a scheduled event in EventBridge. Now you might be thinking that this all sounds very similar to CloudWatch events, and you'd be correct in thinking that. Now, according to AWS, EventBridge is the preferred way to manage your events and CloudWatch events and EventBridge are actually using the same underlying service and API. However, EventBridge provides more features. So that's why they say it's the preferred way to manage your events. And any changes that you make in either the CloudWatch or EventBridge console are going to appear in both consoles and that is according to the AWS documentation. So let's take a look at some example use cases. Let's say that your company requires that all EC2 instances must have encrypted disks volumes and somebody has created a new EC2 instance without encrypting the attached EBS volumes. And it gets detected by Config and marked as non-compliant. An event will be generated and sent to EventBridge, which triggers a rule which invokes an action to send you an email using SNS. So you will be able to find out if somebody has created a non-compliant EC2 instance. What about if CloudWatch detects that one of your EC2 instances is showing CPU utilization of 99%? An event will be generated and sent to EventBridge, which triggers a rule to invoke an action to send you an email using SNS. And this is all really easy to configure because CloudWatch events and EventBridge and SNS are all really well integrated. And if this all sounds a little bit familiar, then remember that EventBridge is using the same underlying technology as CloudWatch events. So onto my exam tips. And just remember that EventBridge allows you to really easily configure event-driven systems and define tasks that can be run on a predefined schedule. And it's using the same underlying technology as CloudWatch events. Events are state changes generated by services like AWS Config, CloudWatch, and CloudTrail. Rules can be configured to match the events and route them to the correct target and targets can be services like Lambda, SNS, or EC2. And they respond to the event by taking some predefined action. So that's it for this lecture and you are going to be getting your hands dirty with EventBridge quite a lot in this section of the course. So if you have any questions, please let me know, otherwise I will see you in the next lecture. Thank you.

Demo: Using Amazon EventBridge

Hello Cloud Gurus and welcome to this lesson where we'll be getting our hands dirty using Amazon EventBridge. And we'll begin by launching an EC2 Instance. Next, we'll create an SNS topic, and we'll subscribe to the topic using our email address. And we need to remember to confirm the subscription as well so that we can receive the SNS notifications. Next, we'll create an EventBridge rule to notify us of any EC2 state changes using SNS. And then finally, we'll be changing the state of our EC2 instance by stopping the instance. So we'll go in and stop the instance, and then we can check if we've received an email notification from EventBridge. And the notification is going to come from SNS. So if you are ready to play with EventBridge, then please join me in the AWS console. So from the AWS console, we're going to start by searching for EC2, and we're going to launch a new instance. So select Launch instance. The name will be EventBridge Demo. We'll use Amazon Linux. And instance type will be t3.micro. We can proceed without a key pair and then go ahead and Launch instance. Next, let's create an SNS topic. So search for SNS. I'll open it in a new browser tab. In the left-hand menu, select Topics, Create topic. It's going to be a standard topic because we're going to be using the email subscription protocol. So make sure that Standard is selected. Name it MyTopic with no spaces and then scroll down to the bottom and Create topic. So that is our topic created. But in order to receive our notifications, we need to subscribe to the topic. So under Subscriptions in the left-hand menu, Create subscription. Select your topic ARN from the

drop-down. Protocol will be email. Under Endpoint, this is where you need to give your email address. So just type your email address in here. And down here, it's just reminding us that after our subscription is created, we'll need to confirm it. So you'll need to confirm it within your email for everything to work. So for now, just scroll down and Create subscription. And now, let's head over to our email. So in your email, you should see a message like this from AWS Notification - Subscription Confirmation,. And if it's not in your inbox, then make sure you just check your spam folder just in case it ends up in there because it sometimes can do by accident. And within the email, it says to confirm this subscription. Just click the link below. So click on the link that says Confirm subscription. Once you've done that, you should get a message similar to this to say that you have successfully subscribed to the topic. So now, we are ready to go ahead and create our EventBridge rule. So back in the AWS console, we're going to search for EventBridge. Under Rules in the left-hand menu, the event bus is going be default and select Create rule. Call it EC2 State Change Rule. You can optionally add a description. We're using the default event bus. And select Rule with an event pattern. So our rule is going to be triggered by an event happening in AWS. And you'll also notice that You can use EventBridge to invoke your rules on a schedule that you define, and this is the option to use if you want to run a task on a specific day or time. But we are going to be using an event pattern instead, so just make sure that this rule type is selected. Now, hit Next. The event source is going be AWS events. Scrolling down, we're not going to use a sample event. So scroll down again. Under Creation method, select Use pattern form. Then under Event pattern, the event source is going be AWS services. The AWS service is going be EC2. So just search for EC2. Then under Event type, search for state. And we're going to select EC2 Instance State-change Notification. By default, it's set to any state change, but you could also select a specific state. And if you select that and select the drop-down, it will list all of the relevant EC2 states. We're not going to go with any of these. We're just going to select Any state. And this means that any time an instance changes state, so let's say it goes from pending to a running state or from running to shutting down or from shutting down to terminated, any state change like that is going to trigger our rule. Down here, this is where we can select specific instance IDs, but I'm just going to keep it to any instance. Hit Next, and this is where we can select our targets, and our target is going be an AWS service. So make sure that AWS service is selected here. And of course, we want to receive an SNS notification any time the rule is triggered, so SNS is going be our target. So search for SNS. Select SNS topic. And then down here in the drop-down list, you can select the topic that you created earlier. So now, we can hit Next, Next. We can review our options and Create rule. So that is our rule created, and now we're ready to have some fun changing the state of our EC2 instance. So if you come back to your EC2 console, come to our instances view, here's our instance. I'm going to select that and then select Instance state. And here, we can either stop, reboot, or terminate our instance. So I'm going to select Stop instance and confirm. We can see our instance state has changed to Stopping, so let's see if we've received an email. And if it's all worked correctly, you should firstly have received an email saying your subscription confirmation. And I've also received another couple of emails, so let's take a look at these. And this first one, it's telling us that we've hit that EC2 Instance State-change Notification rule. Here's the detail of our instance ID. And the state has changed to Stopping. And then there's also one further message just letting us know that the instance state has actually changed to stopped. And don't worry if you don't receive these email messages immediately. It can take just a few minutes sometimes for these messages to come through. So if they're not there immediately, just be patient. Give it a few minutes, and you should see them appearing. And then there's one last thing I wanted to show you. It's not related to the exam, but EventBridge is actually using the same underlying technology as CloudWatch Events. And if we search for CloudWatch up

here, select CloudWatch. In the left-hand menu, select Events and select Rules, there is our rule that we created using EventBridge. If you select the rule, here's the summary of our event pattern, and here's our SNS topic that it's associated with. So I think it's really interesting that you can see these rules in both parts of the AWS console. And if we head back to the Rules page, it's actually giving you this message here, saying that Amazon EventBridge, formerly known as CloudWatch Events, provides all functionality from CloudWatch Events. And it's actually letting you know that CloudWatch Events is now Amazon EventBridge. So if you do see anything in the exam and they might be talking about CloudWatch Events or they might be talking about EventBridge, just remember it's using the same underlying technology. But they're probably not going to ask you anything directly relating to CloudWatch Events now being EventBridge, but it's one to just keep in mind for background information. So onto my exam tips. And just remember that Amazon EventBridge receives events relating to state changes in AWS, for example if an EC2 instance changes state or a CloudWatch alarm changes state. You can use EventBridge to create rules that take actions based on the events it receives, for example sending you an SNS notification in response to a rule being triggered. And if you're already familiar with CloudWatch Events, then it's useful to know that EventBridge and CloudWatch Events are using the same underlying technology. So that is it for this lesson. If you have any questions, please let me know. Otherwise, I will see you in the next one. Thank you.

Demo: Scheduling Automated Tasks Using EventBridge and AWS Config

Hello Cloud Gurus and welcome to this lesson. And in this lesson, we are going to explore how to schedule or trigger automated tasks using EventBridge and AWS Config. And we'll begin by creating an AWS Config rule, and our rule is going check that all EC2 instances have been created without a public IP address. Next, we'll create an SNS topic, and we'll subscribe to our topic using our email address. We'll then configure an EventBridge rule, and this EventBridge rule is going to trigger an SNS notification whenever a Config rule's compliance change event is detected. So basically, it's going to send an SNS notification whenever an AWS resource is identified as being noncompliant by Config. And then finally, we're going to test our setup, so we'll create an EC2 instance with a public IP address. Config should mark it as being noncompliant. And if everything is working, then we should receive an email notification from our SNS topic, letting us know that we've got to a noncompliant resource. So if you're ready to get your hands dirty using EventBridge and AWS Config, then I will see you in the AWS console. So from the AWS console, the first thing we'll do is create our Config rule. So search for Config and select that. Make sure you're operating in the Northern Virginia region. And if you haven't used Config before, then select the 1-click setup. And then all you need to do is hit the Confirm button, and we'll be good to Continue. Next, select Rules from the left-hand menu. We'll add a rule. It's going to be an AWS-managed rule. Scroll down, and we're going to search for the rule that checks whether your EC2 instance has a public IP address or not. So search for no-public, and this is the one we're looking for, ecw-instance-no-public-ip. So select that one, scroll down to the bottom, and hit Next. If we want to, we can customize the name of our rule and the description, but I'll just stick with the defaults. Scrolling down to Evaluation mode, under Scope of changes, we'll stick with the default. So this rule is going to trigger when any resource that matches the specified type is created, changed, or deleted. Under Resources, we're going to stick with AWS EC2 Instance. We don't need to define any parameters, so hit Next. And then we can review all of our options and hit Add rule. So that is our rule created. The next thing we'll do is create our SNS topic. So search for SNS. Open the menu on

the left-hand side. Select Topics, Create topic, and we're going to create a standard topic because we want to be able to subscribe using email. And don't worry too much about the finer details of SNS. The main thing that you need to know for the exam is that it allows you to send notifications. And to send an email notification, we need to use a standard topic. We'll give our topic a name. I'm going to call it Non-Compliant-Resources and the same again for the display name. Then scroll down to the bottom and Create topic. Now we've created our topic, we need to subscribe our email address to this topic. So select Subscriptions on the left. Create subscription. Select your topic. Protocol will be email. And then under Endpoint, this is where you can add your email address. So this is the email address that's going to receive notifications from this SNS topic. So just type your own email address in here. And then, after your subscription is created, you will need to confirm it by checking your email and clicking the link in the email. So now scroll down. We're going to accept the rest of the defaults and Create subscription. Once that's been created, I'm going to head over to my email. And in your email account, you should see an email just like this with a subscription confirmation, letting you know that you've chosen to subscribe to this topic. And the name of the topic is Non-Compliant-Resources. And to confirm the subscription, you will need to click this link. So just click on Confirm subscription. And you should get a message like this telling you your subscription is confirmed, and this is going to allow the SNS topic to send email notifications to your email address. And once we're done with that, we are now ready to create our EventBridge rule. So in the search box, search for EventBridge and select Create rule. And I'm just going to call it ConfigRuleComplianceChanges. And this is going to be an EventBridge rule to send an email notification when a Config rule compliance change is detected. Scrolling down, select the default event bus. And we can either trigger our rule based on an event, or we can schedule it. But we're going to be using a predefined event pattern. So select Next. Event source is AWS event. We don't need to configure a sample event. So scroll down to Creation method, and the creation method will be to use a template provided by EventBridge to create an event pattern. So select Use pattern form. Under Event pattern, the event source is AWS, and the service is going to be Config. So select Config. Event type is going to be a Config Rules Compliance Change, and there it is. So just select that one, and this rule is going to trigger any time that Config generates a compliance change notification. We can accept all the rest of the defaults down here. Hit Next. Under Select targets, we'll select an SNS topic, so select SNS topic. And then down here, select our topic, Non-Compliant-Resources. Then hit Next, Next. We can review our options and create the role. So that is our EventBridge rule created, and now we're ready to test that everything is working. So search for EC2. Launch instance. The instance name is my-public-instance. Operating system is Amazon Linux. Instance type is t3.micro. We'll proceed without a key pair. Under Network settings, make sure that auto-assign public IP is set to Enable because we're deliberately trying to create an EC2 instance that is noncompliant. Once you've checked that, we are ready to launch the instance. Select your instance ID and wait for it to initialize. Once your instance has finished initializing, we can head back to Config. So search for Config. I'm going to open it up in a new tab. And if it's all worked, then Config should have detected that we have got at least one noncompliant resource, and there it is. If we click on our rule, here's the rule that's being flagged up that we've got one noncompliant resource, and the rule name is ec2-instance-no-public-ip. So Config has detected our noncompliant resource. And if we click on the rule, it will take us to the resources in scope, and here's our EC2 instance that it's identified, and we should also have received an AWS notification message in our email. So if I come back to my email inbox, here's my message from the Non-Compliant-Resources topic. Here's the name of my rule. The message type is a ComplianceChangeNotification. The resource type is my EC2 instance, and there's my resource ID.

ComplianceType is NON_COMPLIANT, and there's also this really useful annotation message here that lets us know that this EC2 instance uses a public IP. So that is how you can use EventBridge to trigger an automated task when AWS Config detects a compliance change. So we created an EventBridge rule, which received notifications from Config when it detected a noncompliant resource, and that triggered an SNS notification to send us an email and let us know that we had a noncompliant resource. Well, that's it for this lesson. If you have any questions, please let me know. Otherwise, I will see you in the next lesson. Thank you.

Demo: Exploring Health Dashboards

Hello Cloud Gurus and welcome to this lesson where we'll be exploring the health dashboard that is available with AWS. And we'll begin by exploring the AWS Health Dashboard, and this is a dashboard which allows you to view the status of all AWS services by region. But this just gives us a generic view of all the AWS services, including the ones you're not even using. But what about the services that you are actually using? Well, they've got to a section in the health dashboard that is dedicated to your own account, and this is where you can view issues with AWS services that are going to impact you based on the resources that you're actually using in your AWS account. And this is a great tool for understanding if there's a problem going on with an AWS service. And let's say you're troubleshooting an issue. This is a really good place to come and check if there's a problem within AWS and get the latest status. So if you're ready to get your hands dirty with AWS Health Dashboard, then let's get started. So from the console, search for health and select AWS Health Dashboard. Select Open and recent issues from the left. And then down here, this is where you'll see the current status of all the AWS services, and it's organized by region. And it basically provides a log of AWS service interruptions over the last 12 months. Now we can see at the moment there are no open and recent issues. And then over here, this is the service history. So you can actually go in and change the dates in here and take a look at some of the past service events. And we can see at the moment that there are no events. But if there does happen to be an issue with any of these services, it's going to appear down here. And let's just check some of the other regions as well and see if there are any issues that have happened in any of these other regions. Well, AWS is pretty reliable, and we haven't got any issues to show you. But if at any time when you're using AWS you notice that the service is not operating as expected, then you're going to find information about any related events and the status of the service on this page. So now, let's take a look at a more personalized view. Under your account health on the left, select Open and recent issues, and this is where you're going to find information relating to any issues that are going to affect the AWS resources that you're actually using. At the moment, we don't have any open issues, but this view is only showing the events from the past 24 hours. Under the Scheduled changes tab, this is going to show any upcoming planned events that might impact you. Other notifications is going to let you know about things like certificate rotations, billing notifications, and security vulnerabilities. And here, I've got one notification about SageMaker, and it's just letting me know that all new SageMaker notebook instances need to use Amazon Linux 2, and they're recommending that we migrate any workloads to Amazon Linux 2 instances. And then if I select the event log, this is where I can see any past events and their status. And you don't actually get 12 months' history in here. It looks like they are only giving us about 3 months' history. But the really cool thing about this view is that they will even identify which of your resources could be affected by this event. So in my account, it's identified this SageMaker notification, and this is the operational notification that we saw earlier relating to using Amazon Linux 2. And under the affected resources, it's letting us know

that we've got one affected resource. And here's the ARN of the affected resource. Pretty cool, eh? So for the exam, just be aware that the AWS Health Dashboard provides the status of all AWS services per region with a service history of the past 12 months. Under Your account health, it also provides information about issues with AWS services that you are actually using, and it's even going to identify the affected resources. So that is it for this lesson. If you have any questions, please let me know. Otherwise, I will see you in the next lesson. Thank you.

Section Review: Monitoring, Logging, and Remediation Summary - Part 1

Hello Cloud Gurus, and welcome to this lecture which is Part 1 of the Monitoring, Logging, and Remediation Summary, beginning with CloudWatch. And just remember that CloudWatch is all about monitoring the performance and health of your systems. By default EC2 sends host-level metrics like CPU, network, disk, and status check into CloudWatch. But you will need to use the CloudWatch agent for operating system-level metrics like memory usage on your instance, processes running on your instance and CPU idle time. And we can use Dashboards to create a custom view. For example, you can create a Dashboard which displays the metrics of all your production systems in the same place. Dashboards are multi-region so you can display the metrics for any region you like. Just select the region and add the widget that you want. So do remember to save it once you've configured everything onto CloudWatch logs. And just remember the CloudWatch logs terminology. So log events consist of an event message and a timestamp. For example, think of the kind of events that you might see in an Apache log or in var log messages on your Linux system. A log stream is a sequence of log events from the same source. For example, an Apache log coming from the same EC2 instance and each log stream belongs to a log group. And, of course, a log group is a group of log streams that share the same retention, monitoring, and access control settings. And you can manage all of the settings from within the log group. For example, think of multiple web servers running Apache, and they're all sending their logs into CloudWatch. And if you remember, in our demo, we configured an EC2 instance to send operating system level metrics and logs to CloudWatch. So first of all, we installed the CloudWatch agent, and that is the agent that runs on your EC2 instance. And it's responsible for sending the operating system level metrics and any logs that you define to CloudWatch. And when we talk about operating system level metrics, we mean things like CPU idle time, memory swap and disk usage. And those are the kind of metrics that are only visible from within the operating system. So we need to use the CloudWatch agent to collect those metrics and send them to CloudWatch. And we also use the CloudWatch agent to send our var log messages into CloudWatch, and you can send any log files you like, not just the system logs. You can also configure the CloudWatch agent to send application log files. For example, the var log HTTPD access log for a web server. Now, for the exam it is important that you understand how to configure metric filters and these allow you to filter for specific phrases in your logs. For example, warnings, errors, or HTDP status codes. They're really easy to set up and you configure them at the log group level. And you can configure metric filters for any of the log groups in your account. And an example use case is if you wanted to receive CloudWatch alerts for specific errors or warnings in your log files. We can use CloudWatch logs insights to interactively query and analyze data stored in our CloudWatch logs. We can query the logs directly and also generate visualizations like bar charts, line graphs, or pie charts, et cetera. For example, we can use CloudWatch logs insights to display the 25 most recently added log events. Search our VPC flow logs to find out which IP addresses are using a specific protocol or find the most expensive Lambda requests. And those are just a few examples of the kind of thing we

can do with CloudWatch logs insights. Now, you can also receive notifications with CloudWatch. For example, CloudWatch can be used to monitor your service quotas or limits and notify you if you are about to reach the limit. You can receive an SNS notification, for example, an email to your IT support team. And you can also view the alerts in the CloudWatch Dashboard and you'll find the alerts in the alarm section. And a great use case is say you've reached 90% of the quota value or the service limit for on-demand EC2 instances. You can use CloudWatch to monitor the quota and send an email to your IT support team when you reach 90%. And, of course, we can also receive notifications relating to AWS health events as well. So the AWS Health API provides information about the changes of health in AWS resources, and it can send health events into Event Bridge, formerly known as CloudWatch Events, which in turn will trigger a CloudWatch alarm to send an SNS notification, send a message into SQS, or trigger a Lambda function. And AWS Health is the same API that is used by the Personal Health Dashboard. And, finally, you can also use CloudWatch to create alarms and you can use an alarm to notify you when a threshold is hit. And that's exactly what we did in the demo. So you can create an alarm for any metric. And in the demo we used the CPU utilization metric. So we created an alarm to trigger if our CPU utilization hit 90% or greater for 5 minutes on our EC2 instance. And then when the alarm triggered, CloudWatch sent us an email notification using an SNS topic. So that's it for Part 1 of this lecture. And if you're ready to continue to Part 2, I'll see you in the next lecture. Thank you.

Section Review: Monitoring, Logging, and Remediation Summary - Part 2

Hello, Cloud Gurus, and welcome to part 2 of the Monitoring, Logging, and Remediation Summary. And we'll continue with CloudTrail. And just remember that CloudTrail logs all of the API calls that are made in your AWS account. So it's a really good audit trail of everything that is going on. And for each request it's monitoring who, when, what, and where the source IP for the request, any request parameters, and the response from AWS as well. And it's great for incident investigation, near real-time security analysis of user activity, and compliance. You get 90 days history by default, but you can create your own trail in the console to store your CloudTrail logs indefinitely in S3. And when you create a trail in the console, encryption will be enabled by default. AWS Config continuously monitors the configuration of your AWS resources for compliance with a desired state that you define. And it provides a dashboard which gives you an inventory, and shows compliance and non-compliance for your AWS resources. Rules define the desired state of your resource configuration. And there are loads of different managed rules that you can choose from. For example, you can check that public read is prohibited for your S3 buckets. You can check that your EC2 instances are of a desired instance type, check that CloudTrail encryption is still enabled and it hasn't been disabled, and check that EBS volumes are set to be encrypted by default. Conformance packs are a set of rules that are managed as one. And AWS provides lots of different conformance packs for things like best practices for S3, EC2, and identity and access management, etc. And one of the very cool things about Config is that it can perform automatic remediation. So it can remediate your non-compliant resources by triggering an action that you define. For example, stop or terminate a non-compliant instance. And, of course, it's doing this using Systems Manager. So AWS Config integrates really well with Systems Manager allowing you to remediate non-compliant resources. For example, stopping or terminating EC2 instances, publishing an SNS notification, deleting unused resources, enabling S3 encryption, or disabling public access for a security group to name just a few examples. So the way it works is that Config is continuously monitoring the configuration of your AWS resources. It's using its rules to evaluate and determine non-compliance.

For example, overly permissive security groups, EBS volumes that are not encrypted, or instances that are using the wrong AMI. And then, in order to perform automatic remediation, it integrates with Systems Manager to remediate those resources that have been found to be non-compliant. For example, it can go in and stop or terminate an EC2 instance which has been marked as non-compliant. Onto EventBridge. EventBridge allows you to easily configure event-driven systems and define tasks that can be run on a pre-redefined schedule. And it's using the same underlying technology as CloudWatch Events. And if you remember, events are state changes generated by services like AWS Config, CloudWatch, or CloudTrail, etc. Rules match events and route them to the correct target, and targets can be services like Lambda, SNS or EC2, and they respond to the event by taking some action on your behalf. And EventBridge can be configured to trigger an automated task when AWS Config detects a compliance change. So if Config detects that an EC2 instance is non-compliant, this can trigger EventBridge to send an SNS notification in the form of an email to your IT support team. And then, finally, we have Health Dashboards. So we've got the Service Health Dashboard, which provides the status of all AWS services per region. And you can use this dashboard to find out if a service is operating normally or if there's an issue with that service. And then, there's also the Personal Health Dashboard. And this provides information about issues with AWS services that you are actually using in your account. And if there is an issue with one of the services that you're using, it will appear down here under Open Issues. So that is it for this section. If you have any questions, please let me know. Otherwise, we will see you in the next section. Thank you.

Storage and Data Management

Section Introduction

Hello Cloud Gurus and welcome to this section of the course, which is going to cover storage and data management. We'll cover S3 in S3 101, Introduction to Elastic File System. We'll also introduce Athena, Amazon OpenSearch. And we will also cover an Overview of Storage Gateway. So if you'd like to join me in the next lecture, we'll get started.

S3 101

Hello, Cloud Gurus, and welcome to this lecture, which is going to cover the fundamentals of S3. And we'll begin with what is S3? We'll take a look at some S3 basics, working with S3 buckets. We'll discuss how S3 is a key-value data store, a safe place to store your data. We'll take a look at S3 characteristics, how to secure your data, and my exam tips as well. So what is S3? Well, S3 stands for Simple Storage Service, and it provides secure, durable, and highly scalable object storage. It has a simple, easy to use web services interface and it is super scalable, allowing you to store and retrieve any amount of data from anywhere on the web at a very low cost. Now, S3 is object-based storage, so it manages data as objects rather then file systems or data blocks. You can upload any type of file you can think of to S3. For example, photos, videos, code, documents, or text files. But it cannot be used to run an operating system or database, and for that, you will need to use EBS volumes. Now with S3, you get unlimited storage, so the total volume of data and the number of objects that you can store is unlimited. So this means that you don't need to worry about allocating storage space or predicting how many TB you're going to need. Now, S3 objects can be up to 5 TB in size. So they can range in size from a minimum of 0 bytes to a maximum of 5 TB per object. And within S3, the files are stored in buckets, and a bucket is similar to a folder. And it's

really just a container, and it's the name that AWS uses for the location where you are storing your files. Now, when we come to work with S3 buckets, S3 actually has a universal namespace, and that means that all AWS accounts share the same S3 namespace, and each S3 bucket name must be globally unique. So it's actually similar to a DNS address, or an internet address, which must also be globally unique. So now, let's take a look at an example of an S3 URL, and this is how we can reference S3 resources. And it looks like this. So first of all, we have the bucket name, followed by the Region, and then finally, the key name. So that's the name of the object or the name of the file that you are storing in S3. And here is an example of a bucket name that I own. So the bucket name is fayecloudguru, the Region is us-east-1, and the name of an object that I've got stored in that bucket is Ralphie.jpg. And it's just a photo of my little sausage dog, Ralph. And then finally, when you upload a file into an S3 bucket, you are going to receive an HTTP 200 code if the upload was successful. And this isn't a code that you're going to see on your browser or on your webpage, if you're uploading something using the AWS Console. You will only see this code if you're uploading using the API or the Command Line Interface. But it's just good to know that this is a code that you're going to see if your upload was successful. Now, S3 is a key-value store, so what does that mean? Well, S3 objects consist of the following. First of all, we've got the key, which is simply the name of the object, and in my example, my object is called Ralphie.jpg. And then we have the value, and this is really the data itself, which is made up of a sequence of bytes which make up the object. There's also a version ID, which is important when we come to store multiple versions of the same object. And then there's metadata. And if you haven't heard that term before, it really just means data about data and when you upload a file into S3, you can add your own metadata about the data that you're storing. For example, content type, the date it was last modified, or you could even add the name of the team that owns the file, or the project that the file is related to. So S3 is a safe place to store your data files, and the data is spread across multiple devices and multiple facilities to ensure availability and durability. So Amazon could lose one of their devices or facilities, and the S3 service will still be available. Now, S3 is designed to be both highly available and highly durable, and when we talk about availability, that is all about the service being available when you need it. So S3 is built for 99.5 - 99.99 service availability, depending on the S3 tier that you select. And we'll review the different S3 tiers later on in this section of the course. S3 is also designed for durability, and it's designed for 11 9's durability for all data stored in S3, no matter which tier you select. And when we say durability, that is all about your data being stored safely and not getting lost or corrupted. For example, if you stored 10 million objects within S3, and you've got 11 9's durability, you could expect to incur a loss of a single object once every 10, 000 years. So that's what they mean when they say that S3 is a really safe place to store your data. Now, S3 offers a range of tiered storage classes designed for different use cases, depending on the type of data that you're storing, and your own business requirements. There's also lifecycle management, which allows you to define rules to automatically transition objects to a cheaper storage tier, or delete objects that are no longer required after a set period of time. And there's also versioning, and with versioning, all versions of an object are stored and can be retrieved, even including deleted objects. So we can keep multiple historical versions of the same file, allowing you to roll back to a previous version if you accidentally changed or deleted your file. On to securing your data in S3. First of all, we've got server-side encryption so you can set default encryption on your S3 bucket to encrypt all new objects when they're stored in the bucket. There's also access control lists, which allow you to define which AWS accounts or groups are granted access, and the type of access as well. And you can attach S3 ACLs to individual objects within a bucket for fine-grained access. And then finally, we could also secure our buckets using bucket policies, and these are used to specify which actions

are allowed or denied. For example, allowing a user named Alice to put but not delete objects in a bucket. So, on to my exam tips. And just remember that S3 is object-based and it allows you to upload your files. You can store any kind of file that you like, but it's not suitable to install an operating system or run a database on. Files can be anything from 0 bytes to 5 TB in size. And you get unlimited storage, so the total number of data and the total number of objects that you can store is unlimited. Files are stored in buckets, and S3 has a universal namespace, meaning that the bucket name must be globally unique, and this is what an S3 URL looks like. So we've got a bucket name, followed by a Region, followed by a key name or object name. And successful CLI or API uploads will generate an HTTP 200 status code. S3 objects consist of a key, which is the object name. And in my example, my object was called Ralphie.jpg. The value, which is the data itself, made up of a sequence of bytes which make up the file. A version ID, which allows you to store multiple versions of the same object. And metadata, which is data about the data that you're storing. Well that's it for this lecture. If you have any questions, please let me know. Otherwise, I will see you in the next lecture. Thank you.

Reviewing S3 Storage Classes

Hello Cloud Gurus and welcome to this lecture where we'll be reviewing the different S3 storage classes. We'll cover S3 Standard, S3 Standard-Infrequent Access, One Zone-Infrequent Access, Glacier and Glacier Deep Archive, Intelligent-Tiering, the relative costs, and my exam tips. So S3 Standard is designed for high availability and durability, and the data is stored redundantly across multiple devices in multiple facilities. And they store your data in a minimum of three availability zones. So you get 99.99% availability and 11 9s of durability. Now availability is all about the percentage of time that the S3 service is up and running and available to use; whereas, durability is all about your data being safe, and it's intact, and it's not getting lost or corrupted. S3 Standard is perfect for frequently accessed data and suitable for most workloads, is the default storage class, and use cases include websites, content distribution, mobile and gaming applications, and big data analytics. We then have S3 Standard-Infrequent Access, and this is designed for infrequently accessed data. So it's used for data that is accessed less frequently, but requires rapid access when needed. You pay extra to access your data, and there's a low per-GB storage price plus a per-GB retrieval fee, and this is great for long-term storage, backups, and disaster recovery files with a minimum storage duration of 30 days. And with this one, you get 99.9% availability and 11 9s durability. Onto One Zone-Infrequent Access, and this is just like S3 Infrequent Access. However, the data is stored redundantly, but within only a single availability zone. It costs 20% less than the regular S3-IA. It's great for long-lived, infrequently accessed non-critical data. There is a minimum storage duration of 30 days, and you get 99.5% availability and 11 9s durability. But the one thing to be aware of with this one is that If the single availability zone goes down or it's lost altogether, then you may lose your data. So it is only suitable for non-critical data. Onto Glacier, and there are a few different options with Glacier. Now Glacier is a very cheap and cost-effective way of storing your data. It's optimized for data that is very infrequently accessed. You pay each time you access your data. However, the data storage costs are relatively low, and you should only use it for archiving data. So first of all, we've got Glacier Instant Retrieval, and this is for long-lived data accessed approximately once a quarter. And when you access it, you need millisecond retrieval time. You get 99.9% availability and 11 9s durability. And then the second option is Glacier Flexible Retrieval, and this is for long-term data archiving for data that occasionally needs to be accessed within a few hours or minutes. And you get 99.99% availability and 11 9s durability. And for both

of these options, there is a minimum storage duration of 90 days. Onto Glacier Deep Archive. And this is long-term archiving for rarely accessed data with a default retrieval time of 12 hours. And this one is great for financial records that you only need to access maybe once or twice a year, or maybe you never need to access them at all. You just need to keep these records for regulatory purposes. The minimum storage duration is 180 days, and you get 99.99% availability and 11 9s durability. Now you might be thinking, I don't know whether I'll be accessing my data frequently or infrequently. Sometimes I need to access my data frequently, and other times I'll access it infrequently. Well, that is where S3 Intelligent-Tiering comes in, and you get two tiers, frequent and infrequent access. And with S3 Intelligent-Tiering, it will automatically move your data to the most cost-effective tier based on how frequently you access each object. So this is all about optimizing cost, and they do add a very small monthly fee of $0.0025 per 1000 objects that you store in S3 Intelligent-Tiering. So they're charging a fee, but it's a really, really low fee. And with this one, you get 99.9% availability and 11 9s durability. Now for the exam, you don't need to memorize how much the different storage classes cost. However, it is good to be aware of the relative or comparative costs. So S3 standard is generally the highest cost. We then have Intelligent-Tiering, which, of course, is cost-optimized and suitable for unknown access patterns. And then we have the infrequent access classes and Glacier. And with each of these, the storage cost is low. However, a retrieval fee applies. So for the exam, you will need to understand the characteristics of each storage class and suitable use cases for each of them. So S3 Standard is suitable for most workloads, for example websites, content distribution, mobile and gaming, etc. Standard-Infrequent Access is for long-term, infrequently accessed critical data. One Zone-Infrequent Access is for long-term, infrequently accessed non-critical data. And do be aware that with this one, it's storing your data redundantly, but in one single availability zone. So if that availability zone goes down or is lost altogether, then you may lose your data. Glacier Instant Retrieval is for archiving long-lived data, which is accessed approximately once a quarter, but needs a millisecond retrieval time. Glacier Flexible Retrieval is for long-term data archiving for data that occasionally needs to be accessed. And when you access, it needs to be within a few hours or minutes, and you've got retrieval options of 1 minute to 12 hours. Then there's a Glacier Deep Archive, which is for rarely accessed data archiving with a default retrieval time of 12 hours. And this one is great for financial records that you need to keep for regulatory purposes. And then finally, there's S3 Intelligent-Tiering, and this is the one to use if you have unknown or unpredictable access patterns. So that is it for this lecture. If you have any questions, please let me know. Otherwise, I will see you in the next lecture. Thank you.

Demo: Creating an S3 Bucket

Hello Cloud Gurus and welcome to this lesson where we'll be creating an S3 bucket. And we'll begin by creating our S3 bucket, and we'll give our bucket a DNS-compliant name. Next, we'll explore the available options in the S3 console. We'll upload a file from our local machine. And then finally, we'll grant public access to our bucket and the contents of our bucket, and we'll test that we can access our file. So if you're ready to get your hands dirty with S3, I'll see you in the console. So from the AWS console, search for S3 in the search box. And first of all, we're going to create a new bucket. So select Create bucket, and this needs to be a globally unique name. So I'm going call mine acloudguru-faye and then just add a load of random numbers on the end. And you might find that your first choice of name is not available. So if you do find that, just add some random numbers on the end to make a unique name. We'll create our bucket in the us-east-1 region. And then down

below, it gives you a few different configuration options and properties. So firstly, they've got the object ownership settings, and these are used to control the ownership of objects in S3. And they also control the use of ACLs, or access control lists. Now by default, the use of ACLs is now disabled. So they want you to use bucket policies to control access to S3, and we're going to have a go at creating our own bucket policy in a few minutes. Scrolling down to public access settings, and these are the default settings which block all public access. So they prevent public access to the data in your buckets. And by default, everything is going to be private. So there is no public access at all to your S3 bucket. And if you want to actually enable public access, for example if you're hosting a website or you want to enable anonymous access to read or write objects in your bucket, then you would need to uncheck all of these default boxes. However, to block all public access is really the recommended setting. Scrolling down to bucket versioning, and this is where you can enable versioning on S3. And that just means that S3 is going to keep all the different versions of an object within the same bucket. So let's say you accidentally delete an object. Well, the previous version will still be there. So it's a really good way to protect your data against accidental deletion. The next section is tags, and you can use tags to track your project or team costs, and tags are completely defined by you. For instance, I could add a tag of team, and my team name is the SysOps team. So this bucket is going to be owned by the SysOps team. The next option is encryption. And if we select Enable, you'll see that we've got a choice between two different types of server-side encryption. So we've got SSE-S3, which encrypts your data using an encryption key that S3 manages for you. And you've also got SSE-KMS, and this encrypts your data using an encryption key that is provided by the AWS Key Management Service, and we're going to go through S3 encryption in a lot more detail later on in this section. Then scrolling down to Advanced settings, this is where we have object lock. And this allows you to store objects using a write-once-read-many model, and it's also known as WORM. And that always makes me smile whenever I say it. And of course, the write-once-read-many model is going to help to prevent objects from being modified or overwritten for a fixed amount of time, or you can also set it to indefinitely as well. So this is a great option if you've got files that you want to store for a really long period of time, and you want to make sure that they don't get modified by anyone. So now, if you're happy with all the options we've selected, you can go ahead and create your bucket. So that is our bucket created. And if you've been successful, you'll see your bucket name appear down here in this global view. So now, let's select our bucket, and select Create folder. And we can actually create different folders within our bucket that we can use to organize our files in a way that makes sense for us. So I'm going to create a new folder, and my folder name is images. And select Create folder. Then click on our images folder, and I'm going to upload a file. So select Upload, Add files, and I'm going to select an image file from my local machine. Select Open. And now if we scroll down, select Destination details. First of all, it's reminding us that we can enable bucket versioning. Under Permissions, it's reminding us to use a bucket policy to grant access to our files. And then down at the bottom under Properties, this is where you can select the storage class that you want to use, and the default is Standard S3 designed for frequently accessed data that is accessed more than once a month with millisecond access. We then have Intelligent Tiering, which is great for data with changing or unknown access patterns. Then there's Standard-IA, which, of course, is for infrequently access data. There's One Zone-IA, and that's the same as Standard-IA, except the data only exists in one single availability zone, so it's not as highly available. However, it is more cost-effective, around 20% cheaper than Standard-IA. So it's a great option for data that's infrequently accessed, but for which you don't need the high availability option. Next, we've got Glacier Instant Retrieval, and that's great for archived data that's accessed once a quarter. You've got

instant retrieval times, and the minimum amount of time that you can store the data is 90 days. Then there's Glacier Flexible Retrieval, and that's great for long-term archiving for data that's accessed once a year with retrieval times ranging from minutes to hours, and the minimum amount of time that you can store the data is 90 days. Then there's Glacier Deep Archive, and this is great for longer-term data archiving for data that you're accessing less than once a year with a retrieval time of hours. And the minimum time you can store your data in Glacier Deep Archive is 180 days. And then finally, they've got reduced redundancy. And this was originally designed for frequently accessed data that was non-critical, so data that could easily be recreated if it ever got lost. And the reduced redundancy option is actually starting to disappear from the AWS documentation, and I think it's probably going to be discontinued very soon. And they even say down here that they no longer recommend this option because the S3 Standard option is now more cost-effective. And I really like that they've included this table here just to remind you of all the different storage classes available and the different use cases for each one. The next section is server-side encryption. And if you want to specify an encryption key, then you can do so. It's also possible to configure a checksum to verify the integrity of the data that you upload, and you can learn a little bit more about it by clicking on this link. But it's out of scope for the exam at the moment. Down here, we can add a tag to our object. And once again, tags are completely user-defined. So I'm going to add a tag of Project, and my project name is NYSummit. And then finally, we've got metadata. And of course, metadata is simply data about data. So we can add our metadata in here. We'll select system-defined metadata. I'm going to provide some metadata about the content type, so select Content-Type. And the value is going to be JPEG because we're adding a JPEG file. And of course, the metadata is completely optional. So if you're happy with that, scroll down to the bottom and hit Upload. So that is our file uploaded, and there it is. So if you select your file name, you'll see the that there are actually two ways to access your file. First of all, we can select Open, and there it is. It's just an image of me and Ryan at the AWS New York Summit. And then, if we head back to the console, the second way we can try and access the file is by using this object URL down here. So I'm going to right-click and open in a new tab. And we're actually getting this AccessDenied error. And the reason we get AccessDenied is because, by default, all S3 buckets are created without public access. So to access the object in this way using this URL, we're actually attempting to access the file as an unauthenticated user over the internet. So if we want to access the file in this way, we will need to explicitly enable public access for our bucket. So, how do we make this file publicly available? Well, let's head back to our console. We're going to select our bucket. Select Permissions. And then down here under the Block public access settings, select Edit, and we'll need to deselect all of these block public access settings. Scroll down and select Save changes. We need to confirm our options. And then the next thing we need to do is create a bucket policy that's going to allow public read access. So scroll down until you see the bucket policy. And by default, there is no active policy. So select Edit, and we're going to use the policy generator to create our bucket policy. But before we select that, I want to just copy this bucket ARN because we're going to use that in our bucket policy. So copy that to your clipboard, then click Policy generator, and this is going to create our bucket policy for us. First of all, select the type of policy, and it's going to be an S3 bucket policy. Next, we'll add our statements to the policy. So the effect is going be Allow. The principal is going to be Everyone. So we'll use the star wildcard to represent that. AWS service is Amazon S3. The actions are going be GetObject and GetObjectVersion. So scroll down until you find GetObject and GetObjectVersion. Next, we need to provide the Amazon resource name of our bucket, and this is the ARN that we copied to our clipboard earlier. So paste that into this section. And then, after the ARN, I want you to type forward slash and star so that it looks like that. And this just means that the

policy is going to apply to all the objects within the bucket. Once you've done that, click Add Statement. And the statement will appear down here, and this is what it should look like. So if you're happy with that, go ahead and click Generate Policy. And it just creates the policy code for you in JSON format. So you can go ahead and copy everything. Come back to your S3 bucket and paste that policy. And this is what it should look like. So this policy effectively is going to allow the S3 GetObject and GetObjectVersion, which are the permissions required for read-only access. So it's going to allow those permissions for this resource, so everything within this bucket for this principal, which means everybody. So if you're happy with that, scroll down to the bottom. If you see an error like this, just ignore it. Don't worry. It won't affect us. And just hit Save changes. And if it's all worked, your policy will appear down here. So now, if you scroll up to the top, select Objects, go into your images folder, select our object, find the object URL and click on that, your image should appear. And there we go. We've made our object public. So onto my exam tips. And just remember that S3 buckets and objects do not allow public access by default. We can enable public access, but we need to explicitly configure it. And to do that, first of all, we disabled the block public access settings on our bucket, and then we created a bucket policy, allowing anonymous read access to the contents of our bucket. And then finally, we can use S3 to store any kind of file that you can think of, for instance photos, videos, code, documents, or text files. But of course, you can't run an operating system or a database on S3. So that's it for this lesson. If you have any questions, please let me know. Otherwise, I'll see you in the next lesson. Thank you.

Working with S3 Lifecycle Policies

Hello Cloud Gurus, and welcome to this lecture, which is going to cover S3 lifecycle policies. And we'll begin with what are S3 lifecycle policies? We'll take a look at some examples, and we'll cover my exam tips, as well. So what are S3 lifecycle policies? Well, they are a very cost effective way to store your files in S3 and they allow you to save money and ensure that your files are stored using the most cost-effective S3 option throughout their lifecycle. We use lifecycle rules to tell S3 to transition objects to less expensive storage classes, archive them, or even delete them after a set period of time has elapsed. And S3 can do all of this automatically for you, based on the lifecycle rules that you define. And this is great for objects with a well-defined lifecycle. For example, log files, which may not be useful once they reach a certain age. So let's take a look at some examples of when we might use lifecycle policies. What about transaction logs, which may be useful, say, within the first 90 days of being created, but after a certain amount of time, you aren't going to be accessing them that frequently. So you could consider transitioning your log files to an Infrequent Access Storage class 90 days after you created them. Alternatively, you could archive objects to Glacier or Glacier Deep Archive, and this works best for things that you know you are not going to be accessing very frequently, if at all, once they get to 1 year old. And you can also configure objects to expire 1 year after creating them. And S3 will automatically delete objects which have expired, and you can set the expiry to a period of time that works for you and S3 will automatically delete the files for you. And one really good example for using lifecycle policies is with server access logging. And this is where you log all the requests to your S3 buckets. And of course, these logs end up being stored in another S3 bucket. And if you have many S3 buckets, and you have Server Access Logging enabled on all of them, and your accessing your buckets on, say, a daily basis, these access logs can really accumulate over time. And once those log files reach a certain age, you are most likely not going to be reading them, or accessing them, and you may not even want to keep them, so you can set an expiry date, and have S3 delete them for you. So for the exam,

just remember that you can use lifecycle policies to ensure that you are using the most cost-effective option to store your objects in S3. You can use a lifecycle policy to transition your objects to Infrequently Accessed Storage or to Glacier based on the rules that you configure. The lifecycle rules are based on an object creation date and you can also set an expiry date for objects that you want S3 to delete after a certain period of time has elapsed. So that's it for this lecture. If you have any questions, please let me know. Otherwise, I will see you in the next lecture. Thank you.

Protecting Data from Accidental Deletion Using S3 Versioning

Hello, Cloud Gurus and welcome to this lecture which is going to cover protecting data from accidental deletion using S3 versioning. And we'll begin with what is S3 versioning. We'll cover how it works, how it protects your data, and my exam tips as well. So what is S3 versioning? Well, with versioning enabled, S3 stores multiple versions of the same object, allowing you to revert to a previous version of an object should you ever need to. And by default it is not enabled. So if you would like to use S3 versioning, you will need to go in and enable it yourself. Now with S3 versioning enabled, multiple versions of an object are stored in the same bucket and you can restore every version of every object stored in your bucket. And with S3 versioning, a delete request doesn't actually delete the object. Instead it's going to apply a delete marker. And the delete marker actually becomes the current version of the file. And you can still access all of the previous versions of the file. You just need to specify the version ID that you want in the get request. So if you want to access the version of your file with a version ID of 1121, you just need to specify that version ID in your get request. And this gives you great protection from accidental or malicious deletions. But what if you want to permanently delete an object? Well, you can still do that. You just need to provide the version ID in the delete request. So to delete version 1121, we would need to provide the version ID of 1121 in our delete request. And when we perform a deletion, only the version corresponding to the version ID that we provided is going to be deleted and all of the other versions will remain. So for the exam, remember that with S3 versioning, multiple versions of an object are stored in the same bucket. A delete request doesn't delete the object. Instead, it's going to apply a delete marker. And the great thing about S3 versioning is that it protects your files against accidental deletion and you can still access all the previous versions of the file just by specifying the version ID in the get request. And you will also need to specify this version ID if you would like to permanently delete any version of your file. So that's it for this lecture. If you have any questions, please let me know. Otherwise, I will see you in the next lecture. Thank you.

Protecting Data from Accidental Deletion with MFA Delete

Hello Cloud Gurus, and welcome to this lecture, which is going to cover protecting your data from accidental deletion using MFA Delete, and we'll begin with, what is MFA Delete? We'll take a look at how it works and how it protects your data, and my exam tips as well. So what is MFA Delete? Well, it uses multifactor authentication to provide an additional layer of protection to S3 versioning, and it requires S3 versioning to be enabled. So how does it work? Well, you need a physical, or virtual, MFA device to generate an authentication code, which is used as an additional layer of authentication to any delete requests. So how does it protect your data? Well, you will need a valid code from your MFA device in order to permanently delete an object version or to disable or re-enable versioning. And this is how you would specify it. So you specify the ARN of your MFA device and provide the authentication code that it is displaying. So, onto my exam tips. And for the exam, you don't need to be able to configure MFA Delete, you just need to know what it is and why

you would use it. So with MFA Delete, you will require a valid code from your MFA device in order to permanently delete an S3 object. S3 versioning must be enabled in order to enable MFA Delete, and a valid code from your MFA device is also required in order to suspend or reactivate S3 versioning. And in combination, MFA Delete and S3 versioning protect your data against accidental or malicious deletions of your version-controlled S3 buckets. So that's it for this lecture. If you have any questions, please let me know. Otherwise, I will see you in the next lecture. Thank you.

S3 Encryption

Hello, Cloud Gurus, and welcome to this lecture, which is going to cover S3 Encryption. And we'll begin with the types of encryption available. We'll cover how easy it is to enable S3 encryption. And my exam tips, as well. Now, there are a few different types of encryption available with S3. So let's begin with encryption in transit. And that really means that you're encrypting the data when you're sending it to and from your S3 buckets. And that is done using SSL or TLS. And TLS, or Transport Layer Security, is really replacing SSL, but sometimes these terms are used interchangeably. So if you see SSL or TLS, then just know that that is encryption in transit. And typically, that means that we are using HTTPS to upload and download files between our local system and S3. So moving on to encryption at rest. And this is where the data is encrypted when it is stored on disk, and it's also known as server-side encryption. And with S3, there are 3 different types of server-side encryption available. So firstly, we have SSE-S3, which encrypts your data using encryption keys that are managed by S3, and each object is encrypted using its own unique key. And as an additional step, they also encrypt the key itself with a master key, which AWS rotate for you. So Amazon manages the keys, and you don't need to worry about managing your own keys. And it uses AES 256-bit encryption. So that is the advanced encryption standard, which is an industry standard method of encryption. Now the second method of encryption they offer is called SSE-KMS. And this uses the AWS Key Management Service. So once again, AWS manage the keys for you, but KMS comes with some additional benefits. So firstly, you get separate permissions for the use of an additional key, and they call it an envelope key. And the envelope key is a key which actually encrypts your data's encryption key. And it gives you this added level of protection against unauthorized access. And in addition to that, you also get an audit trail, and the audit trail records the use of your encryption key. So you can see when your key was used, who used it, and what they did as well. And we'll be covering KMS in much more detail in the Security section of this course. And then finally, we also have SSE-C, and this uses an encryption key that you provide yourself. So it's a customer-provided key. So AWS will still manage the encryption and decryption activities, but you are responsible for managing your own key. And the advantage of that is that you are in charge of administering the key, rotating the key, and the lifecycle of your key as well. And this is great if you're operating in a highly regulated environment and you need to be responsible for your encryption keys. And then lastly, we have client-side encryption, and this is another form of encryption at rest. And with client-side encryption, this is where you encrypt the files yourself before you upload them into S3. And with this option, you can choose your own encryption method, and you encrypt your files locally before uploading them into S3. And when it comes to enabling S3 encryption, it is really easy, and you can configure it in the console either when you create your bucket or afterwards. So you just select Enable, and then select the encryption type that you would like to use. So on to my exam tips for S3 encryption. And we use encryption in transit to encrypt data as it travels over the network. And that uses SSL or TLS, also known as Transport Layer Security. And we can use the HTTPS protocol when we upload files to S3, and that will give us

encryption in transit. And when it comes to encryption at rest, that can either be server-side encryption or client-side encryption. And with server-side encryption, we've got 3 options. So we've got SSE-S3, so encryption using S3 managed keys. We've got SSE-KMS, which uses Amazon KMS managed keys, or Key Management Service. And we have SSE-C, which uses customer managed keys. And then we've also got encryption at rest, also known as client-side encryption. And this is where you encrypt the files yourself locally before you upload them into S3. So that's it for this lecture. If you have any questions, please let me know. Otherwise, I will see you in the next lecture. Thank you.

Demo: Configuring Encryption on an S3 Bucket

Hello Cloud Gurus and welcome to this lesson where we'll be configuring encryption on an S3 bucket. And we'll begin by creating a new S3 bucket. We'll then take a look at the server-side encryption options that are available within S3. We'll then enforce server-side encryption using a bucket policy. And you can also do this very easily using the AWS console. But in the exam, they do sometimes test you on how to enforce encryption using a bucket policy as well, so we're going to focus on doing it using that method. And then finally, we're going to test our setup. So we'll test our policy by trying to upload an unencrypted file, and we'll see if it gets blocked by our bucket policy. So if you'd like to join me in the AWS console, we'll get started. So from the management console, search for S3 and select Create bucket. Give your bucket a unique name. The region is going to be us-east-1. And then scroll down until we find Default encryption, and there it is. Now there are two ways to enable encryption. You can either enable it here on the console, and this is the easiest way. And then you can go ahead and choose between the S3-managed keys or use KMS-managed keys. And both of these types of encryption are server-side encryption. But you can also enforce encryption on your S3 bucket by using a bucket policy. And they do test you on that sometimes in the exam, so it is well worth knowing about. And for this lesson, that's exactly what we're going to do. So disable that here, and we'll enable it using a bucket policy. So just go ahead and hit Create bucket. And then to set up the policy, we just select the bucket, select Permissions, and scroll down to the bucket policy. Select Edit. Select the policy generator. And this is where we're going to build our policy. So the type of policy is going be an S3 bucket policy. The effect is going be deny. So select Deny because we're going be denying requests which do not enforce encryption. The principal is going be star, so this will apply to any request. The AWS service is Amazon S3. Under Actions, we're going to select PutObject. So select the drop-down and scroll down until you find PutObject. And there it is, so select PutObject. The Amazon resource name is going to be the ARN of your bucket. So I'm going to head back to the screen with my bucket, my bucket policy. Here's my bucket ARN, so I want to copy that. Come back to the generator. Paste in the bucket ARN. Next, we need to add a condition. So select Add Conditions. And the condition is going to be when the string does not equal. So select StringNotEquals. And we want to deny if the string does not equal AMZ server-side encryption. So select the drop-down. And before we find that, I just want to make you aware that we can also use this same policy method to enforce encryption in transit. So if we selected AWSSecureTransport, this is the one to choose if you want your policy to enforce encryption in transit or HTTPS. But for this lesson, we're going to select server-side encryption. If we scroll down, here it is. It's under s3:x-amz-server-side-encryption. So that's the one that you need to select. And then down here under Value, this is where we need to specify the type of encryption that we want to use. So let's use KMS. So enter aws:kms and then hit Add Condition. So now, we can go ahead and hit Add Statement, Generate Policy, and this is the JSON code that the

policy generator has created for us. So here's our policy ID, here's our version, here's our statement, and this is the action that the policy is going to enforce. So our action is going be s3:PutObject. Effect is deny. Here's our resource, and here's our condition. So this policy is going to deny any requests to put a new object when the request header does not include x:amz-server-side-encryption, aws:kms. So it's searching for this string in the request header of the put request. And the principal is star, so that means it applies to anybody who tries to upload a file into this bucket without using server-side encryption. So we need to copy all of this code, come back to our console, paste into our policy editor, scroll down, and save changes. And straightaway, we've got an error message. We can actually ignore this message up here, but this is the one I'm interested in, the API response. It says Action does not apply to any resources in statement. And I wanted to show you this error message because this is something that you see sometimes when you create new bucket policies using the policy editor. So the error we've got here is saying that the action does not apply to resources in statement. So if we come back up here, take a look at our resource section, it's got a problem with how we've specified our resource. And this happens sometimes just because some services do not let you specify specific actions for individual resources. And in this case, you usually need to add a wildcard to the resource element. So that just means that the action will apply to all resources within that service and not the bucket itself, so everything within the bucket. And we can manually resolve this just by adding a wildcard. So we'll do that right now. So just add forward slash and then star, and that should fix our problem. So scroll down to the bottom, save changes, and there we go. Our policy has been saved. So now, we are ready to go ahead and test that our policy is working correctly. So scroll up to the top, select Objects, and we're going to try and upload an object. So select Upload, Add files, and I'm just going to select a text file from my local machine. Any file will do. Select Open. If you select Properties, scroll down, this is where we can select our encryption options. And it's warning us that if your bucket policy requires encrypted uploads, you must specify an encryption key or your upload will fail. So let's go ahead and try it and see if our upload fails. Scroll down to the bottom and select Upload. And there we go. If it's all worked correctly, you should get this message saying upload failed. And it's giving us this Access Denied message, saying you do not have permission to upload files and folders. Okay, so let's try again. And this time, we'll try and upload our file using encryption. So close that down, select Upload again, Add files, select our file, Open, scroll down to Properties. And this time, we are going to select Encryption. So we'll specify an encryption key. And this time, we're going to select AWS Key Management Service. We'll stick with the rest of the defaults. Scroll down and Upload. And there we are. That should be successful. And we can see that our upload has succeeded. So now, let's try and upload the file again using a different method of encryption. Close that down, select Upload, Add files, select the file, Open, scroll down to Properties. We'll specify an encryption key. And this time, we're going to use SSE-S3. So scroll down and Upload. And once again, that should show you the same error message that we saw before, Access Denied. And if we select the error down here, it's saying we don't have permission to upload files and folders because even though we used encryption this time, we didn't use the correct type of encryption that our bucket policy was expecting. So for the exam, just be aware of the different types of encryption that are available with S3. So there's encryption in transit, which covers SSL or Transport Layer Security. There's also encryption at rest, which is server-side encryption. And the options available include SSE-S3, SSE-KMS, and SSE-C. And encryption at rest is the type of encryption we've been working with just now. So just remember that you can enforce that using either the AWS console or with a bucket policy like we did just now. And then there's also encryption at rest using client-side encryption, and that's when you encrypt the files

yourself before you upload them into S3. So that is all for this lesson. Any questions, please let me know. Otherwise, feel free to move on to the next lecture. Thank you.

Introduction to Elastic File System (EFS)

Hello Cloud Gurus, and welcome to this lecture which is going to introduce Elastic File System or EFS beginning with what is EFS? Next, we'll take a look at an EFS example scenario, enabling encryption at rest. We'll review the EFS storage classes that are available and my exam tips as well. So what is EFS? Well, it's a managed network file system and it's an AWS managed service, it is highly available and scalable as well. It uses the standard NFS protocol, which is a standard used by Linux Systems and EFS is for Linux based workloads only. Multiple EC2 instances can access the file system at once. So that means it's great for applications which need to access shared files. For example, shared configuration files or state information for your application or even user home directories. And sharing a file system like this is not possible with Elastic Block Store. So you cannot share your Elastic Block Store volumes with multiple servers, but you can share your file systems between multiple servers using EFS. It also has a lifecycle management capability so any files in your Elastic File System which are not accessed for a set period of time that you define will automatically move so that EFS infrequent access storage class so that will save you money as well. And, finally, you can configure encryption for your data both at rest and in transit. So let's take a look at an example scenario. Now, EFS is a fully managed network file system. So it's a file system that can be accessed by multiple servers at once. So it's a place where you can store your files that you want to share between multiple systems. And when we say multiple systems, this can be across different availability zones, across regions, and even across VPCs as well and even servers located in your own data center can connect to an EFS file system. But what about enabling encryption at rest? Well, it's really easy to enable. It's just a checkbox in the console. However, encryption at rest can only be enabled when you first create the file system. So you enable it at file system creation. And if you decide to encrypt an EFS file system later on after it's already been created, you will not be able to do that. Instead, you must create a new encrypted EFS file system and then migrate your files from the original file system to the encrypted EFS file system. So just remember that you can only enable encryption at file system creation. So now let's review the different storage classes that are available with Elastic File System. So first of all, we've got EFS standard which is the default, and this is great for storing frequently accessed data that needs the highest availability. Your file system has a durability of 11 nines and an availability of 99.99 %, and the data is stored in a minimum of 3 availability zones. Next, we've got EFS standard infrequent access and this is designed for infrequently accessed data which needs the highest availability. So once again, we've got 11 nines durability availability of 99.99 %. The data is stored in a minimum of 3 availability zones and you do pay a per GB retrieval fee. So it's only going to be cost effective for infrequently accessed data. We then have EFS one zone and this is designed to store frequently accessed data which doesn't require the highest availability. So once again, we have a durability of 11 nines but the availability is only 99.9 % because the data is only stored in one availability zone. However, it's stored redundantly within a single AZ. So that means it is not resilient to the loss of an availability zone. And then finally we've got EFS one zone IA and this is designed for infrequently accessed data which doesn't require the highest availability. We've got a durability of 11 nines and availability is 99.9. And once again, it's storing the data redundantly in one single availability zone so it's not resilient to the loss of an availability zone. And you will also incur per GB retrieval fees. So it's only going to be cost effective for infrequently accessed data which doesn't require the highest

availability. And just remember that data in the one zone classes may be lost if a fault or disaster affects all copies of the data within the availability zone or in the event of AZ destruction. So the one zone classes are not suitable for your mission critical production data. So let's review our exam tips for EFS. And just remember it's a managed network file system so it's a highly available and scalable shared file system for Linux based workloads. It's great for applications which need to access shared files, for example, shared configuration files or state information for your application which needs to be accessed by multiple EC2 instances. And remember, it can also be accessed by on-premises systems as well. And when we talk about sharing data in this way, you cannot do this with Elastic Block Store. EFS also has lifecycle management capabilities and that means that files which have not been accessed recently, based on a period of time that you define, can get moved to an EFS infrequent access storage class. And that is going to save you some money. And then when it comes to encryption, you can enable encryption in transit and at rest but for encryption at rest, you will need to enable it at creation time and you cannot enable encryption at rest on an EFS file system that has already been created. But the best way to learn about EFS is to actually have a go at creating your own EFS file system. And that's exactly what we'll do in the next lecture. So if you've got time, please join me for the next lecture. Thank you.

Demo: Working with EFS

Hello Cloud Gurus and welcome to this lesson where we'll be getting our hands dirty working with AWS. And we'll begin by creating an EFS file system. And note that in the console, encryption will be enabled by default. Next, we'll launch an EC2 instance, and we're going to install the Amazon EFS utilities on our EC2 instance. And then finally, we will mount our EFS file system. So we'll need to create a new directory on our EC2 instance, and then we'll mount our EFS file system to our new directory and create a file. So if you'd like to join me in the AWS console, we'll get started. So from the console, first of all, we're going to search for EFS. Select Create file system. I'll call it MyEFS. Use the default VPC. And by default, EFS is going to store your data across multiple availability zones. And by default, EFS will store your data across multiple availability zones, and that is the Standard storage class. But you can select One Zone instead if you want to save money, and you don't mind reduced availability for your data. So with One Zone, it's going to store the data redundantly within a single availability zone. So now, I'm going to select Customize so that we can review the options that are available. Scrolling down, you'll see that backups are enabled automatically. Lifecycle management is also enabled by default. So by default, it's going to move files into the Standard-Infrequent access storage class if they have not been accessed for a number of days. And by default, it's 30 days. And you can disable this by selecting None, but let's stick with the default. And then it's going to transition out of Infrequent Access back to Standard the first time you try to access the file. Now encryption is also enabled by default using the default encryption key for the EFS service. Alternatively, you can use your own encryption key by creating one here, but we'll just stick with the default. Moving on to performance settings. Now for throughput, you've got two different throughput modes available. And throughput is, of course, all about the amount of data transferred to and from a storage device. Now bursting throughput gives throughput that scales with the size of your file system, and this is going to be fine for most applications with basic performance requirements. However, if you find your applications are struggling and the file system is just too slow, then you can go for enhanced mode, and this is going to be better if you have any applications that need higher throughput levels. Under Additional settings, you'll find some more performance settings. And these settings are all about the number of IOPS that can be processed or

I/O operations per second. So this relates to the number of read and write operations that can be performed in a second within the storage device itself. And they recommend general purpose for most file systems, so use cases like web servers or content management systems, home directories, and just general file serving, including high-performance and latency-sensitive applications. So this is the one that they recommend. However, there's also the Max I/O performance mode, which is optimized for applications where you've got tens or hundreds of thousands of EC2 instances, and they are all accessing the same EFS file system. So this is designed for highly parallelized workloads that can tolerate higher latencies. So you've got lots of systems accessing, but the latency is going to be a little bit higher, so it's a different type of use case. So then, if you are happy with those settings, hit Next. Then, under Network access settings, this is where you can configure your mount targets. Now mount targets provide an NFS version 4 endpoint that you can use to mount your EFS file system after it's been created. So your EC2 instances are going to connect to the EFS file system using these mount targets, and they recommend creating a mount target in each of your VPC's availability zones. And in this particular VPC, it's us-east-1, and I've got six different availability zones. And by default, EFS is going to create a mount target in each one of those. So I'm going to stick with the default and hit Next. Under the file system policy, we can optionally select some of these security options on the left-hand side. So we can prevent root access by default, enforce read-only access, prevent anonymous access, or enforce encryption in transit, and I'm going to select that one. So if we select one of these options, it will automatically populate the policy on the right-hand side. So now, scroll down to the bottom and hit Next. On the screen, we can just review everything. And scroll down and Create. So that is our file system created. And if we click on our file system and select Attach, it's going to show us the commands hat we need to run on an EC2 instance in order to mount this file system. So now, let's go ahead and create an EC2 instance, and we'll give it a go. I'm going to close down that message and open up the EC2 console in a new tab. Now you don't need to know how to connect to an EFS file system for the exam. It's just a fun thing to do, and it's only going to take us a few minutes. So let's quickly create an EC2 instance, Launch instance, call it EFS Client, scroll down. Instance type will be t3.micro. We don't need a key pair. Under Network settings, select Edit. And I'm going to change the name of my security group so that I can identify it later on and then Launch instance. Now while our instance is initializing, I'm going to head back to EFS. Within my file system, I'm going to select the Network tab and scroll down. And then, on the right-hand side, here is the security group that is associated with our EFS file system, and it's the same security group for each of our availability zones. Now in order to allow our new EC2 instance to connect to this EFS file system, we need to update this security group. So just make a note of the name of this security group and head back to the EC2 console. Select EC2. On the left-hand side, scroll down and select Security Groups. So here's my two security groups. This one is the one associated with my EFS Client, my EC2 instance. And then it's this one that I'm looking for. This is the one associated with my EFS file system, and this is the one that we need to change. So select that one, scroll down, and Edit inbound rules. We'll add a rule. The type is NFS. Protocol is TCP. It's going auto populate the port range of 2049. Source type is custom, and then the source of this NFS traffic is going to be the security group of my EC2 instance. So type sg, and here is the security group. So select your EFS client security group and Save rules. And now, we should be ready to mount our file system. So first of all, let's connect to our EC2 instance. Select Instances on the left-hand side. Select your instance using this box and select Connect. We'll use EC2 Instance Connect, so hit Connect. And before we can mount our EFS file system, there's one utility that we need to add to our EC2 instance, and it's called the EFS mount helper utility, and it's part of the Amazon EFS utils package. So I want you to type sudo yum install

-y and then amazon-efs-utils and hit Enter. And that has installed the EFS utils package. Next, we need to create a mount point locally on our EC2 instance that we're going to use to mount the EFS file system too. I'm going to clear my screen and create a new directory. So type sudo mkdir efs and hit Enter. And this command just creates the directory, and the directory's name is going be efs. So now, for the next command, we'll head back to the EFS console. From your EFS file system, click Attach. And this is going to show us the command that we need to use to mount our file system, and this will actually make the file system usable so that we can change directory and start creating files inside. So let's copy the first command, come back to our EC2 instance, and paste. So with this command, we are mounting a file system of type efs. We're using the TLS option for transport layer security, which is going to give us encryption in transit. We're providing the EFS file system ID. And we're going to mount the file system to our local directory called EFS. So now hit Enter. And the way that you can check is just to type the following command, df -T and hit Enter. And then down at the bottom here, this is our EFS file system. So the type is NFS 4, and here is our mount point, which is the EFS directory inside our home directory. So now, we should be able to use the EFS file system. So I'm going to change directory to my EFS file system. Just type cd efs and hit Enter and then sudo touch test.txt. So we're just creating a text file in our directory. And then, if we run ls, there is our file, and it's been saved into our EFS file system. So that is EFS. Let's take a look at my exam tips. And for the exam, just remember that EFS stands for Elastic File System, and it is file-based storage that can be mounted by multiple EC2 instances. It's supported for Linux-based workloads only, and it runs the NFS protocol. And finally, encryption is enabled by default when you create an EFS file system using the AWS console. But do remember that you can only add encryption when you create the EFS file system. You cannot add it on as an option later on. So that is it for this lesson. If you have any questions, please let me know. Otherwise, I will see you in the next one. Thank you.

Advanced EFS

Hello Cloud Gurus, and welcome to this lecture where we'll cover Advanced EFS. And we'll begin by reviewing the throughput modes available with EFS, and we'll then take a deeper dive into bursting throughput and how it works. We'll cover what is metered throughput. We'll also cover provisioned throughput in more detail, and my exam tips as well. Now as you know, there are two kinds of throughput mode available with EFS. So firstly, we have bursting throughput, which is the default mode. And with bursting throughput, your throughput scales as your file system grows, and it supports periodic bursting to cater for peaks in workload. And then we have provisioned throughput, and this is where you optionally define the throughput that you want. And this is designed for applications which consistently need high performance. An EFS file system can support many thousands of simultaneous connections, and throughput is measured in mebibytes per second, which is just a unit of measurement that is used when we talk about data storage. And for those interested, 1 MiB is the equivalent of around 1.04 MB. So bursting throughput increases with your file system size, and as workloads can often be spiky, EFS has been designed to periodically burst to higher throughput levels to accommodate peaks in usage. And as a minimum, all EFS file systems can burst to at least 1 MiB/s. File systems that are over 1 TiB in size of standard storage class can burst to 100 MiB per TiB of data stored. Therefore, if you have an EFS file system that is 10 TiB in size, that file system will be able to burst to 1000 MiB of metered throughput. And we'll explore what they mean by metered throughput in just a second. And you might be wondering, how long can an EFS file system burst for? Well, the larger the file system, the greater the bursting

throughput and the longer the duration that it will be able to burst for. And there's a great web page here, which gives you an idea of what to expect, and I've added a link to this web page in the resources section for this lecture. So, an EFS file system that is 100 GiB in size is capable of bursting to 300 MiB/s for read-only workloads for up to 72 minutes per day plus 100 MiB of write-only throughput for up to 72 minutes per day, which brings us to what is metered throughput. Well it's actually a blend of read requests and write requests. And with EFS, read operations are meted at a 1:3 ratio of write requests. So that means that you get up to 3 MiB of read throughput and 1 MiB of write throughput for every 1 MiB of throughput provisioned. And it might sound complicated, but you don't need to remember this for the exam. It's just good to know that when we talk about metered throughput, we are actually talking about a blend of read and write throughput. And of course, an EFS file system is generally going to be read heavy, so it makes sense that they're thinking about throughput in this way. So now moving on to provisioned throughput. And this is the option to choose if you want the freedom to define your own throughput capability. Let's say you're storing 1 TiB of data and you want greater throughput than the 100 MiB/s that you are going to get with the default burstable mode. Well, in this scenario, you will need provisioned throughput. And provisioned throughput will allow you to define the throughput that you want in mebibytes based on the requirements of your application and irrespective of the size of the file system. For example, we can define 500 MiB of provisioned throughput. Now when it comes to charging for provisioned throughput, you are only charged for the throughput which is provisioned over and above what you get based on the data you have stored. And in practice, you should start off with the default burstable option, and if you find that it's not performant enough for your application, then you can switch to the provisioned throughput option. And as a rule of thumb, if your applications need more throughput than you are getting with bursting throughput, then you should switch to provisioned throughput. And you can always change back if you need to. So you can change back from provisioned throughput mode back to bursting throughput mode as long as it is over 24 hours since your last throughput mode change. So, onto my exam tips. And just remember that a single EFS file system can support many thousands of simultaneous connections. By default, EFS file systems are created using the burstable throughput option, and this means that the throughput that you get is determined by the amount of storage you have. All file systems get a minimum of 100 MiB of burstable throughput, and that 100 MiB is a blend of read and write throughput. And standard class file systems which are greater than 1 TiB in size can burst to 100 MiB per TiB of data stored. However, if you want to optimize performance for your application, then you should use provisioned throughput, and this will allow you to define your throughput capacity in mebibytes based on your requirements. And as a rule of thumb, if you need more throughput than you're getting with burstable throughput mode, then you should switch to provisioned throughput. So that's it for this lecture. If you have any questions, please let me know. Otherwise, I will see you in the next lecture. Thank you.

EFS and Multi-AZ Applications

Hello Cloud Gurus, and welcome to this lecture which will cover EFS and multi-AZ applications. And we'll begin with working with multiple AZs and One Zone EFS. We'll cover working with multiple AZs and Standard EFS and we'll finish off with my exam tips. Now when you create an EFS file system, you will need at least one mount target and a mount target is used by EC2 instances to mount the file system. And if you are using an EFS One Zone storage class, then you can only create one mount target and it must be in the same availability zone as your EFS file

system. And that means that you will incur data access charges for any instances that are located in a different availability zone to the mount target. So in my example, any instance that is located in us-east-1b will incur data access charges for accessing the mount target that is located in us-east-1a. However, if you are using EFS Standard storage class, then you can configure mount targets in each availability zone in the region that you are operating in. And this will avoid any additional data access charges. And if there is a problem with the mount target located in us-east-1a, then the instances in us-east-1b will be unaffected. And this is the preferred option if you are working with instances in multiple availability zones because it gives you the highest availability. And for the exam, just remember that EFS Standard includes multi-AZ resilience so you can create a mount target in each availability zone in the AWS region that you are operating in. Multiple mount targets are preferable because this avoids any additional data access charges for hosts that are located in a different availability zone to your mount target. And with multiple mount targets, you also get increased availability. So if there is a problem with the mount target in one availability zone, then the instances in the other availability zone will be unaffected. So that's it for this lecture. If you have any questions, please let me know. Otherwise, I will see you in the next lecture. Thank you.

Introducing Athena

Hello, Cloud Gurus, and welcome to this lecture, which is going to introduce Athena. And we'll begin with, 'what is Athena? '. We'll take a look at some Athena use cases. And my exam tips, as well. So, what is Athena? Well, Athena enables you to run the standard SQL queries, or Structured Query Language, on your data stored in S3. So, it's an interactive query service. Athena is also serverless, so there's nothing to provision. And you pay per query and per TB scanned. It's really easy to get started, and there's no need to set up any complex Extract Transform, and Load, or ETL, processes. And it is integrated with your data, so it works directly with data stored on S3. So, when would we use Athena? Well, it's really useful for querying log files stored in S3. For example, Elastic Load Balancer logs, S3 access logs, and you could also use it to query your CloudTrail logs as well. Athena is also great for performing cost analysis, and you can use it to analyze your AWS Cost and Usage reports stored in S3. You can also use it to generate business reports on data that you've stored in S3. And analyze click-stream data. So you can use it to run queries on your click-stream data stored in S3. So, on to my exam tips. And for the exam, just remember that Athena is an interactive query service. And it allows you to query data stored in S3 using standard SQL, or Structured Query Language. And it is also serverless, so you don't need to worry about configuring any infrastructure if you want to use Athena. So that is it for this lecture. And to really get your head around how Athena works and what it does, you will need to get your hands dirty with Athena. And that's exactly what we'll be doing in the next lecture. So, if you're ready to get started using Athena, then I will see you in the next lecture. Thank you.

Demo: Working with Athena

Hello Cloud Gurus and welcome to this lesson where we get some experience working with Athena. And we'll begin by creating a trail in CloudTrail. And this is going to generate an audit log of all the user activity on our account. Now, as you know, CloudTrail sends logs to S3, and our new CloudTrail is going store all the logs it creates in a brand new S3 bucket. Next, we'll create an Athena table, and we're going to use standard SQL or structured query language to query the data stored in our S3 bucket. So if you're ready to get your hands dirty with Athena, then please join me in the AWS console. So from the AWS console, first of all, search for CloudTrail. We can ignore

these messages, so just close them down. And then from the CloudTrail dashboard, select Create trail. We can ignore this message, so just close that down. And our trail is going be called management-events. And as you know, the CloudTrail logs are going to be stored in an S3 bucket, and it's going to create a brand new S3 bucket for us. By default, it's going to encrypt our CloudTrail. And we need to enter a new KMS key alias because we're going to create a new KMS key to handle the encryption. Scrolling down, select Next. Make sure that the event type of Management events is selected. Hit Next, Review, and Create trail. So there we go. It's created our trail. And if we select the S3 bucket over here, we should see the that it's already started to create some folders. So if you select the CloudTrail folder, after a few minutes, it should start to populate this folder structure. So just be patient. And after a few minutes, we should start to see some logs appearing. But it might just take 5 minutes or so to get started. After a few minutes, I'll refresh my view, and here are all the logs it's created. And if you select one of the regions where it's already started logging and drill down, here our the logs it's created. So now, we need to create another S3 bucket that's going to be used to store our Athena query results in, and this is mandatory before we can use Athena. So select Buckets on the left-hand side. Create bucket. I'm going to call it myathena-results-faye. The region is going be us-east-1. Scroll down to the bottom and Create bucket. So now we are ready to configure our Athena database. So search for Athena and open it up in a new browser tab. Now before you run your first query, you will need to set up the query result location in S3. So first of all, on the left-hand menu, select Query editor. And it's telling us here that before we run our query, we need to set up our query result location. So select Edit settings. But if you don't see the this message, you can just edit your settings by selecting the Settings tab here at the top of the screen. Browse S3, select your results bucket, and hit Save. So now, we are good to go to create our Athena database, and we're going to do that using a simple SQL query. So select Editor, and this is where we're going to type our query. Click in the box, and I want you to type the following command, CREATE DATABASE athenadb and then select Run. And if it's been successful, if you scroll down at the bottom, you will see a message saying Query successful. So that has created our database. And on the left-hand side under Database, you should find our Athena DB database. And if you don't see it, you can just select the drop-down, and you should be able to find it. Now our database is empty at the moment, so we need to create a new table and then tell Athena where to find the data that we want to use to populate our table. And I've already prepared the SQL command that we're going to use to do this. And in the Resources section of this lesson, you will find a link to the GitHub repository that contains my query commands. Now the first one I want you to take a look at is the Athena_Query_Create_Table.txt. If we select Raw, we can then copy everything. Come back to Athena. Hit the plus sign to create our second query. Paste the code in here, and let's take a look at our code. So this is a really long piece of SQL code. And all it does is create our table, and our table is going be called cloudtrail_logs. And then down here, it's creating all the different attributes that are going to be part of our table. So things like the user identity, account ID, event time and event source, AWS region, etc., all the kind of information that you would expect to see in a CloudTrail log. So these are all of the attributes of our table. And then right down at the bottom here, this is where we need to specify the location of our data. So we need to tell Athena where to find this data, and it's going be in our CloudTrail log S3 bucket. So if we head back to the S3 console, select the CloudTrail log, go into AWSLogs, and then the next folder, which is named after your account number, and your folder structure should look like this. So here's your bucket name, then AWSLogs, and then your AWS account number. So now just copy the S3 URI, come back to Athena, and we need to replace this URI with the one from our bucket. And once you've done that, it should look like this. So here's our bucket name, then AWSLogs, and then our

account number. So if you've done that, just hit Run. And if everything's worked correctly, you should get a message at the bottom saying Query successful. And if it hasn't been successful, just make sure that you've specified this location correctly because if you don't specify this correctly, then Athena will not be able to find your data. So now, under the Tables section on the left, you should see a new table named cloudtrail_logs, expand the table, and here are all the data attributes that our query has just created. And now, we are actually ready to start querying this data using Athena. So at the top, click on the plus sign to create another query, and I've prepared this query for you already, and you're going to find it in the resources of the course. So I'm going to hit the Back button on this screen, go up one directory, and it's this file here called Athena_Query.txt. I'm going to copy that, paste into my query window, and hit Run. And this is just a really simple query that selects the user identity ARN, event name, source IP address, and event time from our cloudtrail_logs table. And then if you scroll down, you can take a look at the results, and here they are. So it's selected the user identity, the event name, source IP address, and event time from the data in our cloudtrail_logs table. So that is Athena, and I hope you can see that it's really, really simple to use. And if you know how to use SQL, then you can run all sorts of queries on your data, and you can really just treat it just like a database and run whatever queries you want. So let's move on to my exam tips and just remember that Athena is a query service that supports standard SQL or structured query language. It's really easy to configure, and all you do is point Athena to the data that you want to query in S3, define your table schema, and then you can query your data using standard SQL. So that's it for this lesson. If you have any questions, please let me know. Otherwise, I will see you in the next lesson. Thank you.

Introducing Amazon OpenSearch Service (Formerly Amazon Elasticsearch Service)

Hello Cloud Gurus, and welcome to this lecture which is going to introduce the Amazon OpenSearch Service. And it used to be called the Amazon Elasticsearch Service. And we'll begin with what is Elasticsearch, what is the Amazon OpenSearch Service, OpenSearch features, and my exam tips. So what is Elasticsearch? Well, it's a widely used open-source data analysis technology, and it allows you to get real-time insights from your data. But what does that actually mean? Well, many companies use Elasticsearch to analyze their business data to gain insights and make better business decisions. And you can also use it to search your application, infrastructure, and security logs to really understand how your systems are operating. But what about the Amazon OpenSearch Service? Well, designing, deploying, and administrating your own Elasticsearch cluster can be a time-consuming and complex process, and that is where the Amazon OpenSearch Service comes in. And just remember, it was formally known as the Amazon Elasticsearch Service, so you may see it referred to by either name in the exam. So what is the Amazon OpenSearch Service? Well, it's a fully managed Elasticsearch service, and AWS does all the heavy lifting for you. So that means that they take care of the hardware provisioning and configuration of the Elasticsearch cluster, the software installation and patching, and they also handle failure recovery, automated backups, and monitoring as well. So let's take a look at some of the cool features of the Amazon OpenSearch Service. Now, when you launch an OpenSearch environment it actually consists of a number of different EC2 instances that make up what they call an OpenSearch service domain, and you might see it referred to as either a domain or a cluster, and that really means the same thing. So first of all, you've got the master nodes, and these are responsible for managing the cluster and monitoring the health of all the nodes in the cluster. So these are really the management nodes. And you've also got

the data nodes, which are responsible for storing the data, and performing searches and query requests on the data. And these are really like the worker nodes. And one of the great things about the Amazon OpenSearch Service is that they make it really easy to create a cluster with all of the enterprise-level features that you would need in a production environment. For example, your cluster can automatically detect and replace a failed node. If you need more data nodes, because you've suddenly got more queries and searches to run, you can scale your cluster with a single API call, or a few clicks in the console. And you can store up to 3 petabytes of data in a single cluster, and you can even replicate your data to a different availability zone for high availability. So it's basically got all of the enterprise features that you would expect, all in an easy-to-configure AWS console interface. So it's much easier than configuring all of this on your own. Now, the Amazon OpenSearch service is compatible with all of the standard Elasticsearch open-source APIs, and it integrates with data ingestion tools like Logstash for data collection and processing, and visualization tools like Kibana for search and data visualization, allowing you to create bar charts and line graphs and scatter plots, etc. It also integrates with CloudWatch for monitoring. And you can use CloudWatch Logs and Kinesis Data Firehose for ingesting data into the cluster. Or alternatively, you can use a Lambda function to respond to new data that is stored in S3, for example, or DynamoDB, and the function can process the data and stream it into your OpenSearch domain. So it really integrates with a whole lot of AWS services. So for the exam, just remember that OpenSearch is basically a fully managed Elasticsearch cluster, and it's based on the open-source Elasticsearch technology. It's fully compatible with industry standard tools. So you can use it with the standard Elasticsearch open-source APIs, and commonly used tools like Logstash and Kibana. It's integrated with lots of different AWS services, allowing you to ingest data directly from CloudWatch Logs and Kinesis Data Firehose, and you can also ingest data from S3 and DynamoDB using a Lambda function. And finally, in terms of use cases, it's great for real-time analytics of your data. For example, log analytics, analyzing your application monitoring data, security analytics, and complex business data analytics. So that's it for this lecture. If you have any questions, please let me know. Otherwise, I'll see you in the next lecture. Thank you.

OpenSearch Deployment Best Practices

Hello Cloud Gurus, and welcome to this lecture, which is going to cover OpenSearch Service Deployment Best Practices. And we'll begin with OpenSearch Master Nodes and Data Nodes, Deployment Best Practices, Multi A-Z Deployments, and finally, my exam tips. Now, as you know, an OpenSearch Service domain consists of master nodes and data nodes, and master nodes are responsible for the cluster management tasks, so they review the health monitoring data, track nodes in the cluster, maintain routing information, and update the cluster state after changes. Whereas, the data nodes are responsible for data and searches, and they store the data in shards, and a shard is simply a unit of storage and computation, and it's used to distribute the data and processing evenly across all the data nodes in the cluster. And the data nodes also perform searches, query requests, and CRUD operations, so create, read, update, and delete functionality. Now when you first create an OpenSearch Service domain or cluster, you will need to select an appropriate deployment type, and there are 3 different deployment types that you can choose from. So we've got production, where your cluster is distributed across multiple Availability Zones, and it uses dedicated master nodes for higher availability. You've got development and testing, which uses just one Availability Zone, and this is great for when you don't need high availability, because it's a very cost-effective option, and it's great if you just want to test something out. And then there's also the custom

deployment, and this is where you get to select your own configuration settings. But of course, in this lesson, we are talking about best practices, so we'll focus on the production deployment type, which is a multi-AZ deployment. So for a production deployment, they recommend 3 master nodes, and this gives you 2 backup nodes in the event of a failure. Everything should be deployed in multiples of 3 and distributed equally across 3 Availability Zones. So in this example, our master nodes and data nodes are distributed equally across the 3 Availability Zones, us-east-1a, b, and c. And this configuration with 3 master nodes and 3 data nodes is the minimum production deployment for applications that need the highest availability. And if you need to scale your cluster, you don't need to add any more master nodes, you simply add more data nodes. And for a production cluster, they recommend that you deploy your data nodes in multiples of 3, distributed equally across 3 Availability Zones, and that will give you the highest availability. So for the exam, just remember that for a mission-critical production environment, they recommend 3 dedicated master nodes. And the master nodes take care of management tasks like health checks, maintaining routing, and managing the cluster state. Data nodes should be deployed in multiples of 3, and of course, the data nodes store the data in shards and perform searches, query requests, and CRUD operations. And finally, for a mission-critical production environment, it is best practice to deploy across 3 Availability Zones, and distribute your data nodes and the master nodes, as well, equally across the 3 Availability Zones, and that will give you the highest availability. So that is it for this lecture. If you have any questions, please let me know. Otherwise, I'll see you in the next lecture. Thank you.

Demo: Creating an Amazon OpenSearch Service Domain

Hello Cloud Gurus and welcome to this lesson where we'll be creating an Amazon OpenSearch service domain. So from the console, we'll create our OpenSearch domain, and we'll select the production deployment type. So that is going to be a multi-AZ domain with dedicated master nodes for higher availability. Next, we'll explore the network configuration options that are available. And we can either make our cluster publicly accessible, or we can keep it private within our own VPC. But for this lesson, we are going to make our domain public, so it's going to have a public endpoint. And then, once everything has been installed and our domain is ready, we'll be able to connect to it using the public endpoint. And I would like you to note that after our domain has been created, we cannot then remove the public endpoint or make our domain private. It is not possible. So if you're ready to get started with Amazon OpenSearch, please join me in the console. So from the console, I'm going search for OpenSearch and select that. Create domain. I'm going to call it my-opensearch-domain. Scroll down. Under Custom endpoint, we're going to stick with the auto generated endpoints. Deployment type is going be Production, which utilizes multiple availability zones, and you also get dedicated master nodes for higher availability. So just make sure that Production is selected. On the version, we'll use the latest version available. And down here, you can enable compatibility mode, and this is the one to use if you have certain Elasticsearch clients that need to check the cluster version before connecting. Down here, under Auto-Tune, we'll accept the default to automatically apply performance tuning on our cluster. Then, under Data nodes, this is where we can select the EC2 instances that will make up our cluster. We're deploying to three availability zones, and this is the recommendation for production workloads with higher availability requirements. Then down here, under Instance type, this is where you can select the type of instances that you want to use. And by default, they want you to use large instances, but I'm going to change that to t3.small and select t3.small.search. Scrolling down, the number of nodes is going

be 3. Storage type will be EBS, and we'll stick with the minimum storage size of 10 GB. UltraWarm is a feature that allows you to store large amounts of read-only data, and it's backed by S3, and it's normally used when you have large amounts of log data that you want to analyze using OpenSearch. And we're not going to enable this, and don't worry. This is not covered in the exam. Down here under Dedicated master nodes, it's telling us that for production domains, three is recommended. And we can also specify the instance type, and I'm going to make this instance type match the one that we chose for the data nodes. So select t3.small.search. Scrolling down, we've got snapshots. And Elasticsearch version 5.3 and above has hourly snapshots only, so we will get hourly snapshots. Under Network, here's our network configuration, and we've got two choices available, VPC access or Public access. And it says here to enable VPC access, we use private IP addresses from your VPC, which provides an inherent layer of security. And then you control the network access within your VPC using security groups; whereas, with public access, you get an internet endpoint that is publicly accessible. And if you select Public access, then you should secure your domain using an access policy that only allows specific users or IP addresses to access your domain. So it's kind of reminding us that our domain is going to be easier to secure within a VPC, and VPC access is actually the recommended option. But to make things more interesting for this demo, I'm going to select Public access instead. So select Public access. And then down here, it's asking us to configure fine-grained access control. And for this, we need to create a master user. So select Create master user, and this is going to be the admin user for our OpenSearch domain. So I'm going to call my master user faye. Down here, we need to type in a master password, and the password must be at least eight characters long, one uppercase character, one lowercase character, one number, and one special character as well. Down here, we can use an existing identity provider for single sign-on, or we can use Cognito for authentication. And we can control access to our domain using a domain access policy. And for this demo, we are just going to allow open access to the domain so that we can access it easily later on. So select Only use fine-grained access control, and this is going to allow open access to the domain. And that will create the relevant access policy for you so that we can access the domain later on. And don't worry because all of these settings are out of scope for the exam, so you don't need to worry too much about them now. Here's our encryption settings, and as we enabled fine-grained access control, that requires HTTPS, node-to-node encryption, and encryption at rest to be enabled. So by default, encryption in transit and encryption at rest have both already been enabled by default, and we cannot change those settings. Scrolling down, we'll just stick with the default AWS-owned encryption key. Then just scroll down to the bottom and hit Create. And there is our cluster. Under Domain status, you'll see the that it's still loading, and you can expect it to take around 10 or 15 minutes to create everything. And then our domain status will change from Loading to Active as soon as the domain is ready to use. So, now is a great time to go and have a cup of tea, take a break away from the screen, and in a few minutes, we should be good to continue. A few minutes later and here is our cluster. And if it's not showing as active after about 10 or 15 minutes, then just refresh the browser. And there we go. Domain status is Active, and cluster health should be showing as green. So now, scroll down and select Instance health. And here are our instances. So we've got three master nodes and three data nodes. And then if we come back up to the top of the screen, on the right here is our domain endpoint, and this is our public endpoint. So let's try and log into our domain. Just click on your public endpoint, and we'll provide the username and password that we created earlier. So mine was faye, hopefully you've remembered your password, and sign in. And there we go. So that all looks great. We've connected to our cluster using our public endpoint. So now, let's head back to our cluster settings. Scroll down and select Cluster configuration and select Edit. And here are all of our cluster settings. And if you scroll

down, you can review all the settings that we selected earlier. And you will see that there is no option to remove the public endpoint. So after our cluster has been created, there is simply no option to do that. Under Custom endpoint, we've got the option to define a custom endpoint, but we can't get rid of the public endpoint altogether, and we cannot make our domain private now that it is already public. And that's actually the same the other way around as well. So if we created our cluster in our own VPC and we made it private, then we wouldn't be able to go in later and add a public endpoint. And that is a really important point to remember for the exam. So once you've finished exploring, just scroll down to the bottom and hit Cancel. And then from this page, I'm just going to go in and delete my domain, and that is going to delete everything. So onto my exam tips. And just remember that when we first create an OpenSearch service domain cluster, we can make our cluster either publicly accessible, or we can keep it private to our own VPC. And if you created a domain with a public endpoint like this, and it's the same the other way around. If you create the domain in a VPC, you cannot then add a public endpoint later on. Instead, you will need to create a new domain with the correct network configuration settings and then migrate your data to the new domain. So that is it for this lesson. If you have any questions, please let me know. Otherwise, I will see you in the next one. Thank you.

Demo: Configuring Static Website Hosting Using S3

Hello Cloud Gurus and welcome to this lesson where we'll be creating a static website hosted in S3. And we'll begin by creating an S3 bucket, and we'll enable public read access. So first of all, we're going to edit the block public access settings. And then secondly, we'll create a bucket policy allowing public access. Next, we'll upload a file, and we're going to upload an index.html file that's going be our little website. Then, we'll enable static website hosting in the bucket properties. And then finally, we'll test that we can access the web page using the bucket website endpoint. So if you're ready to configure static web hosting in S3, then please join me in the AWS console. So from the AWS console, search for S3. Make sure that you've selected Buckets on the left-hand side and select Create bucket. I'm going to call it mystaticwebsite, and I'm going to add some random numbers onto the end of that name. Next, scroll down until you find the block public access settings. And I want you to deselect all of the block public access settings so that we can make this bucket publicly accessible. Scroll down and acknowledge your settings. Then, go right down to the bottom and select Create bucket. Next, we need to add our bucket policy. So select your bucket name. Select Permissions. Scroll down until you get to the bucket policy. Select Edit, copy your bucket ARN, then select the policy generator. The type of policy is going be S3 Bucket Policy. Effect is Allow. The principal will be star because we're allowing anonymous access. AWS service is Amazon S3. The actions will be GetObject and GetObjectVersion, and that's going to allow anonymous read-only access to our bucket. Under the Amazon ARN, just paste in the ARN of the bucket that you've copied. But remember that you need to add forward slash and star to the end, and that's going to ensure that this policy is applied to all of the objects within the bucket. Once you've done that, select Add statement. There's our statement. Generate policy. Copy your policy. Come back to S3. Paste your new policy. Scroll down, and we can just ignore this message, it's not going to affect what we're doing, and Save changes. And if that's worked correctly, you'll see your bucket policy here. And then, at the top, it will also say that your bucket is publicly accessible. So next, we're going to upload a file, and you will find a link to this file in the resources for this lesson. But you can just download it from your local machine from our Git repository for this course. And it's this file here called index.html. So that is the file that we're going to be uploading, and here it is. It's

just a very simple index.html web page. So once you've downloaded that file to your local machine, I want you to upload it to the S3 bucket. So select Objects, Upload, Add files. Here's my index.html on my local machine. Select Open and Upload. So that is our file uploaded, and the next step is to enable static website hosting in our bucket properties. So I'm going to close this page, select Properties, and you'll find static website hosting right down at the bottom of this page. So scroll down to the bottom. There it is, and it's disabled by default. So select Edit, Enable. The hosting type is going to be a static website. Down here, we need to specify the home or default page of the website, and it's going be index.html. The error document is optional, and this is the document that's returned if an error occurs. We don't have one of those, so I'm just going to scroll down to the bottom. Hit Save changes. And if that's been successful, if you scroll right down to the bottom again and find Static website hosting, down here you will find your bucket website endpoint. And this is the URL that we can use to access our web page. So if you click on your bucket website endpoint, you should see something like this if it's all worked correctly. And there we go. We are running a very simple website from S3. And the great thing about running a website this way is that you get all the benefits of S3, so the durability, high availability, and performance without having to manage any web servers of your own or any EC2 instances. And it's a really, really cost-effective way to run a static website and get up and running with your website really, really quickly. So for the exam, just remember that static web hosting is a really easy cost-effective way to host your static web content. To enable it, all you do is enable static web hosting in your bucket properties. And remember that you also need to enable public access on your bucket. So you'll need to disable all of the block public access settings in order for it to work. And you will also need to configure a bucket policy, allowing public read access for the objects within your bucket. And we did that to allow public read for our index.html file. So that's it for this lesson. If you have any questions, please let me know. Otherwise, please join me in the next lesson. Thank you.

Demo: Leveraging Presigned URLs with S3

Hello Cloud Gurus and welcome to this lesson where we'll have a go at creating a presigned URL. And we'll begin by creating an S3 bucket, and we're going to upload a file to our bucket. Next, we'll create a presigned URL. And to do that, all we need to do is select the object. And then from the Actions menu in S3, select Share with a presigned URL. It's really simple. Next, we're going to test our presigned URL, so we'll try and access the file using the URL. And then finally, just to show you that we haven't made our file public, we're also going to try and access the file anonymously using the S3 object URL. And we should see an Access Denied message. So if you're ready to get started, then please join me in the console. So from the AWS console, search for S3. Select Create bucket. I'll call it mysharedfiles-faye. Scroll down and Create bucket. Next, I'm going to upload a file from my local machine, and it can be any file that you have to hand. So select your bucket, select Upload, Add files, and I'm just going to select a picture of me and Lars. Select Open and Upload. After you've uploaded the file, select Close and then select your file using the checkbox on the left. Then select Actions and Share with a presigned URL. At this point, we need to decide when we want the URL to expire, and it can be valid for up to 12 hours, but I'm going to set it to 5 minutes. And after you create the presigned URL, it's automatically copied to your clipboard. So select Create presigned URL. We've got to a message here saying that it's been created and copied to our clipboard. So now, we can open a new browser tab, paste in our URL, and hit Enter. And there is our file. So we're accessing the file using the presigned URL. And you might be wondering have I just made the file public. Well, no, we haven't because it will only be accessible in this way

to the people that you've shared the URL with and only for the time that the URL is valid. So in this case, for the next 5 minutes. So if we try to access the file anonymously using the S3 object URL, will we be able to do that? Well, let's find out. Come back to S3. We're going to select our object. Here's our object URL. If we click on that, we should get the Access Denied message, and this is because our file is still private. So for my exam tips, just remember that you can use a presigned URL to provide temporary access to S3 objects that are actually private. Anyone who you give the presigned URL to will be able to access the object, and you can create a presigned URL using either the AWS CLI or SDK, but the easiest way is using the AWS console. When you create the presigned URL, you can set an expiry after which it will no longer be valid. And if you remember, we set ours to 5 minutes. And when you share the URL, anyone who has the URL will be able to use it until it expires. So that is it for this lesson. Any questions, please let me know. Otherwise, I'll see you in the next lesson. Thank you.

Restricting S3 Accessibility with IP Addresses

Hello Cloud Gurus, and welcome to this lecture, which is going to cover restricting S3 accessibility using IP addresses. And we'll begin with, why would we restrict access in the first place? We'll take a look at an example bucket policy that can be used to restrict access, and my exam tips as well. So why would we restrict access to S3 in the first place? Well, imagine you've got an S3 bucket containing confidential financial information, and only the finance team in your company should be able to access this bucket. Well, you can configure this using a bucket policy. So you can restrict access to your S3 bucket so that only particular IP addresses can access the objects in your bucket. And this could be either an external or an internal IP address range. And we just set up the bucket policy to only allow access from computers that are on the address range that we specify. And in this example, it's the 10.0 .12 .0 /24 network So let's take a look at an example bucket policy that we can use to do exactly this. So in this policy, the effect is going to be Allow the principal is everyone, the action is all S3 actions, the resource is the ARN of our bucket, and then is this condition section at the end, which is the important bit. So the condition states that the source IP of all requests must be from this IP address range. So, the 10.0 .12 .0 /24 address range. And this is going to allow our finance team to access the bucket. But then beneath that, it's also stating that any requests will not be allowed from the 54.240 .143 .188 IP address. So we can explicitly state that we do not allow requests from a specific IP address or a range of addresses. So for my exam tips, just remember that we can use a bucket policy to restrict access to only a specific IP address range. For example, let's say you only want to give access to your finance team users who are using computers on the 10.0 .12 .0 /24 subnet. And we can also deny requests as well. And we do this using the condition statement in our bucket policy. This will apply to all objects in our bucket. So that's it for this lecture. If you have any questions, please let me know. Otherwise, I will see you in the next lecture. Thank you.

Introducing S3 Inventory

Hello Cloud Gurus and welcome to this lesson, which is going to introduce you to a very cool feature of S3, and it's called S3 Inventory. We'll begin with what is S3 Inventory? What's S3 Inventory used for? I'll give you some example use cases. We'll go through how it works and finish off with my exam tips. So what is S3 Inventory? Well it's a service that creates an inventory of your S3 bucket, and it produces a flat file. And this file consists of an inventory of all the objects that are stored in your S3 bucket, including the object metadata. Supported formats include CSV, or

comma-separated values, Apache ORC, or optimized row columnar, and also Apache Parquet. And those are just three different formats that you can select for your inventory file. And you can schedule an inventory to be created on a daily or weekly schedule. So it's going to create it automatically for you. What can we use S3 Inventory for? Well, it's really great for helping you to understand how you are using your S3 storage in a given bucket. You decide which metadata you want to include in the report, for instance object size, when the object was last modified, if multipart upload was used, and if the object was successfully replicated and whether or not it was encrypted and which encryption method was used as well. So if you're wondering which objects in this bucket have been encrypted using SSE-KMS, think S3 Inventory. If you're wondering which objects in this bucket have successfully replicated and which have failed, think S3 Inventory. And if you're wondering which storage class is being used for each object in this bucket, then I want you to think S3 Inventory. And if you see anything in the exam asking you how to find out the answers to any of these kind of questions, then they are asking you about S3 Inventory. So, how does it work? Well just imagine that you have an S3 bucket just like this one containing lots of objects. Well, first of all, you will need to create another S3 bucket, and that's going to be where the inventory file is going be stored after it's been generated. Then, in the properties of the first first bucket, in the finance data bucket, we need to configure our S3 inventory to be produced. And it's going to create the inventory and then store it in our inventory data bucket. Now this might be sensitive data, so we can also have S3 encrypt the resulting file using either SSE-S3 or SSE-KMS. So, that's how it all works. Let's take a look at my exam tips. So with S3 Inventory, you get an inventory report, and we can use this report to help us understand how we're storing objects in a given S3 bucket. Supported file formats include CSV, Apache ORC, and Apache Parquet. Example use cases include, for instance, let's say you want to find out which objects have been encrypted, which classes of storage am I using, or which objects have successfully replicated. Basically, anything that you can get from the S3 object metadata, well, that's the kind of thing that you can use S3 Inventory for. The resulting report is stored in S3, and it can be encrypted using either SSE-S3 or SSE-KMS. So that is it for this lesson. Any questions, let me know. Otherwise, please join me in the next lesson. Thank you.

Demo: Introducing S3 Inventory

Hello Cloud Gurus and welcome to this lesson where we'll be exploring how to use S3 Inventory. And we'll begin by creating two S3 buckets. The first one, my-inventory-data, is going to store our inventory. And the second one, my-finance-data, is going to store our S3 objects. Next, we'll upload some files to the my-finance-data bucket. We'll then configure our inventory, and we'll review the available options. And then we're going to review an example inventory that I'll show you from my own account because it takes up to 48 hour for the inventory to appear, so we're just going to take a look at one that I configured a few days earlier. So if you're ready to get your hands dirty with S3 inventory, I'll see you in the console. So from the console, search for S3. Select Create bucket. And we'll call our bucket my-inventory-data and then add some random numbers on the end. Scroll down to the bottom and Create bucket. Now, let's create our second bucket. So select Create bucket, call it my-finance-data, and then just add some random numbers on the end. Scroll down to the bottom and Create bucket. So now I'm just going to upload some random files to the my-finance-data bucket. So select that, Upload, Add files, and I'm just going to upload some random files to the bucket. Select Upload. Close that down. And now we are ready to configure our S3 Inventory. So select the Management tab for your bucket, scroll down, and this is where we can create our S3 Inventory configuration. So select Create inventory configuration. We'll give it a

name. Under Inventory scope, we'll just stick with the current version of our object. Under Report details, this is where we need to define the destination bucket. If we select Browse S3, we can select my-inventory-data bucket, so select that one. It automatically creates a bucket policy to allow S3 to place data in our bucket. Scrolling down, this is where you can select the frequency, so how often the report is going to be generated, and we can select daily or weekly. But just notice that the first report is going to be delivered within 48 hours, so it's not instant. Down here we can select the output format. I'm going to stick with CSV, which is comma-separated values. Do we want the configuration to be enabled to publish inventory reports? Make sure Enable is selected. Down here, we can enable server-side encryption if we want to using either SSE-S3 or SSE-KMS. I'm just going to disable that. And then down here, these are the metadata fields that you can include for each listed object in the report. So I'm going to select a few different metadata fields. So just select the ones that you want and select Create. Now, as it can take up to 48 hours to create the very first inventory, we don't want to wait that long. So let me show you one that I configured in my own account so you can see what it's going to look like. So here I am in my own account, and I've got the two buckets, my-finance-data and my-inventory. So I'm going to select that. This is where I've stored the S3 Inventory files. I'm just going to keep drilling down until I get to this data folder, and this is where my inventory is going to be stored, and I've actually got three different events because it's created an inventory on a daily basis since I configured my S3 Inventory. So let's take a look at the latest one. The inventory is stored in a zipped file, so I'm just going to go ahead and download it to my local machine. And then I should be able to find it in my Downloads directory and open it up. Now if you happen to see a message like this telling you it's an inappropriate file format, we need to find the file in our Downloads directory, and there it is. Then, I'm going to type gunzip and the name of my file and hit Enter. And now we should be able to open the file, and there we go. That is my S3 Inventory report. So here's the name of my bucket. Here's the name of the file. We've got a file size, a timestamp, when the file was last modified. Here's the storage class. This field here is related to the replication status because that's one of the fields I selected in my configuration. And then this field here is related to the encryption status, so it's not using server-side encryption. And I've attached a copy of this file to the lesson so you can take a look at the report for yourself. So that is S3 Inventory. It's pretty easy to set up and start working with. So just remember, for the exam, it's an inventory report stored in S3, and it's used to help you understand how you're storing objects in an S3 bucket. You can decide which metadata you want to include in the report, for instance object size, last modified, multipart upload, replication status, and encryption status. And you can configure it to run on a schedule. So you can use a daily or weekly schedule, but just be aware that it takes up to 48 hours for the very first report to appear. So that is it for this lesson. Any questions, let me know. Otherwise, please join me in the next lesson. Thank you.

Demo: Using AWS Config with S3

Hello Cloud Gurus and welcome to this lecture where we're going to be getting our hands dirty using AWS Config with S3. And we'll begin by navigating to the AWS Config console Next, we'll review the AWS Managed Rules for S3. And the main ones to take note of are the s3-bucket-public-read prohibited and s3-bucket-public-write-prohibited. And these are two of the most important rules that you will need to be aware of for S3. So if you're ready to take a look at the AWS Config rules for S3, then I'll see you in the console. So here I am in the console, and I'm going to head to Services and then select Config under Management & Governance. I'll run the 1-click setup, and we'll accept all the defaults and hit Confirm. And come across to Rules. We'll add a rule,

and we're going to use the AWS Managed Rules. And then down here, in the search box, I want you to search for S3. So type s3-, and this will filter for all of the S3-related managed rules. And there are loads of different rules, and using a Config rule is a great way to check that all of your S3 buckets are configured exactly as you want them. So let's say, for example, that you want to check that public read or write access is prohibited in all your S3 buckets. Well, we can search for s3-bucket-public, and these two checks that appear will check that your buckets do not allow public read or public write access. And there's also a check for bucket versioning. So if we search for s3-bucket-versioning, this check is going to check whether versioning is enabled for our S3 buckets. And there's another really good check for server-side encryption. So if we search for s3-bucket-server, it will bring up this check, which is going to check that your S3 bucket either has default encryption enabled or that it's got a bucket policy, which is going to deny PutObject requests without server-side encryption. And then if we want to enable any of these rules, we can just select the rule, hit Next, scroll down to the bottom, hit Next, and add the rule. And once our rule is added, Config will start checking for compliance with that rule. And it's as easy as that to use AWS Config to check the configuration of your S3 buckets. And once you've finished exploring, just make sure you go back and disable Config. So head to Settings, edit your settings, and disable recording. And that will ensure that you don't get charged anything for AWS Config. Onto my exam tips, and I hope you agree that it's really easy to use AWS Config to check the configuration of your S3 buckets, and there are loads of different managed rules within AWS Config that will check the configuration on your bucket. But the main ones that I want you to remember are s3-bucket-public-read-prohibited and s3-bucket-public-write-prohibited. And these managed rules will check that your Amazon S3 buckets do not allow public read or public write access. So that's it for this lecture. If you have any questions, please let me know. Otherwise, I will see you in the next lecture. Thank you.

Overview of Storage Gateway

Hello Cloud Gurus and welcome to this lesson, which is an overview of Storage Gateway. We'll begin with what is Storage Gateway? We'll then take a look at the different types of Storage Gateway that are available. So we've got the S3 File Gateway, FSX File Gateway, Volume Gateway, and Tape Gateway. And then we're going to finish off with my exam tips. So what is Storage Gateway? Well it's an on-premises appliance, which is installed in your own data center, and it allows you to integrate your on-premises environment with AWS-based storage in the cloud. And it's basically a cost-effective alternative to having a massive on-premises storage solution. And if you've ever had to provision a new storage area network or even purchase new physical storage for an existing storage area network in your data center, then you will know that it can get expensive pretty quickly, and you often end up having to purchase way more than you can actually use. And that is why you should consider Storage Gateway. So with Storage Gateway, a physical or virtual appliance is installed in your data center. If you're using a Storage Gateway virtual appliance, then it can run on VMware ESXi, Microsoft Hyper-V, or on Linux KVM, and this allows your on-premises systems to then integrate seamlessly with storage in AWS. For instance, S3, Glacier, EBS volumes, and even FSX for Windows File Server, and that's a technology that's used to share file systems with Windows Servers. And this basically allows you to implement a hybrid cloud solution for your storage. Now there are a few different types of Storage Gateway available, and you'll need to understand the differences for the exam. So first of all, we've got File Gateway, which is a low-cost alternative to on-premises storage. Next, there's FSX Gateway, and this is used for sharing files stored in Amazon FSX for Windows File Server. There's also Volume Gateway, and there are two

different types of volume gateway that we're going to explore in more detail in a moment. But basically, you can use Volume Gateway to either store your backups in S3, or you can use S3 as your primary storage. But we'll cover that in more detail in a moment. And then finally, we've got Tape Gateway, which is a virtual tape library solution, and it provides virtual tape backups in the cloud. So now, let's take a look at all of these in a little bit more detail using some examples. And we'll begin with File Gateway. Imagine you have your own data center. With File Gateway, your files are stored as objects in S3, and they're accessed using an NFS or SMB mount point, just like a file system that is backed by S3. And of course, the NFS protocol is used by Linux systems to access file systems over the network, and the SMB protocol is the equivalent used by Windows systems. And the great thing about this is that you get all the benefits of S3, like bucket policies, S3 versioning, lifecycle management, and encryption, etc. And if all you need is storage, then it's a low-cost alternative to using on-premises storage. Next, there's FSX Gateway. So here's your data center. And with FSX Gateway, your files are stored in Amazon FSX for Windows File Server, and they're accessed using the SMB protocol. And Amazon FSX for Windows File Server is simply a fully managed shared file system, which is used for Windows applications and home user directory. It's AD-integrated, and it's great for shared file data for Windows servers. Next, we have Volume Gateway. And if you remember, there are two different modes. Now the first one is stored mode. So here's your data center. And with Volume Gateway stored mode, you store all of your data locally and use AWS storage as a backup. It uses the industry standard iSCSI protocol, and this is great because it provides durable, offsite, asynchronous, and point-in-time backups in the form of EBS snapshots, which are stored in S3. So it's a great offsite solution for point-in-time backups. Onto Volume Gateway cached mode. Here's your data center. And with Volume Gateway cached mode, S3 is your primary storage. Only your frequently accessed data is cache locally by your Storage Gateway using on-premises storage. And once again, data is accessed using the industry standard iSCSI protocol. And the great thing about this is that you only need enough local storage capacity to store your frequently accessed data because your entire dataset is stored in S3. So you get low latency access with no large investment in on-premises storage. And then finally, we have Tape Gateway. Here's our data center. And Tape Gateway does what it says on the tin. The clue is in the name. It's a virtual tape library. So it's a virtual tape library, which enables low-cost data archiving to Glacier. It integrates with existing tape backup software like NetBackup, Backup Exec, or Veeam, etc., which connects to the virtual tape library using the iSCSI protocol. This means that you don't need to invest in your own tape backup infrastructure. Instead, we're using virtual tapes, which are stored either in S3, in Glacier Flexible Retrieval, or in Glacier Deep Archive. So it's a great low-cost backup solution, especially if you've already invested in enterprise backup software. So for the exam, you will need to understand the differences between the various options available. So File Gateway is where you access files stored on S3 using either NFS or SMB. FSX Gateway is where you access files stored in Amazon FSX for Windows File Server using SMB, and this is ideal for Windows applications and home directories. With Volume Gateway stored mode, your entire dataset is stored on site, and it's backed up to S3 as EBS snapshots and accessed via iSCSI. With Volume Gateway cached mode, your entire dataset is stored in S3, and only your frequently accessed data is going be cache on site. And once again, it's accessed via iSCSI And then Tape Gateway should be pretty easy to remember. And this is where your backups are archived to S3 or Glacier, and it can be used with or without your own backup application and accessed using iSCSI. So that is it for this lesson. If you have any questions, please let me know. Otherwise, I'll see you in the next lesson. Thank you.

Introducing AWS Backup

Hello Cloud Gurus and welcome to this lesson, which is going to be an introduction to AWS Backup. We'll begin with what is AWS Backup? How it works. We'll take a look at what is a backup plan? And we'll also cover up my exam tips as well. So what is AWS Backup? Well, it's a managed service that provides backup and restore services for data that is stored in various data stores, for instance in compute, storage, and databases. It's integrated with loads of different AWS services, and it allows you to manage your backups for multiple AWS services centrally. For instance, you can centrally manage your backups for data stored in S3, FSX file systems, EC2 instances, EBS volumes, RDS databases, DynamoDB tables, and even VMware on AWS and VMware on-premises systems as well. And I've included a link to all of the different services that are supported. So this is where you'll find the latest on what is supported with AWS Backup. And it protects your data by mitigating the risk of accidental deletion, data corruption, or data loss. So how does it work? Well, first of all, you need to create a backup plan, and we use a backup plan to define how and when we want to back up our data and the retention period as well. When the backup plan is ready, we can then go ahead and assign resources to our backup plan. And that means that we need to identify the resources that we want to back up using this plan. And then after we've done that, AWS Backup will do the rest. So it creates and retains backups based on what you've defined in your plan. And in addition to planned backups, you can also perform on-demand manual backups as well with no backup plan required. Now when you create a backup plan, it's going to consist of one or more backup rules. And a backup rule includes a schedule, so it has a frequency. Let's say, for example, you're backing up once a day, and that is going to determine your recovery point objective. So if you're backing up once a day at the same time each day, then your recovery point objective will be to restore data from the latest backup, which should never be more than a day old. You also need to define what is your backup window. So when would you like the backup to start and complete? You've also got lifecycle rules, which define when to move the backup to cold storage. They also define the retention period. And when the retention period is over, then the backup is going to be deleted. The backup vault defines where the backups are going to be stored. And by default, AWS Backup will create a vault for you, and it's simply a container for all of your backups. And you can also define tags to be added to each backup, which is also known as a recovery point. So the tags are added when your recovery point is created. So you might want to add a tag like monthly backup or daily backup or something meaningful to you to help organize everything. So for my exam tips. AWS Backup is a managed backup service, allowing you to centralize your backups for different AWS services like S3, FSX, EC2, EBS volumes, RDS databases, DynamoDB tables, VMware on AWS, VMware on-premises, and many more. To get started, you need to create a backup plan, which consists of one or more backup rules defining things like a schedule, lifecycle rules, the backup vault, and any tags that you want to apply. After you've created the backup plan, you can go ahead and assign your resources, and then AWS Backup will do the rest. So that is it for this lesson. Any questions, let me know. Otherwise, please join me in the next lesson. Thank you.

Demo: Using AWS Backup

Hello Cloud Gurus and welcome to this lesson where we'll be getting our hands dirty using AWS Backup. And we'll begin by creating an EFS file system, and this is the resource that we're going to be backing up using AWS Backup. Next, we'll configure a backup plan. And once we've done that, we'll be able to assign our resources and review the options that are available. Now it takes a while for the new backup plan to actually run a backup. So for that reason, we're going to run an

on-demand backup as well, and we'll be able to view the progress of our on-demand backup in the AWS console. So if you're ready to get started with AWS Backup, please join me in the console. And please note that you will not be able to do any of this using our AWS sandbox or cloud playground. You will need to do it using your own AWS account. So first of all, I'm going to search for EFS and Create file system. I'm going to call it my-efs-file-system and select Create. So there is our file system created. Next, I'm going to search for backup. Select AWS Backup. And we need to create a backup plan, so select Create backup plan. We're going to start with a template because they provide some really good template options. Down here, we'll select the daily 35-day retention template. We'll give our backup plan a name. I'll call it my-backup-plan. Then down here, this is where we can configure the backup rules, and our template has already added this rule called DailyBackups. So select that rule, and this is where we can edit the options. Here's our backup rule name. We can store the backups in our default backup fault, or you could go ahead and create a new one. I'm going to stick with the default. And if you remember, the backup fault is simply a container for all of your backups. Backup frequency is going be daily, but you can change it to one of these other options as well. For some services, there's an option to create continuous point-in-time backups with 1-second of precision, going back to a maximum of 35 days. And this is available for these services. Down here is the backup window so this is when the backups can happen. And the default is set at 5 a.m. UTC with a backup window lasting for 8 hours, but it is customizable. Down here, we can set the backups to transition to cold storage after a set time period. So if we set it to Transition, let's say after 7 days, you will also need to change the retention period because if you transition to cold storage, you must store it in the cold storage for at least 90 days after transitioning. But if you set it to never transition to cold storage, then you can set your retention period to whatever you want. And the default is 35 days, so I'm going to set it to 35 days. Down here, we can optionally create a cross-region copy to have another copy of our backup saved in a different region. And then down here, we can add a tag to our recovery points to help us manage them. And I'm just going to add a tag of Backup Type, and the value is going be Daily Backup. So once you've selected your options, we'll save backup rule. Then here, under Advanced backup settings, there's the option for application-consistent snapshots, and this is for Windows applications that are using the Volume Shadow Copy service. And then down here, it's just reminding us that we can add resources and more rules to the backup plan after it has been created. So for now, select Create plan. So now we've created our plan. And now, at this point, it's going to allow us to assign our resources. So we need to configure our resource assignment. I'll give it a name. I'm going to call it my-resource-assignment. It's going to create an Identity and Access Management role that will let AWS Backup manage your recovery points or the backups that it creates. So I'm going to stick with the default role. Down here under Resource selection, this is where we define the resources that we want to back up. So I'm going to select Include specific resource types. Then under Select specific resource types, we're going to select EFS. And then under File system IDs, we want to deselect all file systems and just select your EFS file system that we just created. Down here, we can exclude specific resource IDs from our selected resource types, and we can also refine our selection using tags as well. But once you've selected the EFS file system, you can go ahead and select Assign resources. So that is our backup plan and our backup rule created. And then the last thing I wanted to show you is that you can also run an on-demand backup. So on the left-hand side, if you select Protected resources, Create on-demand backup, select your resource type, it's going be an EFS, select the file system ID, there it is. If we create the backup now, it's going to start within 1 hour. I'm not going to transition to cold storage. I'll set the retention period to just 1 day. We'll use the default backup vault and the default IAM role and Create on-demand backup. Once the backup has started,

you can view the status of all your backup jobs here in this Jobs section. So this is the backup job I just created, and here's a couple of jobs that I did earlier on today. And it will just take a few minutes to complete the backup job even though there's no data on our file system. So if you use the Refresh button, it should refresh the status of your backup job. And then, when the backup job is completed, it will show as Completed in the status over here. And you can also head to the backup vault. Come to your default backup vault. And this is where the recovery point is going to appear. And recovery point is just their name for the completed backup that you can use to recover your data. Now if you have done all of this in your own AWS account, please do remember to delete everything after you've finished so that you don't get charged. And to do that, first of all, delete your on-demand backup. So select it and select Delete. Then, in your backup plan, select my-backup-plan. And you will need to delete the resource assignment before you can delete the backup plan itself. So delete your resource assignment. And then once you've done that, you can delete your backup plan. And then finally, head back to EFS and delete your EFS file system as well. So onto my exam tips for using AWS Backup. Just be familiar with the process of configuring backups. So first of all, you need to create your backup plan and add one or more rules that are going to define the backup. After creating the backup plan, you can then assign the resources that you want to back up. And after the backup job is complete, it's going to appear as a recovery point in the backup vault. And you can also view the status of your backup job in the Jobs section of the console. So that is it for this lesson. Any questions, let me know. Otherwise, please join me in the next lesson. Thank you.

Section Review: Storage and Data Management Summary - Part 1

Hello Cloud Gurus and welcome to this lecture, which is part 1 of the storage and data management summary, beginning with S3. And just remember that S3 is object-based storage, and it allows you to upload files like images, text files, code, or documents, pretty much any kind of file that you can think of. But it is not suitable to install an operating system or run a database on. Files can be from 0 bytes to 5 TB in size. And with S3, you get unlimited storage. So the total volume of data and the number of objects that you can store is unlimited. Files are stored in buckets, and S3 is a universal namespace, which means that bucket names must be unique. And this is what an object URL looks like. And in my example, my bucket name is fayecloudguru, my region is us-east-1, and my object is Ralphie,jpeg. And successful CLI or API uploads will generate an HTTP 200 status code. An S3 object has the following attributes. So the key, which is the object name, for example Ralphie.jpeg. The value, which is the data itself. A version ID, which allows you to store multiple versions of the same object in the same bucket. And metadata, which is data about the data that you are storing, for example the content-type. Remember the use cases for the different classes of S3 storage. So S3 Standard is suitable for most workloads. Infrequent Access is for long-term, infrequently accessed, critical data. You pay less to store your data, but you do pay a small fee every time you access the data. One Zone-Infrequent Access is for infrequently accessed, non-critical data, and they are storing your data redundantly, but only in one single availability zone. So there is reduced availability for this option. Glacier is designed for long-term data archiving for data that occasionally needs to be accessed within a few hours or minutes. And there is a minimum storage duration of 90 days, and this is for archiving data that you will access a few times a year. Glacier Deep Archive is for archiving rarely accessed data with a default retrieval time of 12 hours. The minimum storage duration is 180 days, and this is great, for example, for financial records that you need to keep for regulatory purposes, and let's say you need to keep them for many, many years.

And you may never even need to access these files, but you still need to keep them in case you need them. And then finally, we have S3 Intelligent Tiering. And this is the one to go for if you have unknown or unpredictable access patterns. When you create an S3 bucket, it is private by default. So S3 buckets and objects do not allow public access by default. We can use bucket policies and access control lists to enable able public access to our objects in S3. And with S3, you can store any kind of file that you can think of like photos, videos, code, documents, or text files and then use access control lists and bucket policies to control who can access your files. S3 lifecycle policies ensure that you are using the most cost-effective option to store your objects in S3. They allow you to transition your objects to infrequently accessed storage or to Glacier based on the rules that you configure. And the lifecycle rules are based on the object's creation date, and you can also set an expiry date for objects that you want S3 to delete after a certain time period has elapsed. S3 supports a few different types of encryption, including SSL, TLS, or transport layer security, which is what we are using when we access S3 over HTTPS. And that is known as encryption in transit. For encryption at rest, which is server-side encryption, there's SSE-S3, which uses S3-managed encryption keys, SSE-KMS, which uses an encryption key managed in KMS, and SSE-C, which uses a customer-provided encryption key. And client-side encryption is also supported, and this is where you encrypt the files yourself before you upload them into S3. With S3 versioning, multiple versions of an object are stored in the same bucket. A delete request does not delete the object, but instead applies a delete marker, and this protects your objects against accidental deletion. And you can still access all the previous versions of the file by specifying their version ID in the GET request. Another way to protect your data against accidental deletion is MFA Delete. And MFA Delete requires a valid code from your MFA device in order to permanently delete an S3 object. A valid code from your MFA a device is also required if you want to suspend or reactivate S3 versioning. And you cannot enable MFA Delete without S3 versioning. They work hand in hand, and this combination is designed to protect your data against accidental or malicious deletion of your version-controlled S3 buckets. Onto presigned URLs, and you can use a presigned URL to provide temporary access to S3 objects that are private. Anyone with the presigned URL will be able to access the object, and you create a presigned URL using the AWS CLI or SDK. And you can't do it using the AWS console. By default, a presigned URL expires after 1 hour. However, this is configurable, and you can set the expiry time of your presigned URL using the --expires-in option, and we specify the expiry time in seconds. We can restrict access to an S3 bucket based on an IP address range, and we use a bucket policy to do that. For example, imagine you only want to give access to your finance team users who are on the 10.0.12.0/24 subnet, and we use this condition statement to restrict access to only that IP address range. And finally, AWS Config provides loads of managed rules that allow us to continuously monitor the configuration of our S3 buckets. And the important ones to be aware of are these rules that check for public read and public write access. And to find the S3-related rules, just search in the managed rules section on the Config console, and you will find loads of rules relating to S3. So that's it for part 1. And if you're ready for part 2, then I'll see you in the next lecture. Thank you.

Section Review: Storage and Data Management Summary - Part 2

Hello, Cloud Gurus, and welcome to this lecture which is part 2 of the storage and data management summary. Beginning with static web hosting. And S3 static web hosting is an easy and cost effective way to host static web content. It's really easy to set up. You just enable static web hosting in your bucket properties, then enable public read access in your bucket permissions. And

finally, grant public read access for your objects in the object ACL. Moving on to Elastic File System, which is a fully-managed NFS file system, so it's a highly available and scalable shared file system for Linux-based workloads. A single EFS file system can be used by multiple EC2 instances, and it's great for applications which need to access shared files. For example, a shared configuration file or shared state information for your application. EFS supports lifecycle management, so files that have not been accessed recently can get moved to EFS Infrequent Access and it also supports encryption in transit and at rest, but you must enable encryption at creation time. You cannot go back and enable it later. So a single EFS file system can support many thousands of simultaneous connections. And by default, EFS file systems are configured with burstable throughput which means that throughput is determined by the amount of storage that you have. All file systems will get a minimum of 100 mebibytes per second of burstable throughput. And standard class file systems greater than one tebibyte in size can burst to 100 mebibytes per second per tebibyte of data stored. And don't worry, you won't be asked about all these feeds and speeds in the exam. The main thing to remember is that if you want to optimize performance for your application, then use provisioned throughput which allows you to define your own throughput capacity in mebibytes per second based on your own requirements. And as a rule of thumb, if you need more throughput than you are getting with burstable throughput mode, then switch to provisioned throughput mode. EFS Standard includes multi-AZ resilience and this means that you can create a mount target in each availability zone in an AWS region. Multiple mount targets are preferable because this avoids additional data access charges for hosts that are located in a different availability zone. And with multiple mount targets, you also get increased availability. So if there is a problem with the mount target in one availability zone, then instances located in the other availability zone will be unaffected. On to Athena. And Athena is a query service which supports standard SQL or Structured Query Language and it allows you to query your data stored in S3. It's really easy to configure. You just point Athena to the data that you want to query in S3, define a table schema, and then you are good to go to query your data using standard SQL. Amazon OpenSearch Service provides a fully-managed Elasticsearch cluster based on Elasticsearch open source technology. It's compatible with all of the standard Elasticsearch open source APIs like Logstash used for data ingestion and processing and Kibana which is used for search and data visualization. And it is integrated with loads of different AWS services. For example, you can ingest data directly from CloudWatch logs and Kinesis Data Firehose and you can also ingest data from S3 and DynamoDB using Lambda. And the use cases for OpenSearch are things like complex log analytics, application monitoring, security analytics, and complex business data analytics as well. For deployment best practices, a production OpenSearch service cluster should consist of 3 dedicated master nodes. And the master nodes offload the cluster management tasks like health checks and maintaining routing and cluster state. And you've also got the data nodes which should be deployed in multiples of 3. And the data nodes store the data in shards and perform searches, query requests, and CRUD operations, so create, read, update, and delete. And it is best practice to deploy everything across 3 availability zones. So you distribute your data nodes equally across 3 availability zones and this will give you the highest availability. And here is our example of a minimum production high availability deployment. So we've got 3 master nodes which is the recommended amount for a production environment and gives you 2 backup nodes in the event of a failure. Here's our data nodes and we're using 3 availability zones. We're deploying our data nodes in multiples of 3 and distributing everything equally across the 3 availability zones. And just remember that when we first create an OpenSearch Service domain cluster, we can make our cluster publicly accessible or we can keep it private to our own VPC. And if you create a domain with a public endpoint like this, then you

cannot later place it within a VPC and it's the same the other way around. So if you create the domain in a VPC, you cannot then add a public endpoint later on. Instead, you will need to create a new domain with the correct network configuration settings and then migrate your data to the new domain. So that is it for this lecture. If you have any questions, please let me know. Otherwise, I will see you in the next lecture. Thank you.

Reliability and Business Continuity

Section Introduction

Hello, Cloud Gurus, and welcome to this section on Reliability and Business Continuity. In this section, we'll cover the key components, services, and concepts that you'll need to understand, which are also covered in domain 2 of the SysOps Administrator exam guide. This particular section and domain focuses on building resilience and highly available architectures within AWS. And most importantly, your role as a SysOps administrator in the process. This being so, in this section, we'll cover elasticity and scalability, Auto Scaling in AWS. We'll discuss how to differentiate horizontal and vertical scaling. We'll discuss how ElastiCache plays a part in this, as well as RDS. Things like read replicas, multi availability zones, and much more. We'll discuss high availability, fault tolerance and disaster recovery, and then we'll conclude this section with our conclusion and exam tips. So if that sounds exciting to you, join me in the next lecture.

Elasticity and Scalability 101

Hello, Cloud Gurus, and welcome back to this lecture, Elasticity and Scalability 101. In this lecture, we're going to provide a high-level theoretical definition of exactly what is elasticity, as well as what is scalability by giving a real world example of the place that we all love, but hate at the same time, the gym. We'll also talk through some AWS services, which you can implement elasticity and scalability, which will give us a better understanding of how the two can be used to build resiliency within our architecture. And finally, we'll complete this lecture with some exam tips. So let's go ahead and get started. What is elasticity? So when we think about elasticity, I'd like for us to think of our friend over to our left here who has this rubber band. I'm sure many of you have been to the gym and you've seen a rubber band. And the cool thing about the rubber bands is that you stretch them as far out or as far in as you'd like, based upon your level of comfort or the level of resistance you may need for your stretch or for your workout. And the same thing is true from a building or infrastructure perspective. So elasticity in the technical sense is stretching or retracting our infrastructure based on our demand, which gives us the ability in times of high traffic to spin up and create more instances of our application, or spin down for when traffic is lower. And the really neat thing about this is from a cost perspective, this allows us to pay as we go, pay for the resources we need, and spend down and save some money when we don't need all those resources allocated. And when we think of elasticity, we typically think of elasticity being used over a short period of time. So over hours or even days, and this is really good for those businesses who have use cases which may have high traffic spikes, like a social network or something like that, which can spin up more resources when they're needed and spin them down when they're not. Now, let's define scalability. So when we think about scalability, let's think of going to the gym every day, lifting weights, whether it be a squat or a bench press, and every day, our ability to lift heavier and heavier weight increases. And that's exactly what scalability is, is building out our infrastructure in the technical sense to meet our demands, but over a longer term. So when we think about scaling our

infrastructure, this is typically used over longer periods of time, such as days, if it's a very fast growing application, weeks, even months or years. Now let's discuss how elasticity and scalability can be used within several AWS services. We'll get started by discussing EC2. So for elasticity within EC2, you can configure Auto Scaling to increase or decrease the number of EC2 instances which are available for the application. You can also configure Auto Scaling to also use spot instances as well. From a scalability perspective, you can increase the instance sizes, so making the instance larger to have more memory, more compute capacity, or you can configure from a scalability perspective, EC2 to use reserved instances. For Dynamo DB, its elasticity allows us to increase or decrease the IOPS based upon the traffic spikes. From a scalability perspective, DynamoDB has an unlimited amount of storage so it automatically scales. For RDS, unfortunately, there is no elasticity, as RDS cannot scale on demand. However, for scalability purposes, you can increase the instance size or add to the number of instances, which may cause downtime, which we'll talk out a little bit later in this section. For elasticity with Aurora, where you can autoscale up and down to meet varying demand on the database. However, from a scalability perspective, you'll need to modify the instance type as well as scale out with additional read replicas. So for our exam tips for elasticity and scalability, we'll just need to understand the difference between the two. So remember for elasticity, remember the rubber band? We need to be able to stretch out in those times we have high demand, and stretch back in for those times we have low demand, and always think of short term. So if you're asked the question about a shorter term scale solution, that would definitely be elasticity. And remember those things that pertain to elasticity, like Auto Scaling and the services which have Auto Scaling features. And then on the other side, we have scalability. And remember that we go to the gym every day to one day be able to reach our large goal. And our bar eventually gets heavier and heavier, and the same goals for the technical sense here that we wish to one day scale out our infrastructure over a longer period of time, which may be days, weeks, or months and that's scalability. So if you're ready to move on, I'll see you in the next lecture.

Introducing AWS Auto Scaling

Hello Cloud Gurus, and welcome back to this lecture on Introducing AWS Auto Scaling. In this lesson, we'll define the service by answering the question, what is AWS Auto Scaling? We'll also understand its key component, which is scaling plans. We'll talk through how to create scaling plans, as well as scalable AWS services. We'll wrap this lesson up with some exam tips as well. Let's go ahead and get started. Let's start by answering the question, what is AWS Auto Scaling? It's an AWS service, which allows you to configure plans to automatically scale your resources. So, this service is really useful in maintaining your application performance based on the requirements that you, the user, set. And this is how it does it. It allows you to configure and manage scaling of your resources through what's called a scaling plan. AWS Auto Scaling also uses dynamic and predictive scaling to scale your resources, and we'll talk about that in greater detail in just a moment, and it's also useful for scaling across multiple or several AWS services. Now, let's dive a little deeper into the key component of AWS Auto Scaling, which is scaling plans. First, AWS scaling plans is simply a set of directions for scaling your resources. It's within the scaling plan that you tell AWS Auto Scaling everything it needs to know to properly scale your application resources. And so within the scaling plan, you have what's defined as a scaling strategy, and the scaling strategy instructs AWS Auto Scaling on how to optimize your resources in your scaling plan for availability, cost or both. Within your scaling strategy, you can select 2 types of scaling you would like AWS Auto Scaling to conduct. One type would be dynamic scaling, which automatically adjust in response to resource

utilization through target tracking. So, for example, let's say you're using Elastic Container Service, and the service is at 75% of its utilization. You can set through dynamic scaling additional tasks to be kicked off when the service or the application reaches 75% of its compute resources. The other type of scaling is predictive scaling, which scales load by analyzing your resources' historical load through using machine learning and forecasting. Now, let's talk about the ways you can create your scaling plans. One way would be through CloudFormation scripting, which AWS Auto Scaling goes through available CloudFormation stacks and find scalable resources through existing CloudFormation templates. Another, which we'll demo in the demo to follow, is through selecting available EC2 Auto Scaling groups, and through this, you can select one or more existing EC2 Auto Scaling groups to be included in your Auto Scaling plan. And the final way is through tagged resources. Through tagged resources, you can search for scalable resources through using the tags that's been applied to them. Now, let's discuss AWS Auto Scaling in some scalable AWS services. So first, starting with EC2, AWS Auto Scaling is able to maintain an Auto Scaling group through launching or terminating EC2 instances within that Auto Scaling group. For DynamoDB, AWS Auto Scaling enables tables or secondary indexes to increase or decrease read and write capacity. For ECS or Elastic Container Service, it adjusts ECS services and tasks in response to the load variation. And for Aurora, it automatically adjust the number of read replicas within the cluster. Now, let's finish with some exam tips. So we'll definitely need to understand what are scaling plans, which we defined as a set of directions for scaling our resources. We'll need to differentiate between dynamic and predictive scaling. Remember dynamic is setting a threshold, for example, of when we would like additional resources to be allocated, versus predictive scaling, which uses forecasting and machine learning to re-allocate resources. And lastly, we'll need to understand how to create and maintain AWS Auto Scaling plan which we'll walk through in the demo to follow. So, if you're ready to walk through and get some hands-on learning of AWS Auto Scaling, join me in the next lesson.

Demo: Creating Auto Scaling Plans

Hello Cloud Gurus and welcome to this lesson on creating auto scaling plans. And we'll begin by setting up our EC2 instance configuration using launch templates. And within this step, we'll set all the information for our EC2 instances from AMI to security groups and our SSH key as well. And this is going let our auto scaling group and auto scaling plan know how to configure our instances. Next, we'll create our auto scaling group, which is going to launch two EC2 instances. And then after we've done that, we'll look at creating and maintaining an auto scaling plan. So I'll create an auto scaling plan, and you'll see that it is possible to create an auto scaling plan that is able to override some of the settings in our auto scaling group and create additional EC2 instances in our VPC based on the criteria that we define in the plan. So if you're ready to go ahead, please join me in the AWS console So from the console, search for EC2. Then, on the left-hand pane under Instances, select Launch Templates. Create launch template. We'll call it Test-ACG-Launch-Template. Scrolling down, the next thing we need to do is select our Amazon Machine Image, so select Amazon Linux, and we'll use the Amazon Linux 2 AMI. Instance type is going to be t3.micro. We'll create a new key pair and Create key pair. And I'm not going to specify a subnet, but down here under Security groups, we'll select the default security group. We'll keep the storage volumes as is and Create launch template. So that's our launch template created, and now we can create our auto scaling group. So if we go back over to the left-hand pane, scroll down, and select Auto Scaling Groups, Create auto scaling group. Call it Test-ACG-Auto-Scaling-Group.

Select the launch template that we just created, and it will automatically populate all the information in this section below. So here's my AMI, here's the key pair that we created, here's the launch template name, and my instance type is t3.micro. So now click Next. For our network settings, use the default VPC. And under Availability Zones, we're going to add us-east-1a and us-east-1b. And this means, of course, that our EC2 instances are going to be in different availability zones. After that, click Next. We have the option to create a load balancer, and we won't be using a load balancer for this demo. So scroll down to the bottom and click Next. Then we have the option to configure a group size and scaling policies, and this is where we set the desired capacity. Now if you remember, we're going to use this auto scaling group to launch two instances. So our desired capacity, is going to be 2. I'm going to set my minimum capacity to 2. And my maximum, I'm going to set it to 5, which is the total number of EC2 instances in our example architecture. So now go ahead and hit Next. We're not going to add SNS notifications, so hit Next. We can add a tag of CreatedBy and your name. Hit Next. We can review all the options. Then scroll down to the bottom and Create auto scaling group. So now that we can see that my auto scaling group was created successfully, the status is Updating capacity. My desired capacity is 2, minimum is 2, and maximum is 5. So let's give our auto scaling group a moment just to finish updating capacity, and you can refresh using this button. And if our status goes clear like this, we can go over and check our EC2 instances. So I'll open up the EC2 console in a new tab, and there's our two EC2 instances, and they are still initializing. And we should have one in us-east-1a and one in us-east-1b. So now, let's go on to our next step, which is to create an auto scaling plan. So in the search box, search for auto scaling, and we're looking for AWS Auto Scaling. On the left-hand pane, select Scaling plans and Create scaling plan. And the first thing we need to do is add resources to our scaling plan. And if we select Info, we can see that auto scaling plans work with more than just EC2 instances. So you can use them with Auto Scaling groups, with Aurora DB clusters, DynamoDB tables, and ECS services, Elastic Container Service, and Spot Fleet requests as well. So it's not just EC2 instances. Now we can identify our resources in three ways. First of all, we can search by CloudFormation stack, by tag, or by EC2 Auto Scaling groups. So select EC2 Auto Scaling groups. We can select our group down here. Hit Next. Call it Test-Auto-Scaling-Plan. Down here under Scaling strategy, we've got several options to choose from. So first of all, we can optimize for availability, which automatically monitors the CPU utilization at a 40% threshold. So after 40% CPU utilization, auto scaling is going to kick in. Alternatively, we can balance availability and cost. So keep the average CPU utilization of your auto scaling group at 50%. Or we can optimize just for cost and keep the average CPU utilization of the auto scaling group to 70%, and that will give you lower costs. But for this example, let's select Custom, and then we can set our own CPU utilization threshold. Next, we can see that we have predictive scaling, as well as dynamic scaling enabled. And I'm going to uncheck predictive scaling because this demo won't run for long enough to allow any forecasting to predict how our resources should be scaled. However, I am going to keep dynamic scaling enabled, and this is going to enable my capacity to be increased or decreased as needed. Under Configuration details, we can set our scaling metric based on average CPU utilization, average network in, or average network out. So we'll keep it to average CPU utilization, and I'm going to set my target value for 25. And then you should see on the right-hand side the graph has changed, now indicating 25%. And then the last thing we're going to do is replace external scaling policies. So that means that this scaling plan will override any policies attached to our auto scaling group. So now, we can hit Next. We don't need to configure any more settings. So hit Next and Create scaling plan. And once it's completed, click on your auto scaling plan. And scrolling down, we can monitor the CPU utilization right here. And if nothing's appeared yet, just wait a few moments, and it should begin to populate

these two graphs with your CPU utilization. So it's going to monitor the CPU utilization and then invoke the auto scaling plan if the usage gets above our target of 25%. So that is auto scaling plans. They're really easy to configure. And now let's go ahead and wrap up with my exam tips. So for the exam, just remember that auto scaling groups automatically scale EC2 instances through the EC2 service. However, auto scaling plans allow us to scale additional services, not just EC2. So you can scale Elastic Container Service, DynamoDB, and Aurora through tagging and CloudFormation, and you can scale auto scaling groups as well. Also keep in mind that auto scaling plans allow you to optimize for availability or balance availability and cost. We can optimize just for cost or select the custom option and define our own CPU utilization threshold to base our auto scaling decisions on. Well, that is all for this demo. And if you're ready to move on, please join us in the next lesson. Thank you.

Troubleshooting Auto Scaling Issues

Hello Cloud Gurus, and welcome back to this short lesson on Troubleshooting Auto Scaling issues. In this lesson, we'll briefly walk through different issues that may be preventing your resources or instances from Auto Scaling. We'll talk through the common issues that you can look for on the exam and close out with some exam tips. Let's go ahead and get started. Starting with our common issues with Auto Scaling, the first thing you may want to check is that the Auto Scaling group is not found. You also want to check that the Auto Scaling service is enabled within your account. This particular error is really common with those accounts that may be enrolled within AWS Organizations, or may have active service control policies preventing you from the use of the service. Also, check that your Auto Scaling config is working correctly. Now for some common issues with Auto Scaling as it relates to your compute and storage. First, you may have an invalid EBS device mapping. Also, check that the instance type is compatible within the Availability Zone. You also may be attempting to attach an EBS block device to an instance-store AMI. And lastly, you may want to check that your instance is supported within your Availability Zone, which is not likely, but that may be an issue as well. For issues around security, you may want to check that the associated key pair still exists, as well as the security group exists. For your exam tips, first, troubleshoot easy to Auto Scaling groups through the EC2 console and understand what errors may be coming from your EC2 Auto Scaling groups specifically. Additionally, monitor your Auto Scaling plans through AWS Auto Scaling, so definitely understand and be able to differentiate EC2 Auto Scaling groups and their troubleshooting mechanisms from troubleshooting the Auto Scaling plans which are auto scaled through AWS Auto Scaling service. And that's all for this lecture. If you're ready to move on, join me in the next lecture.

Vertical Scaling vs. Horizontal Scaling

Hello, Cloud Gurus. Welcome back to this lecture on vertical versus horizontal scaling. In this lecture, we'll be learning the basic understanding of vertical scaling versus horizontal scaling. We'll get started by differentiating the two through a real-world use case, which will be planning a trip for a couple of friends. We'll also talk about how vertical and horizontal scaling can be used through a couple of AWS services. We'll then wrap up with a couple of exam tips. So let's go ahead and get started. Let's imagine we're planning a trip for some friends across the country. We've sent out our text or emails, and 4 friends have confirmed to come along on the journey. As we're hosting the trip, we need to find a house that houses our 4 friends. So we find a house on Airbnb, and we book the house, perfect right? But then the unexpected or, if your friends are like mine, the expected happens

and they tell other friends. So now, we don't need just a house for 4 but we need a house for 8. And this is where we have a dilemma. We can do 2 things here: we can horizontally scale and get 2 houses which house 4 people each, or we can vertically scale and get a bigger house, which houses all 8 friends. Now, that we have a understanding of vertical and horizontally scaling let's talk a little bit more about what this means from a technical sense. So vertical scaling, which would be my larger house, which houses 8 people, in a technical sense would mean increasing our machine, or system capacity. This would be needed anytime where our machine needs either more disk I/O, storage, CPU, or memory to complete its task. On the other end, horizontal scaling, which we said finding 2 houses, which house 4 people each in a technical sense, would mean adding instances to share the load. This would mean adding instances or nodes to our infrastructure, adding load balancers, using Auto Scaling groups, or making our application multi-availability zone. Now let's talk through how vertical and horizontal scaling can be used in a couple of AWS services. Starting with EC2, if I would like to vertically scale, I would increase the instance size giving my instance more compute power. However, if I would like to horizontally scale I would add instances through configuring Auto Scaling. For RDS, I can vertically scale through increasing the instance size as well and horizontally scale through creating read replicas. For our exam tips for vertical and horizontal scaling just keep in mind the example we covered through understanding these two. Remember vertical scaling, or the bigger house example, means that we'd like to increase our machine, or system capacity by upgrading its disks, storage, CPU, or memory constraints to better be able to handle the load of our application. On the other end, horizontal scaling, remember our two houses, which holds the same amount of people, would be similar or parallel to adding additional instances to share the equal load. And this would be equivalent to adding additional instances or nodes within our architecture, adding load balances, using Auto Scaling groups, as well as making our application available within multi-availability zones. So if you're ready to move forward, join me in the next lecture.

Using AWS ElastiCache

Hello, Cloud Gurus. And welcome back to this lecture on using AWS ElastiCache. In this lecture, we'll be talking more about using AWS ElastiCache by answering the question what is ElastiCache? As well as getting a better understanding of caching. We'll talk about Cache Hit and Cache Miss. As well as talk about the benefits of ElastiCache. We'll talk about the two types of ElastiCache engines. As well as talking about launching ElastiCache, monitoring, ElastiCache, and, finally, we'll wrap up with some exam tips. So let's go ahead and get started. What is ElastiCache? So ElastiCache is an AWS service, which allows you to easily be able to deploy, scale, as well as manage an application's most queried data within in-memory cache in the cloud. So AWS ElastiCache is, simply, AWS' caching implementation within AWS. Let's get a better understanding of caching by using this coffee shop example. Let's say you have a customer who walks into a coffee shop, and he would like to ask some simple questions before he places his order. So he asks the person, or the barista who's working at the front desk of the coffee shop. And he may have questions like, "Does your macchiato come "with regular milk, or almond milk? " And the barista is able to answer that question. However, let's say the customer had a question like, "How many calories are in the macchiato "that I would like to order? " For this particular question because it's not a frequently asked question that the barista may know he would have to, in turn, go to his manager, and ask the question of how many calories is in the macchiato? And this is the exact same way that caching works. Now, let's understand this from a technical sense. Let's say you had an application, and for

the application's most heavily relied upon queries, they would be stored in Amazon ElastiCache. And for other information for the application they would persist within an RDS Database. Now, for those frequently asked questions that the application most likely would have these particular questions existing in ElastiCache that ElastiCache has the answer to would be called a Cache Hit because this particular information is readily available. However, in the instances where information does not exist within ElastiCache, the application must then go to the RDS Database and then retrieve the information and send it back to the application. And this would be a Cache Miss. Using ElastiCache has many benefits. First, it improves latency and throughput for several kinds of applications, which are normally read-heavy applications such as social networking, gaming, media sharing, as well as Q&A portals, as well as computer intensive workloads such as recommendation engines, or machine learning models, which consume a lot of data. ElastiCache also improves application performance by storing critical pieces of data in memory for fast data retrieval. And this type of cached data may include the results of I/O intensive database queries, or the results of computationally intensive calculations. Now, there's 2 types of ElastiCache, and you may see this on your exam, so this may be important to know. You have Memecached and with Memecached, it's good for multithreading. However, it does not allow snapshots and replications. Then you have the Redis ElastiCache engine, which does support snapshot and replication, as well as advanced data structure. Both of these are really good for data partitioning, as well as sub-millisecond latency. For launching ElastiCache there's a couple's quick steps that you can follow within the AWS service. You first, select your cluster and engine settings by choosing whether you would like to use Redis or Memecached. You'd select your node type, as well as the number of replicas. And select if you would like for your ElastiCache to be multi availability zone, as well as connect to any security groups. Next, you would also enable inbound traffic through your security group. And then, you're able to connect and add data. Some information you should know about monitoring ElastiCache is first, from a CPU utilization perspective, if ElastiCache exceeds the CPU utilization threshold you can scale out by adding more nodes. For its swap usage, if the swap usage, or virtual memory exceeds 50 megabytes you can allocate more memory for your application. And with evictions, if you're having a problem with data that you need, or expect to be within ElastiCache if non-expired items are being removed from the cache either add additional nodes, or increase the size of the nodes to accommodate the data. For concurrent connection, if your cache has too many connections you need to check your application as there may be some issues with how it is accessing your cache. So for our exam tips for using ElastiCache, if you're given a scenario where a particular database is under a lot of stress or load ElastiCache would be a good choice if your database is read-heavy and not prone to frequent changes. And secondly, Redshift is a good choice, if the reason your database is stressed is due to OLAP transactions, or online application processing transactions, et cetera. And for the final exam tips for AWS ElastiCache is keep in mind these metrics to monitor, if you're maintenancing ElastiCache keep a lookout for CPU utilization, swap usage, evictions, as well as concurrent connections as this may be questioned on your SysOps exam. So that's all for this lesson. If you're ready to move forward, join me in the next lecture.

Aurora 101

Hello, Cloud Gurus and welcome back to this lecture, Aurora 101. In this lecture, we'll be understanding the fundamentals of Aurora through answering the question, what is Aurora? We'll also discuss more about Aurora's reliability and business continuity through understanding Aurora scaling as well as its database clustering. We'll also discuss from a maintenance perspective what

actions you should take if Aurora is 100% utilized. We'll talk about Aurora Serverless and then we'll finally conclude with some exam tips. So let's go ahead and get started. What is Aurora? So Aurora is Amazon's proprietary MySQL and PostgreSQL-compatible relational database. And what Aurora provides is the speed, performance, as well as scale of popular commercial databases with the simplicity of the more common open-source database. And most importantly, Aurora provides a fault-tolerant, self-healing storage system which lowers the risk of data loss. So this is AWS's most reliable proprietary RDS database instance. Now let's talk about Aurora's scaling. So with Aurora Scaling, Aurora uses database cluster volumes. And what cluster volumes do, they virtually distribute the database storage across multiple Availability Zones. And the storage is also Auto Scaling, which for MySQL and PostgreSQL-compatible database engines, they're customized to take advantage of fast, distributed storage, which can grow up to 128 tebibytes as needed. Aurora also offers self-repairing storage or self-healing storage. Since Aurora keeps data copies in multiple Availability Zones, self blocks, its data blocks and disks are continuously scanning for errors and repairing automatically. And lastly, Aurora offers cache warming, which Aurora pre-populates its buffer pool with the pages for common queries that are stored in an in-memory page cache while provisioning other instances. Aurora offers reliability and business continuity through its Aurora database cluster. And what this is is pretty much Aurora's attempt to spread your data over 3 Availability Zones. And so here we have 3 Availability Zones. And when we configure our primary instance within the first Availability Zone, Aurora creates replicas in the other 2 Availability Zones as well as providing data copies throughout each Availability Zone. And what these data copies arrange throughout each Availability Zones are is defined as its cluster volume. So from the primary instance within that same Availability Zone, Aurora reads and writes to those data copies and each replica within its own Availability Zone reads from its data copy within its Availability Zone. It's also from the primary instance that each data copy is given new information and that's how Aurora strategically spread out its data throughout each Availability Zone. Now, within the SysOps exam, you may be questioned about how you can monitor and manage Aurora in the case that Aurora is 100% utilized. So first, if you're having an issue with writes or being able to write to Aurora, what you would do in this case is scale up or increase the instance size. If you're having an issue reading from Aurora, you would scale out or increase the number of read replicas available within your Aurora instance. Now let's discuss Aurora Serverless. Aurora Serverless is an auto-scaling version of Aurora which automatically scales capacity up and down to meet the needs of your application. Aurora Serverless also supports the existing Aurora features including multi-Availability Zone deployments, read replicas, global database, and more. Aurora Serverless is also ideal for architectures and applications which have infrequent and unpredictable workloads, yet still require high availability. Now let's discuss some exam tips that you may need to know in the SysOps exam for Aurora. Understand that Aurora comes in 2 flavors, which is Aurora and Aurora Serverless. Also understand that Aurora provides redundancy, which 2 copies of your data are contained in 3 separate Availability Zones with a total of 6 copies. Also know that Aurora's storage is self-healing. Understand that Aurora data blocks and disks are continuously scanned for errors and automatically repaired. Also keep in mind how to scale in the instance that Aurora is having issues. So if Aurora is 100% utilized for writing to the database, you'll need to scale up by increasing the instance size. And if Aurora is 100% utilized from reading, you'll need to scale out by increasing the number of read replicas. Also keep in mind the Aurora database cluster and understanding what Aurora database cluster volume is. And for more information, you can access the Aurora Overview whitepaper, which is linked in the Resources section. So if you're ready to move on, join me in the next lecture.

Understanding Aurora Auto Scaling Options

Hello Cloud Gurus and welcome to this lesson where we'll take a look at the options that are available with Aurora Auto Scaling. And we'll begin with what is Aurora Auto Scaling? We'll cover the Aurora Auto Scaling policy, target tracking scaling policy options, and my exam tips. Now Aurora Auto Scaling is a way to add or remove Aurora replicas automatically based on metrics that you define. It's available for MySQL and PostgreSQL versions, and it enables your Aurora DB cluster to handle sudden increases in read work loads by adding additional Aurora replicas automatically. And if the workload decreases, then any unnecessary replicas are going to be removed, and it's only going to remove the ones that it actually added using the auto scaling feature. To set it up, you need to create an auto scaling policy, and this is used to manage the Aurora Auto Scaling actions. Auto scaling decisions are going to be based on metrics from CloudWatch, and you define the target values. And in the next slide, we're going to explore the options for these target values. And the way that it uses the target values is that Aurora replicas are going be added or removed as necessary to try and keep the metric as close as possible to the target value that you've defined. Let me show you what I mean. So when you create the auto scaling policy, you select a target metric that you want to track, and it can be either the average CPU utilization on your Aurora replicas or the average number of connections to your Aurora replicas, whichever makes the most sense for your application workload. So think, are you constrained by CPU utilization, or is it more about the number of connections? You specify the target values either as your average CPU utilization percentage or average number of connections to your Aurora replicas. And then, Aurora Auto Scaling is going to add or remove replicas as required to try and keep as close as possible to the specified value that you've defined. If you want, you can disable any scale in actions if required. And by default, there's a 300-second cooldown period between the scaling actions, and that's adjustable in the policy. And you can also configure a minimum and maximum number of Aurora replicas that you want to maintain, up to a maximum of 15 replicas. So just imagine you've got an Aurora cluster, consisting of one primary instance and one replica. Let's say you create an auto scaling policy, which uses the average number of connections to an Aurora replica, and the target value is 5,000 average connections. If the average number of connections hits 5,000 or more, then Auto Scaling will add a new Aurora replica using the same database instance class as used by the primary instance. But then if the workload drops below an average of 5,000 connections per replica, eventually, the cluster is going scale down again, and Auto Scaling will remove only the replicas that it created itself. So for the exam, just remember that Aurora has this auto scaling feature that allows you to automatically scale Aurora replicas only. It uses target tracked scaling based on average CPU utilization on your replicas or average number of connections to your replicas. And you define what you want the desired value to be, for instance 80% average CPU or 5,000 average connections to your replicas. And Aurora adjusts the number of replicas based on the workload and desired values. So that is all for this lesson. Any questions, let me know. Otherwise, I will see you in the next lesson. Thank you.

RDS and Multi-AZ Failover

Hello. And welcome back to this lecture on RDS multi-availability zone failover. In this lecture, we'll understand a little bit more about RDS and its multi-availability zone feature by understanding the basics of RDS multi-AZ. We'll also understand how its failover works, what exactly is the feature, as well as talk through its advantages. We'll wrap up this lecture with some exam tips and we'll be on to the next lecture. So let's go ahead and get started. RDS's multi-availability zone

feature provides a couple of values to us for our database persistence. First, it keeps a copy of our production database in a separate availability zone in the case of a failure or a disaster. It also manages the failure from one availability zone to another automatically through AWS. With RDS's database technology, specifically MySQL, Oracle, and PostgreSQL. These technologies use synchronous physical replication to keep standby instance data current with the primary instance. SQL server, on the other hand, uses a logical replication and native mirroring technology to ensure the standby is up to date. Keep in mind that both these approaches safeguard your data from database instance failure, as well as the loss of an outage in an availability zone. Now let's discuss how failover works. Let's say one of my instances which are currently within the US-West-2 availability zone, is currently experiencing a local weather event. Let's say like a tornado or a hurricane, and the power goes out as well as the generator. So what happens in this scenario is RDS will detect that and basically will take the endpoint and update the DNS, which will point to an availability zone which is up and running without failure. With this concept in mind, this helps us understand that multi-availability zone within RDS is for disaster recovery only. It is not primarily used for improving performance. If you'd like to improve your performance, you'll need read replicas. For the advantages of RDS multi-availability zone failover, first would be high availability. So high availability would protect against these particular instances where we have natural disasters or weather events, and making sure that your data persistence is highly available to protect from outages. Another advantage would be backups and restores, and this would ensure that your backups and restores are taken from your secondary instances, which would avoid input/output suspension from your primary database instance, which would also prevent downtime or outages. So an exam tip for here is to understand that you can force a failover from one availability zone to another by rebooting your instance. This can be done through the AWS management console or through using the reboot DB instance API call through the AWS CLI. Now let's talk through some exam tips for the RDS and multi-availability zone failover. You'll need to remember that read replicas are what's used to scale and you can force a failover from one availability zone through simply rebooting an instance or using the AWS CLI RebootDBInstance API call. Another thing that you'll need to know is that RDS multi-availability failover is not a scaling solution. And Amazon also handles failover for you by updating the private DNS for the database endpoint. Backups and restores are taken from the secondary multi-availability zone instance. Thanks for joining me on this lecture and I'll see you in the next.

RDS and Read Replicas

Hey Cloud Gurus and welcome back to this lecture on RDS and Read Replicas. In this lecture on RDS and Read Replicas, we're going to talk about what are Read Replicas as well as when is it best to use Read Replicas? We'll talk through some supported versions of RDS that you can use for Read Replicas as well as how to create and connect to Read Replicas. And as always, we'll wrap up with some exam tips. So let's go ahead and jump in. What are Read Replicas? So, Read Replicas make it easy to take advantage of supported engines' built in replication functionality. And that functionality allows us to elastically scale out beyond the capacity constraints of a single database instance for read-heavy database workloads. So Read Replicas are simply read-only copies of our database. So this allows us to take a lot of the read functionality off of our primary instance which frees it to be able to do its job as storing and persisting the data that we need it to. And in preparation for your SysOps exam, I highly recommend that you check out the Amazon RDS Read Replica white paper and the link is below at the bottom of your screen. So be sure to check that out. So when should we

use Read Replicas? It's a good idea to use Read Replicas for scaling for additional capacity that allows us to scale beyond the compute or I/O capacity of a single instance for read-heavy database workloads. And what this does is allow access read traffic to be directed to one or more Read Replicas. Another great thing that it's good for is for scheduled maintenance or backups. This allows our applications to serve read traffic while the source database instance is unavailable for some maintenance or backup. And if your source database instance cannot take I/O requests due to suspension for backups or schedule maintenance, you can direct traffic directly to your read replicas. Read Replicas are also great for business reporting or data warehousing. It's great for running business reporting queries against read replicas rather than your primary database instance which can cause I/O or capacity issues within your database. Now let's talk about some supported versions. So with MySQL, PostgreSQL, MariaDB, as well as SQL Server, Amazon uses native synchronization replication to update the read replica. For Amazon's proprietary RDS database, Aurora DB, Aurora uses a virtualized storage layer backed by SSD, which is specifically built for database workloads. Amazon Aurora replicas also share the same underlying storage as the source instance which allows us to lower costs and avoid needing to copy the data to the replica nodes. And we'll talk a little bit more about how Aurora uses clustering and database volumes to share and create Read Replicas in another lesson. There's a couple things you should know for Creating and Connecting to Read Replicas. We'll start with creating. When creating a new read replica, AWS takes a snapshot of your database. And there's a couple ways that AWS does this if multi-availability zone is enabled, the snapshot will be of your secondary database and you won't experience any performance hits on your primary database. However, if multi-availability zone is not enabled this snapshot will be of your primary database and you may see a brief suspension of around 1 minute. So for connecting, when a new read replica is created you will be able to connect to it using a new endpoint DNS address. So whenever you're trying to connect to a new read replica that has spun up, you'll be able to find it within RDS in the AWS console, or CLI. For our exam tips for RDS and Read Replicas there's several things you may need to remember. First, you can have up to 5 Read Replicas for MySQL, PostgreSQL, SQL Server, and Maria DB. Also, you can have Read Replicas in different regions for all engines and you'll need to understand that replication is asynchronous only, not synchronous. And Read Replicas can be built off multi availability zone databases. Read Replicas themselves can now be multi-availability zone. You can have read replicas of read replicas but beware of latencies. So keep that in mind if you're making a read replica from an existing read replica there may be some latency there within your reporting. Also database snapshots and automated backups cannot be taken of read replicas. And the key metric to look for is replica lag. And the final exam tip would be to know the difference between Read Replicas and multi-availability zone where keep in mind that Read Replicas are our read only database, which helps us in giving our architecture's additional read capacity. They assist for scheduled maintenance, as well as those instances where our application may conduct intensive querying and multi-availability zone on the other hand, is our proactive approach to disaster recovery. So if you're ready to move on, join me in the next lecture.

Demo: Creating and Encrypting RDS Snapshots

Hello Cloud Gurus and welcome to this demo on creating and encrypting RDS snapshots. Now one of the things you need to remember for the exam is that there is not a one-click button or solution that will allow you to add encryption to your snapshots once they've been created. And you will need to follow a sequence of steps in order to do so, and we're going to talk through each of these

steps in this demo. And we'll get started by taking a snapshot of an existing RDS instance. And we can then copy that snapshot, either to the same region or to a different region. We'll then encrypt the snapshot during the copy process. And then finally, we'll finish off by restoring using the snapshot. So if you're ready to get started, please join me in the AWS console. So from the console, search for RDS, Create database, select Standard create. Database engine type is MySQL. Scroll down. The community edition is fine and just accept the version that it gives you. Under Templates, select Free tier. Scroll down. And under DB instance identifier, it's going to be acloudguru test. And I'm going to use the same name for my master username and for my master password as well. Just confirm the master password. Scrolling down to Instance configuration, make sure that Burstable classes is selected and select db.t3.micro. For storage, we'll stick with all the defaults and then the same for connectivity as well. Scrolling down until you get to the database authentication, and it's already selected Password authentication. Then scroll down to Additional configuration. Open the drop-down. For the initial database name, we'll just make that acloudgurutest. Then scroll down until you get to Encryption. And notice that when you use the AWS console to create an RDS database, even a free tier one, it's going to automatically enable encryption by default, and it's going to use the default encryption key that is owned by the IDS service. Now this is a configurable option, and I want to show you what happens when we have an RDS instance that does not have encryption enabled. How are snapshots going to behave? So let's remove encryption and then scroll down to the bottom and Create database. Now we need to wait a few minutes for it to finish creating. And once it's showing a status of Available, then we are good to continue, and you can refresh just by using this button. So now my database has successfully created, and it's showing a status of Available. The next thing I need to do is click on my database, select Actions, and select Take snapshot. And all I need to do is give my snapshot a name. I'll call it unencrypted-acg-snapshot and select Take snapshot. And if you scroll to the right, you'll see the status is Creating. You can watch the progress here. And if you scroll further, you should get to the Encrypted section, and Encrypted will be set to No. And this is because our original RDS instance was unencrypted. Now it is going to take a few minutes to complete this snapshot. We can watch the status and progress here. And in a few minutes, we will be good to continue. After your snapshot has successfully created, it should show a status of Available. I'm then going to scroll back to the left and select my snapshot using this box on the left-hand side. And now, I can select Actions, Copy snapshot, and this is how we can actually create an encrypted copy of our snapshot. And there's several configuration options here. First of all, we can set the destination region. So we can actually copy our snapshot to another region if we want to, but let's just keep it to US East and Northern Virginia. For the new database snapshot identifierm let's call it encrypted-acg-snapshot. Scroll down. Under Encryption, this is where you can enable encryption for your copy, and I'm going to stick with the default key, which is the aws/rds default KMS key. And if you select Info up here and check the message that appears on the right, it's just telling us that we can choose to encrypt the copy of the source database snapshot. However, if the source snapshot had already been encrypted, we would not have the option to remove encryption. So you cannot remove encryption from an encrypted database snapshot. So now, we can go ahead and scroll down and select Copy snapshot. We'll give this a few moments to finish creating. But once again, if you scroll to the right, you will see the that the new snapshot that we're creating has encrypted set to Yes. So we can see that our copy is going to be encrypted. So now let's just wait a few moments for it to complete. After a few minutes, it should have completed, and now we're ready to restore using this encrypted snapshot. So make sure you select the right one. For me, it's this one here, the one at the top of the list. So scroll across to the left-hand side and select your encrypted snapshot. Then select Actions,

Restore snapshot, and we need to set a couple of configurations before we can restore from our snapshot. We'll keep the database engine the same, MySQL Community. Under Availability, I'm just going to create a single database instance because that's just going to be faster for this demo. Under Settings, here's our snapshot ID, and we need to provide a new database instance identifier. And I'm going to call-it new encrypted-db. Then scroll down. We'll keep all the connectivity information the same. Instance configuration is going be t3.micro, so select Burstable classes to include the t classes, and make sure that db.t3.micro is selected. The storage settings can all stay the same. Database authentication is going be Password authentication. And then, under Encryption settings, encryption is going to be enabled using the default RDS KMS key, and this has all been grayed out because we'll be using the same settings that are in the snapshot that we're restoring from. So we'll be using the same RDS master key for KMS. So now, we can go ahead and select Restore database instance. And now we can see that our new encrypted ACG snapshot database is currently coming up, and we don't even need to wait until it's available. We can actually select our encrypted database and select Configuration. And then if you find the Storage section, you will see that encryption is enabled. And we're using the AWS KMS key that is owned by the RDS service. So we are successfully restoring an encrypted snapshot that was based on a copy of an unencrypted snapshot and an unencrypted database. So finally, let's finish off with some exam tips. And there's a few additional things that are worth remembering when it comes to snapshots. Firstly, you can use the AWS Management Console, the CLI, or the Amazon RDS API to share encrypted snapshots with other AWS accounts, for instance with other AWS accounts that you own like development, test, or production accounts. When sharing encrypted snapshots across accounts, you will also need to share the KMS key that was used to encrypt the snapshot in the first place. So that also means that if you are going to share an encrypted snapshot, then it will need to have been encrypted by a customer-managed CMK and not the default one because you won't be able to share the default key with any other account. And then finally, you can manually create RDS snapshots within the RDS service just like we did in this demo. Or you can create them in an automated way using AWS Backups. So that is all for this lesson. Any questions, please let me know. Otherwise, please join us for the next lesson. Thank you.

Differentiating Single Availability Zones vs. Multi-AZ Deployments

Hello, Cloud Gurus, welcome back to this lecture on Differentiating Single Availability Zones vs. Multi-Availability Zone Deployments. In this lecture, we'll talk about some important things you should know when differencing whether you should make an application single or multi-availability zone. We'll understand this through diving into the factors you should consider and questions you should ask yourself as well as going through some exam tips. Let's go ahead and jump in. As we know, single availability zone deployment would be having all of your resources within one availability zone while a multiple availability zone deployment would house all your resources in separate availability zones to support a failure. These are some questions and factors you may like to consider as you decide whether your application should support single or multiple availability zone deployment. First, should your application have fault tolerance? You should probably ask yourself, "Should components "of my application exist in other availability zones "in the case of a failure? " This will be important for those businesses and use cases which would suffer tremendously if the application was not available in the case of a failure. Another instance to consider is the cost. Could you save money by limiting the application to a single availability zone? You should also consider the level of availability that you would like. Should your application be

spread across multiple AZs to meet the traffic demand? Now let's talk through a couple factors to consider as it relates to EC2 and Auto Scaling for single and multi-availability zone. So for EC2 Auto Scaling, keep in mind a single availability zone would be great for non-critical workloads and architectures. This would be something like a development or a testing architecture of your application. And this would also introduce low fault tolerance, low availability, as well as low cost. For EC2 Auto Scaling in multiple availability zone, this would give you the ability to automatically route traffic to different availability zones through Elastic Load Balancers. For RDS, this is also great for non-critical workloads and architectures, however, it also creates a single point of failure. For RDS and multi-availability zones, this would introduce limited downtime in the case of a failure, as well as no interruptions during backups, patches, or upgrades. So for your exam tips, as it relates to single versus multi-availability zone deployment, keep in mind these 3 factors: fault tolerance, if your application components should exist in other availability zones in case of a failure; think about your cost, could you save money through limiting the application to a single availability zone, as well as the availability, understanding should your application be spread across multiple availability zones to meet your traffic demand? And also remember these particular trade-offs as it relates to having single or multiple availability zones within EC2 Auto Scaling as well as RDS. So that's all for this lecture. If you're ready to move on, join me in the next lecture.

Implementing Fault Tolerant Workloads Using Amazon Elastic File System (EFS)

Welcome back, Cloud Gurus. And in this lecture, we'll be discussing implementing fault tolerant workloads using Amazon Elastic File System or EFS. In this lecture, we'll talk about how to improve fault tolerance within our workloads using EFS through getting a better understanding of fault tolerance. We'll also briefly refresh on EFS as it's previously been covered within this course. We'll talk through bursting versus provision throughput as well as discuss how EFS handles encryption. And then we'll wrap up with some exam tips. So let's go ahead and get started. Let's get started by understanding fault tolerance. Fault tolerance is the ability for a system to remain operational even if some of the system components used to build the system fail. And to assist with building fault tolerance within your architecture or workloads, AWS offers several services which scale in the event of any failure. And these services range over compute services, storage, networking, as well as database. And for more information on how AWS handles fault tolerance be sure to check out the white paper at the bottom of the slide here as well as that's made available in your resources section. Amazon Elastic File System is a simple, fully managed Elastic File System used with several AWS compute services. Now by Elastic File System, we mean that this particular file system has elasticity or the ability to expand and retract its size and throughput availability based on the demand of our application. Now, the compute services that Elastic File System supports includes Amazon EC2, Amazon Container Service or ECS, Amazon EKS, or Elastic Kubernetes Service, as well as Amazon Fargate. As it relates to implementing fault tolerance within our architectures within AWS, just using Elastic File Service alone does not make your architecture fault tolerant. However, the ability for your architecture to be in multiple availability zones. So for this example where we have Elastic File System, the important thing to note here is that in order for this to be a fault tolerant architecture, we would need to put mount targets within each availability zone so that our compute would have access to EFS through each availability zone. And that's how fault tolerance is introduced within this workload. There's several other things to note about EFS and its ability to implement fault tolerance within our workloads. First, it offers high availability

and durability. Data written to EFS is written to 3 availability zones and accessible through all availability zones within a given region. And this offers 99.99 % availability. EFS is also elastic and scalable, which EFS only charges for the capacity used and also scales to meet storage and throughput capacity, which we'll talk about a little bit more in detail in just a moment. EFS is also great for container and serverless file storage and support. EFS integrates with multiple container services, such as Lambda and EC2. EFS also integrates with AWS Backup for automatic backups. And lastly, EFS offers storage classes and lifecycle management. EFS offers 4 storage classes which includes EFS Standard, Infrequently Accessed, One Zone, and One Zone Infrequently Accessed. The lifecycle management moves files based on usage patterns through lifecycle policies. Amazon Elastic File System handles capacity through 2 different throughput options. First, which is bursting throughput where Amazon EFS scales with the file system size as well as uses the burst credit system to determine burst as well as its cost. EFS also offers provision throughput, which is fixed at a specified amount and can increase any time, but only decrease after 24 hours of the most recent decrease. Keep these 2 throughput settings in mind as they may be questioned on your SysOps exam. Now let's discuss how EFS handles encryption. For encryption at rest, EFS allows data at rest to be encrypted through the AWS Management Console using the integration of KMS or Key Management Service. For encryption in transit, EFS allows transitional data to be encrypted by enabling TLS or Transport Layer Security through the EFS Mount Helper. For our exam tips for implementing fault tolerance using EFS, you'll need to remember the 2 types of throughput, starting with bursting throughput. Remember that this is the throughput that allows the file system's throughput to scale as the system grows, as well as scale with the file system size. This also uses the burst credit system to determine burst as well as determine cost for EFS. On the other hand, our provision throughput allows you to instantly provision throughput of the system, independent of the file system size. And this is fixed at a specific amount. And remember, you can increase this at any time but only decrease throughput 24 hours after the most recent decrease. From a security perspective, keep in mind that encryption at rest allows the data to be encrypted using the integration with KMS or Key Management Service. However, encryption in transit uses the transport layer security or TLS through the EFS Mount Helper. That's all for implementing fault tolerance using EFS. So if you're ready to move on, join me in the next lecture.

Building Fault Tolerance Using Elastic IPs

Hello, Cloud Gurus, and welcome back to this lecture on building fault tolerance using Elastic IP addresses. In this lecture, we'll get a better understanding of how Elastic IP addresses can be used within your architecture to build fault tolerance. We'll do that through getting a better understanding of what Elastic IPs are, how they can be used with EC2 instances, and then we'll wrap up with some exam tips. So let's go ahead and get started. Let's better understand what Elastic IPs are through understanding the issue we would have if we did not have them. Let's take our cell phone for an example, which is our personal machine which runs all the apps that we like from finance to social to news and any other applications you can think of. And with our cell phone, we're given a number that others can reach us by calling. Now, let's say every time we received a notification that another version was available for download as the applications update and install, we'd be given another number. And then another version be made available and we'd be given another number. Thankfully, this isn't the case because we'd have a very unique challenge of redistributing our cell phone and contact information to all of our contacts within our phone. And this is the same issue that Elastic IP solved for from a technical sense for when we update our resources within AWS and those

resources are reassigned an IP address, we then have to go back to all of our applications which reach out to these resources and tell them where the resources can be found. Now let's better understand what an Elastic IP address would help solve with an EC2 instance. So let's say I have an EC2 instance, which is given a public IPv4 address. And for some reason that particular instance reboots or terminates and another one comes back up. Well then the IP address changes, which makes this particular IP address dynamic or changeable. However, what the Elastic IP address would do is give one particular IP address, which does not change or is static, which will point always to this particular EC2 instance or service, no matter how much it changes. And that's generally the purpose of the Elastic IP address. So for your SysOps exam, it's important to note that Elastic IP addresses are a great way to introduce fault tolerance within your architecture so that in the case of a disaster or some of your services going down, Elastic IP addresses will help make sure that your services can automatically be routed to new resources that are made available. Now let's cover some exam tips to know. First AWS accounts are limited to 5 Elastic IP addresses per region. Also Elastic IPs are public static IPv4 addresses. And the primary purpose of the Elastic IPs are the ability to remap network traffic in the case of a failure. So if you're ready to move on, join me in the next lecture.

Readying for Disaster Recovery

Welcome back Cloud Gurus to this lecture on readying for disaster recovery. In this lecture, we'll be covering the information you need to know for preparing for disaster recovery. We'll do that through understanding 2 key concepts which help you frame your disaster recovery approach, which are RTO and RPO. We'll also discuss the service AWS Backup and its role within disaster recovery. We'll discuss backups and snapshots through getting a better understanding of which AWS service conducts them. We'll also discuss some disaster recovery strategies, which are outlined within the AWS whitepaper. And finally, we'll conclude with some exam tips. So let's go ahead and get started. Let's get started by understanding RTO and RPO as these 2 concepts are important to how you build for disaster recovery within your infrastructure and architecture. So let's get started by saying within a given timeline, although we really don't want it to happen, we unfortunately have to plan for any type of disaster recovery that can happen. And by disaster recovery, we mean some type of natural outage that happens within our system. Well, the one particular part that we need to understand is our recovery point objective. And our recovery point objective tells us how many backups or how much data do we need to capture for a given point. And even if you've never heard of this concept, you may understand it as how frequent do your backups need to be taken? Maybe it's on an hourly basis, or every couple of hours, every day, every 2 days, etc. And the question that we have to ask ourselves with recovery point objective is how much data can my application afford to lose? And this goes on an application-by-application basis. So a file system with some PowerPoints or some data, that's not really important, we may not want to spend as much money to backup that data hourly. However, our customer information, we may want to backup that data more frequently. So that's RPO, our recovery point objective. Now on the other end, is our recovery time objective, or our RTO. Our RTO guides the infrastructure or architecture that we decided to use by understanding how long it would take for us to get back on track or get our application back on track after the disaster recovery has occurred. So the question for RTO that we're generally asking is, how long can I afford for my application to be down? So this helps us understand should we choose architectures, or compute resources that are more easily available to be brought back up after a given disaster has occurred. And that's how we understand RTO. So those 2 concepts are very important for us to

know not only for the SysOps exam, but for general disaster recovery purposes, as they frame how we should respond and prepare for disaster recovery. Now, let's discuss AWS Backup, which is a centralized service which helps you automate backups and manage data protection at scale through the console APIs or CLI. And what you're able to do within AWS Backup is first create a backup plan based on the RTO and RPO, which we just discussed. You're also able to assign AWS resources to that plan, as well as monitor, modify, and restore your backups. As it relates to configuring your backups, you need to understand which services offer service-level backups and snapshots. Let's start with service-level backups, which would include Amazon FSx, which is a Amazon file system. Very similar to Amazon EFS, which is Elastic File System, which we've covered recently within this course, as well as DynamoDB and EC2. For our service-level snapshots, we have Aurora, Amazon RDS, Amazon EBS, which is Amazon Elastic Block Storage, as well as Amazon Storage Gateway. Now let's discuss some disaster recovery strategies. Now as we've mentioned before, disaster recovery should be assessed on an application-per-application basis in order to save costs. As the difference in disaster recovery approaches, costs can range from low to high. So on the lower scale, we have the traditional backup and restore approach. And in this approach, our RTO and RPO can range to multiple hours. For a pilot light approach, our RTO and RPO can be tens of minutes. For a warm standby approach, the RTO and RPO can be minutes. And for the multi-site active, our RPO and RTO can be real time. Traditional backup and restore approach, this is for lower priority use cases, which may be something like a development site or a lower priority application. This would allow us to restore data as well as deploy resources after the event has occurred. For the warm pilot light, this would be a less stringent RTO and RPO, where our core services can start and scale resources after the event. For a warm standby approach, this will be a more stringent RTO and RPO for business critical services. And this will scale resources after the event. For the multi-site active or active event, this would be the most costly approach. However, this would be the approach which would allow for us to have 0 downtime, near 0 loss, and this would be for mission critical services. And for more information on these, be sure to reference the disaster recovery workloads on AWS whitepaper, which has been linked in the resources section. Now let's discuss the exam tips for this section. You'll first need to understand the recovery point objective, RPO, which makes us ask the question, how much data can I afford to lose within my application? And this helps us to answer, how often should we have backups? You'll also need to understand the recovery time objective, which causes us to answer the question, how long can I afford for my application to be down? And this will help us understand the scaling approach and the resources we would like to include within our application to get our application backup and running. Also understand the difference between backups versus snapshots and which services leverage backups and snapshots per use case. That's all for this lecture. If you're ready to move on, join me in the next.

AWS Service Maintenance Windows

Hello, Cloud Gurus, and welcome back to this brief lecture on AWS service Maintenance Windows. In this lecture, we'll have a brief overview of what are AWS service Maintenance Windows, we'll understand them, as well as talk through which services within AWS have service Maintenance Windows, and we'll conclude with some exam tips. Let's understand service Maintenance Windows. And service Maintenance Windows is basically scheduled downtime which AWS uses to apply configuration changes or updates to application components. Just some things that's done within service Maintenance Windows are the application of patches, installing or update of applications, as well as operating system updates. The important thing that we'll need to note is understanding

which services have Maintenance Windows within AWS, which is Amazon RDS, Amazon ElastiCache, Amazon Redshift, Amazon Neptune, as well as Amazon DocumentDB. So for your exam tips, you'll need to know which services within AWS have service Maintenance Windows, just in case you're asked how to allot for service Maintenance Windows within the SysOps exam. So that's all for this lecture. I'll see you in the next.

Configuring S3 Cross-Region Replication

Hello, Cloud Gurus, and welcome back to this lecture on configuring S3 cross-region replication. In this lecture, we'll be discussing how configuring S3 cross-region replication contributes to high availability and business continuity within our architecture. So we'll understand cross-region replication as well as its benefits, walk through some configuration steps and conclude with some exam tips. Let's go ahead and jump in. Cross-region replication as we see here with our map is pretty much allowing our data to be placed in different areas or regions to create a higher level of availability within our application. And there's several benefits to that, which we'll cover here for the best uses of cross-region replication. The first one may be to meet certain compliance objectives, which is perfect for applications that have compliance standards requiring that data be geographically spaced out. Another benefit of cross-region replication is to increase efficiency, which for architectures with cross-regional compute, replicating data would assist with application efficiency. And finally, cross-region replication improves latency. Enabling cross-region replication improves application latency by moving data closer to the consumer. Now let's talk through some configuration steps, which is helpful to understand how cross-region replication in S3 works. The first thing you'll need to do is create S3 buckets in separate regions. So we create our S3 buckets. For example, I'm in us-east-1 and I'll create another bucket in us-east-2. The second thing we need to do is make sure we turn on versioning in both regions, which versioning within both regions is required for cross-region replication within S3 to work. We'll lastly needs to create a replication rule by selecting the source and destination buckets. Also select the role that you would like to use to create the replication. Now let's talk through our exam tips for configuring S3 cross-region replication. Cross-region replication in S3 is a great way to introduce high availability within our architecture. So we'll need to know that on our exam if we're asked as this is a possible viable solution. We'll also need to remember our reasonings for this as this may be asked on the exam. So remember, to meet our compliance objectives. We'll also need to understand that it's a great way to increase efficiency as well as improve latency within our application by moving the data closest to the available consumer. Thanks for joining this lesson and I'll see you in the next.

Implementing Loosely Coupled Architectures with SQS

Hello, Cloud Gurus. And welcome back to this lecture on implementing loosely coupled architectures with SQS. In this lesson, we'll help you understand how you can implement loosely coupled architectures within your environments using SQS. We'll do this by getting a better understanding of what's SQS. We'll talk about the benefits of why you should decouple your architectures. Then get a better understanding of how SQS is used within a loosely coupled architecture by looking at a architectural diagram. And as usual, we'll wrap up with some exam tips. Let's go ahead and get started. Let's get started by understanding SQS. SQS or Amazon Simple Queue Service is AWS's fully managed message queuing service. This service is great for sending messages to different components of our infrastructure in a serverless and scalable way. In order to better understand how SQS works, we'll need to get a better understanding of the three critical

components that interacts with the service. We have the producers and producers are the infrastructure components or resources which messages are sent from. These could be applications, microservices, or other AWS services. The producers sends the message to the SQS queue where messages are then stored and wait for the next portion which is the consumers to pull. Consumers then process the message that it receives from the SQS queue. Consumers can be Lambda functions, EC2 instances, and other AWS services. So as we see from the left to the right of our screen we see the producers send the message to the SQS queue, where the SQS queue stores and holds the messages as it waits for consumers to receive messages to be processed into its application. Now, let's talk about how introducing this form of serverless messaging into your architecture can assist you with decoupling and the benefits decoupling provides. Why decouple? So simply put the purpose of decoupling is to make sure that components within your application can act and perform their assigned tasks independently of one another. And through the use of SQS, this allows us to decouple within our architectures so systems can process messages, functionally and serverlessly. Here's several other reasons why it's important to introduce decoupling within your architecture. It allows you to simplify updates to your components. It also allows you to reduce your dependencies, as well as increasing autonomous components within your architecture. It allows you to create non-blocking operations, as well as provides a logical decomposition of systems, which makes it very easy to introduce fault tolerance and scalability within your architecture. Now let's discuss how SQS looks within a loosely coupled architecture. So let's say we start with a client who would like to submit a request through our website. The request is then routed through Amazon's Route 53, which is the DNS service for AWS. The request is then forwarded from Route 53 to Amazon CloudFront as it retrieves the static website information from Amazon S3. The information needed to process client request is then forwarded to the elastic load balancer and the elastic load balancer's job is to find available resources within a given availability zone to complete this particular transaction. Within this particular architecture, auto-scaling is then enabled, which allows EC2 instances to be able to handle this request. The request is then executed and the transaction is committed to the RDS instance main database. After the main database has successfully recorded the transaction, the transaction will later be replicated to a different database in a different availability zone or the standby database. Now, let's see how this looks with SQS involved. Now that SQS has been introduced, let's understand how we loosely couple our architecture. In the previous diagram, we saw that our auto-scaling group was directly responsible for applying changes to our Amazon RDS instance. However, in the updated architecture all order messages that are coming from our elastic load balancer are directly applied to the customer order queue and error messages that are contained within the customer order queue are also forwarded to the dead letter queue. It's then after these messages are successfully stored within the order queue later, Amazon EC2 instances within an auto-scaling group can then pull the queue and process the messages as they'd like. Of course, first to the main Amazon RDS instance and later replicated to the Amazon RDS standby instance. And so this is how you would implement loosely coupled architecture with Amazon SQS. For our exam tips for implementing loosely coupled architectures with SQS keep in mind you'll need to know what exactly is SQS and what it does. And SQS is a fully managed message queuing service which creates queues that producers use to store messages and wait for consumers to pull them and receive them and process them. And lastly, remember the benefits of decoupling your architectures or using a service like SQS to do so just simplifies the application updating, as well as increase autonomous components and lastly, create non-blocking operations. That's all for this lecture. If you're ready to move on, you can join me in the next lecture.

Demo: Automating EBS Snapshots Using Data Lifecycle Manager

Hello Cloud Gurus and welcome to this demo on automating EBS snapshots using Data Lifecycle Manager. And we'll begin by launching an EC2 instance with an attached EBS volume. Next, we'll create a lifecycle policy within Data Lifecycle Manager that is going to be responsible for automating our snapshot process. And then after we've completed that, we'll enable cross-region replication for our EBS snapshot. And do be aware that if you would like to create a cross-region snapshot, then you will need to do this in your own AWS account because our cloud playground will not give you permission to create any resources in a region that is not us-east-1. So if you like to do the steps exactly as I do, then please use your own personal account. However, if you only have access to the cloud playground, you can still do the majority of steps, and you can still use the Data Lifecycle Manager to create automated snapshots, but your snapshot will have to be created in the same region and not cross-region. So if you are ready to move forward, please join me in the AWS console. So from the console, and I'm in my personal account, let's search for EC2 and launch an instance. Select Launch instance. Call it DLM Test. I'll use Amazon Linux. Instance type is going to be t3.micro. We can proceed without a key pair. Scroll down to storage, and we're going to add a new EBS volume. So click Add new volume. The volume type is going be an EBS volume, and it's going to be gp3. And I'll go ahead and Launch instance. Select your instance ID, and we'll just wait a few moments for the instance to finish initializing. So now our instance is ready, and we are ready to create our lifecycle policy. Now it's important to note that when automating an EBS snapshot using Data Lifecycle Manager, we need to use a tag to recognize our EBS volume. So first of all, I'm going to select Volumes, and I need to add a tag to my volume. So first of all, you'll need to locate your volume, and it's going to be this one, the gp3 type volume that was created this morning. And this gp2 one, this is my root volume, and this is my data volume. And then this one up here, this was created a few months ago, and it just looks like there's an old volume just hanging around in my account. But this is the one that we're looking for, the gp3 type volume. So I'm going to give my volume a name. I'm going to call it EBS-Automation-Snapshot-Enabled and hit Save. So then, when we select Lifecycle Manager and create a new lifecycle policy, we've got the option to create an EBS snapshot policy and EBS-backed AMI policy and a cross-account copy event policy. But we're going to be using an EBS snapshot policy, so hit Next. For the target source type, it's going be an EBS volume, and it's already identified my EBS volume. But if it hasn't done this, then you can just select a key of Name, and then the value is going to be EBS-Atomation-Snapshot-Enabled. So once you've selected that, make sure it is added, and it appears down here. Moving down to Description, you have to give a description. I'm going to call it EC2-EBS-Snapshot-Daily. For the IAM role, we'll just use the default role that's been created for Data Lifecycle Manager. We can bypass this Tag section. The policy status should be set to Enabled, so click Next. And here's the scheduling details. So for this schedule, let's give our schedule a name. I'll call it EBS-Daily-Snapshot. We need to select a frequency, and I'm going to select Daily. But you could use weekly, monthly, yearly, or a custom cron expression. I'd like it to run every 24 hours, and I'll also choose a time for it to start. But keep in mind, this is in UTC time, and that happens to be the same time as the time zone that I'm in right now. So I'm going to set a time for the next 10 minutes, so I don't need to wait too long for my first snapshot to appear. For the retention type, you can either retain a number of snapshots, or you can retain a certain age of snapshots. And for this example, let's say I want to keep five snapshots on hand. Moving down to Advanced settings, there's a couple of settings that we need to discuss. The first one is tagging, and we can just check this checkbox here to copy the tags from our EBS snapshots. Down here, we also have the option for a fast

snapshot restore. So if you'd like to have any snapshots templatized and available for quick retrieval, then this is a great option. We've got the cross-region copy, which I want to do. So enable cross-region copy for this schedule. And remember, you can only do this if you're working in your own AWS account. And if you are working in the cloud playground, it will let you configure it, but it won't actually work. So I'm going to go ahead and select a target region. And I'm going to select us-east-2, and I want this to expire every 3 days. Scrolling down, we can see that encryption is enabled for the snapshot copies. It's using the default aws/ebs KMS key. And we're going to select Copy tags from source. So now, let's go ahead and review our policy. Then scroll down to the bottom and Create policy. So now, our Data Lifecycle Manager policy has been created, and it's in a state of Enabled. But do keep in mind that it may take around about an hour for the first snapshot process to run successfully. So I'm going to pause the video now and give this time to just run through, and then we'll come back and see if it's created our snapshot. So now, it's been about 15 minutes, and I'm ready to go in and check if my snapshots have been successfully created. So I'm going to select my policy ID up here. Scroll down. And under Schedules, you can select Show snapshots. So here is our first snapshot, but it's only showing one. Well, that is because these are regional, and this is my cross-region copy located in Ohio, which is the us-east-2 region. And if I go ahead and select us-east-1 up here, there is my other snapshot. So now, as we're working in our own AWS account, we need to go ahead and clean up all the resources that we created. So I'm going to select my snapshot. Select Actions and Delete snapshot. I'll do the same in us-east-2. Then, I'll head back to the Lifecycle Manager using the left-hand menu. Make sure I'm in us-east-1 because that's where my policy exists. And it's important to delete the Data Lifecycle Manager policy. So select your policy and Delete lifecycle policy. And then the last thing I'm going to do is delete my EC2 instance. So select your instance. Come to Instance state and Terminate instance. So now that we've cleaned up all of our resources, let's wrap up with our key takeaways from this demo. So just remember that Data Lifecycle Manager is a feature within the EC2 service that allows you to automate the creation, retention, and deletion of EBS snapshots and EBS-backed AMIs using lifecycle policies. You can also encrypt the snapshots using KMS, and you can enable and configure them to have cross-account sharing as well. And as you saw in the lesson, it's really easy to create cross-region snapshots as well. So that is all for this lesson. Any questions, please let me know. Otherwise, if you're ready to move on, please join us for the next lesson. Thank you.

Using DynamoDB Streams for Backing Up Your Table to Another Region

Hello Cloud Gurus and welcome to this demo on using DynamoDB Streams for backing up your table to another region. And let's quickly refresh on what is DynamoDB. Now according to AWS, DynamoDB is a fast and flexible NoSQL database service, which is designed for applications which need consistent, single-digit millisecond latency at any scale. It's a fully managed database that supports both document and key value data models. And it's this flexible data model, reliable performance, and scalability that make it a great fit for mobile, web, gaming, ad-tech, Internet of Things, and many, many more other types of application. Now, let's take a look at DynamoDB Streams. Now, a DynamoDB Stream is a time-ordered sequence of item-level changes occurring directly in our DynamoDB table. And when I say item-level changes, what I really mean is that we've got a stream of inserts, updates, and deletes that have been happening in our source table. And the DynamoDB Stream maintains this sequence of updates in what is called a shard, which is simply made up of a stream of these item-level changes. These changes are stored for 24 hours, and data entered into the DynamoDB Stream can also be combined with a Lambda function to trigger an

automated response to events that are happening in the stream or in our table. Now this lesson is a demo, so let's take a look at our objectives for this demo. And we'll begin by creating a DynamoDB table, and we're going to create it in us-east-1. Then, we'll enable DynamoDB Streams on our table, and we'll add a replica table, and we'll select a desired region to create the replica table in. And do be aware that If you want to create a table in a region that is not us-east-1, and that's what we're going to do in this demo, then you will need to use your own AWS account because we're going to be creating a replica table in us-east-2. Then after we've done that, we'll input some data into our table and confirm that the table is replicating as we'd expect. And, of course, if you want to create a cross-region setup like this, then you will need to use your own AWS account because our cloud playground is only going to allow you to create resources in us-east-1. So please use your own account for this demo. So from the console, search for DynamoDB. Create table. We'll call it dynamodb-streams-test-table. Choose a partition key. I'm going to call it cloud_user, and it's going to be a string. And now we can go ahead and Create table. Now that our table has been created and the status is set to Active, we can go ahead and enable DynamoDB Streams. So click on your table. Then select Exports and streams. Scroll down to DynamoDB stream details, and you'll see it's currently disabled. So select Enable. And now, it's asking me to select a view type, so which versions of the changed items you would like to push to the DynamoDB stream. Is it going to be the key attributes only, the new item as it appears after it was changed, the old item as it appears before the change, or both new and old images of the changed item? And I'm going to go for new and old and Enable stream. And then, if we scroll down, you will see the that stream status is set to be Enabled. And then down here under Triggers, this is where you can use the changes in your DynamoDB table and stream to invoke a Lambda function to perform an action on your behalf. So now, let's scroll up and head over to Global tables and create a replica. So select Create replica. My current region is us-east-1, Northern Virginia. And we can replicate our table to one of these other regions. And I'm going to select US East (Ohio), which is, of course, us-east-2, and then go ahead and click on Create replica. We'll give that a moment to create, and we can refresh the view using this button. And after a few minutes, we can see that our table in US East (Ohio) is now active, and it even has its own DynamoDB endpoint in us-east-2. So now, let's enter an item in our original DynamoDB table and see how quickly we're able to review it in the other region. So on the left-hand menu, select Explore items. We're in our dynamodb-streams-test-table, and we've got no items. So select Create item, and we'll add an item into our table. The attribute name is cloud_user, so let's give it a value. I'm just going to add my name. And why not add a new attribute as well? And I'm going to add a new attribute of mobile number. The type is going to be a string. And I'll add some information in here and then Create item. So now, that item has successfully been entered into our table, which is in the Northern Virginia region. However, let's check that it's been successfully added to our table in the us-east-2 Ohio region. So using the drop-down, I'm going change region to US East (Ohio). And we should instantly be able to see the record that we just created. So we've successfully configured DynamoDB Streams with our DynamoDB table. And it is replicating the item-level changes to our table that is located in us-east-2. And if you're working in your own AWS account, just remember to delete your table and delete the stream as well so you don't get any unnecessary charges. So from my Ohio region, I'm going View table details and Delete table. Then, back in Northern Virginia, us-east-1 region, I'll select my table. Then select Exports and streams. Disable my DynamoDB stream. And then after that, I should be able to delete my table. So now, let's review our exam tips for DynamoDB Streams. And just remember that a DynamoDB stream creates a time-ordered sequence of item-level changes that can be used to replicate changes on your table to another DynamoDB table, that is located in a region of your choice. And this allows you to

configure multi-region redundancy, which is going to be great for disaster recovery and high availability for mission-critical applications, as well as enabling you to create globally distributed applications. And the really cool thing about it is that it does all of this pretty much instantaneously with a replication latency of under 1 second. So that is all for this demo. Any questions, please let me know. And if you're ready to move on, please join us in the next lesson. Thank you.

Section Review: Reliability and Business Continuity Summary - Part 1

Hello, Cloud Gurus, and welcome to this section summary on reliability and business continuity. This lecture is broken into 2 parts where we'll cover the exam tips, which were mentioned and covered in this section. Let's go ahead and get started. We began this section by discussing elasticity and scalability through a real-world scenario of the gym. So for our elasticity, remember our friend here who's stretching out this rubber band to meet his desired level of resistance. He's stretching outward to increase his resistance capacity and inward to reduce it. So keep that in mind as the same meaning is for the technical sense. We scale with demand for elasticity and keep in mind that this is for a short-term period. So let's say we have an application that's having a lot of spikes in traffic. We would introduce elasticity to be able to handle those spikes. On the other hand, we also discussed scalability, which we said, we go to the gym every day and over time, our maximum capacity to be able to lift heavier and heavier weight increases, and that's the same from a technical sense, where we scale out our infrastructure on a longer-term basis. Keep these two, elasticity and scalability, in mind and also remember the gym example which will help you understand these two. We also discussed AWS Auto Scaling, which is the service which helps you scale your resources according to scaling plans, which is just generally a set of directions for scaling your resources. We also discussed the 2 types of scaling that AWS Auto Scaling offers. First, dynamic scaling, which allows you to set a threshold of capacity at which you would like to scale your resources, as well as predictive auto-scaling, which uses forecasting and machine learning to scale your resources, and we also walked through a hands-on demo, which allowed you to create and maintain AWS Auto Scaling plans. Keep those things in mind that you'll need to understand for your SysOps exam. Also, through this demo, you'll need to remember the difference between Auto Scaling groups and Auto Scaling plans. So keep in mind that Auto Scaling groups automatically scales your EC2 instances through the EC2 service while Auto Scaling plans allows you to scale EC2, ECS, which is the Elastic Container Service, DynamoDB, Aurora, through tagging, CloudFormation and Auto Scaling groups within the AWS Auto Scaling service and remember, AWS Auto Scaling allows you to optimize cost and or performance. Additional exam tips for Auto Scaling groups is that you can troubleshoot EC2 Auto Scaling groups through the EC2 console or CLI. You can also monitor Auto Scaling plans through the service, AWS Auto Scaling. So understand what you're being asked, as it relates to Auto Scaling plans, which would be through AWS Auto Scaling, or Auto Scaling groups, which would be through EC2. We also covered vertical versus horizontal scaling. Remember our demonstration with our friends. We're planning a trip, and some more friends were invited. So vertical scaling was for the bigger house. So remember that if you vertical-scaled or, in more simplistic terms, scale up, you increase machine or system capacity. So this would be in including the disk IO, storage, CPU or memory. However, horizontal scaling, where we said we would get 2 houses which would handle the same load, which would be horizontal scaling or scaling out, and in this particular example, we said that this would be adding additional instances to share the load, and an example of this would be adding additional instances or nodes, adding load balancers, using Auto Scaling groups or making your application Multi-Availability-Zone. We also discussed AWS

ElastiCache, which is AWS's in-memory caching, and we discussed these 4 metrics to monitor, as it relates to your ElastiCache, CPU utilization and how you can scale out by adding more nodes if you're ElastiCache exceeds the CPU. We discussed swap usage and reallocating memory to your application. We also discussed evictions, which is the removal of expired items from the cache and how, if you're running into too many evictions, you can either add additional nodes or increase the size of your nodes to accommodate more data, and lastly if you're having concurrent connection issues, you may need to check your application for errors with communicating to ElastiCache. Also, we discussed that if you're given this particular scenario where a particular database is under a lot of stress or load, ElastiCache would be a good choice if your database is read-heavy and not prone to frequent changes, and Redshift would be a good choice if your database is stressed due to OLAP transactions. We also discussed Aurora within this section, and we noted that Aurora comes in 2 flavors, which is Aurora and Aurora Serverless. We also discussed that for redundancy, Aurora places 2 copies of your data, which are contained in 3 separate Availability Zones, which brings a total of 6 data copies. Aurora storage is also self-healing, which data blocks and disk are continuously scanned for errors and automatically repaired. With Aurora, we understood that if Aurora is 100% utilized in writing, you can scale up by increasing the instance size. If you're having 100% utilization with reading, you can scale out by increasing the number of read replicas. We also understood the key architecture of Aurora through understanding the Aurora database cluster and understood how Aurora spreads out its resources and replicas within different Availability Zones as well as how the data copies creates the cluster volumes. You need to keep that in mind as you go into your exam, just in case you're questioned on Aurora. As it relates to RDS in Multi-Availability-Zone Failover, keep in mind that read replicas are used to scale. Also, you can force a failover from one Availability Zone to another by rebooting an instance. This can be done through the AWS Management Console or through the Reboot Instance API call. A couple other things to keep in mind for RDS in Multi-Availability-Zone Failover is RDS Multi-AZ is not a scaling solution, and Amazon also handles the failover for you by updating the private DNS for the database endpoint, and backup and restores are taken from the secondary Multi-Availability-Zone instance. This concludes part 1 of the exam tips covered in this section. If you're ready to move forward, please join me in the next video.

Section Review: Reliability and Business Continuity Summary - Part 2

Hello, Cloud Gurus, and welcome back to part 2 of this section review on reliability and business continuity summary. Let's go ahead and get started. We covered in this section RDS and read replicas, and here's some important facts to remember as you go into your exam as it relates to RDS and read replicas. First, you can have up to 5 read replicas for MySQL, PostgreSQL, SQL Server, and MariaDB. You can also have read replicas in different regions for all of the engines previously mentioned. Also, replication is asynchronously only, not synchronous. Read replicas can be built off of multi-AZ databases as well. Also keep in mind that read replicas themselves can now be multi-availability zone. Also, you can now have read replicas of read replicas but beware of latency lags as data going from your read replica to another read replica may take a while for your application to update. Also, database snapshots and automated backups cannot be taken of read replicas. And a key metric to look for is replica lag. And finally, know the difference between read replicas and multi-availability zone. Read replicas are for high availability. Multi-availability zone is for disaster recovery. That's important to keep in mind. For creating and encrypting snapshots exam tips, you'll need to keep in mind these important details. You can use the AWS Management Console, the AWS

CLI, or the Amazon RDS API to share the encrypted snapshot with other accounts. Also, when sharing encrypted snapshots across accounts, you must share the custom KMS key in order for this shared account to have access to that encryption. And finally, take RDS snapshots within RDS using automated AWS backups. So using the AWS Backup service is great for creating RDS snapshots. We also discussed implementing fault tolerant workloads using Amazon Elastic File System or EFS, and here are some exam tips as it relates to EFS. Keep in mind, you'll need to understand bursting throughput, which allows the file system's throughput to scale as the data on the system grows, and this scales with the file system size, as well as uses the burst credit system to determine burst. Then we also have our provisioned throughput, which allows you to instantly provision throughput of the file system independent of the file system size. And this is fixed at a specified amount. Keep in mind, with provisioned throughput, you can increase at any time but only decrease throughput 24 hours after the most recent decrease. Some exam tips as it relates to encryption with EFS. Remember, for encryption at rest, data at rest can be encrypted through the AWS Console using the Key Management Service. However, encryption in transit must be encrypted by enable Transport Layer Security or TLS through the EFS mount helper. For building fault tolerance with elastic IPs, remember, there's 5 elastic IPs per region. And remember, elastic IPs are public at static IPv4 addresses. And the primary purpose of elastic IPs are the ability to remap network traffic in the case of a failure. We also discussed readying for disaster recovery and some information you'll need to understand is RPO and RTO. Remember, with RPO, that's Recovery Point Objective, how much data can my application afford to lose? RTO is Recovery Time Objective, how long can I afford for my application to be down? Remember and understand backups versus snapshots per use case as that'll be important to understanding disaster recovery and the questions you'll be asked on the SysOps exam as well. For our AWS service maintenance windows, keep in mind which services have Maintenance Windows, and these would include Amazon RDS, Elasticache, Redshift, Neptune, as well as DocumentDB. For configuring S3 cross-region replication, cross-region replication in S3 is a great way to introduce high availability within your architecture, and here's a couple reasons why you might want to do that. First, to meet compliance objectives. It's a perfect way for applications that have compliance standards requiring data to be geographically spaced out. Also, it's a great idea to increase efficiency within your application. For architectures with cross-regional compute, replicating data would assist with application efficiency. And lastly, S3 cross-region replication improves latency. Enabling cross-region replication improves application latency through making data closer to the consumer. For exam tips on implementing loosely coupled architectures with SQS, we covered what's SQS, and we discussed that SQS is AWS's fully managed message queuing service, which creates queues that producers use to store messages, and wait for consumers to pull and receive them. We also discussed a unique tool that you can use within your architectures called decoupling, as well as why you should decouple. And these are the benefits of decoupling. It first allows you to simplify your application updating, as well as increase the autonomous components within your application, and finally, it allows you to create non-blocking operations within your application. We also discussed the process of automating EBS snapshots using Data Lifecycle Manager, and Data Lifecycle Manager is a feature which allows you to automate the creation, retention, and deletion of EBS snapshots and EBS-backed AMIs through what's called lifecycle policies, and Data Lifecycle Manager also allows you to encrypt your snapshots using KMS, as well as enable your snapshots for cross-account sharing, as well as fast snapshot restore. We also discussed how to use DynamoDB streams to back up your table to another region, and here's your exam tips for that demo. DynamoDB is a multi-master, multi-region replication database and it allows the data within this database to be globally distributed for your

application, and the portion of DynamoDB, which replicates your data cross region is based on DynamoDB Streams. And this particular feature offer multi-region redundancy, which is great for disaster recovery or high availability. However, there's no application rewrites. And also, DynamoDB Streams allow you to replicate your data with latency under one second. So that's all for the exam tips for this section on reliability and business continuity. If you're ready to move on, please join us in the next section.

Security and Compliance

Section Introduction

Hello, Cloud Gurus, and welcome back to this section, Introduction on Security and Compliance. In this section, we'll cover several topics that's covered in Domain 4 of the SysOps exam. The AWS services and features that we'll discuss within this section are all centered around helping you build and manage security and compliance policies, as well as implementing data and infrastructure protection strategies. We'll start by understanding more about what is compliance on AWS and some popular controls that you should be aware of. We'll talk through understanding DDOS, as well as understand the AWS Marketplace security products. We'll do some refreshing on identity and access management, as well as talk through multi-factor authentication. We'll understand AWS Inspector as well as Trusted Advisors, AWS Organizations, as well as AWS Single Sign-On. We'll get a better understanding of service control policies as well as how to secure multiple accounts using AWS Control Tower and organizations. We'll talk through STS, or Security Token Service, as well as understand AWS Key Management Service. We'll talk through AWS Certificate Manager as well as AWS WAF for application firewall, as well as AWS Shield and Hypervisors. In this lesson, we'll also talk through AWS Systems Manager Parameter Store and when it should be used, we'll understand AWS Service Limits. We'll also review the AWS Shared Responsibility Model. From a security perspective, we'll talk about how to protect your logs with Cloud Trail as well as AWS Security Hub, Guard Duty, and understand security reporting within AWS, and we'll wrap this section up with the security and compliance summary. Now I know that was a mouthful, but there's no better way than to jump in and get started. So if you're ready to start your learnings on AWS security and compliance, joining me in the next lecture.

Compliance on AWS

Hello, Cloud Gurus and welcome to this lecture on compliance on AWS. In this lecture, we'll get a better understanding of what is compliance on AWS. We'll also discuss some popular and well-known compliance frameworks. We'll discuss PCI DSS v3.2, and how the steps in this particular framework contributes to a more secure architecture. We'll also discuss several other frameworks, and then we'll conclude with understanding FedRAMP on AWS. And as always, we'll wrap up with our exam tips. So let's go ahead and get started. Compliance on AWS. So what is compliance on AWS? It's simply a set of controls, standards, and frameworks, which help customers ensure security and compliance requirements on AWS. And this contains a couple of things. First, it contains the shared responsibility model (which we'll have a moment later in this section to discuss in greater detail). But simply, the shared responsibility model is AWS's guides on which parts of its infrastructure they claim responsibility for and which parts you should be responsible for. So that's one part of compliance on AWS and another component is the compliance frameworks, some of which AWS has already secured for its particular infrastructure. However, you'll need another level

of compliance for some of the applications and architecture or work that you complete within AWS. So those 2 are the main key factors of compliance in AWS. And we'll talk a lot more in detail about this. However, for more information, I definitely suggest you check out the Risk and Compliance white paper, and I'll go ahead and have that link in the resources section to the left. As it relates to compliance frameworks, one compliance framework that you'll need to understand at a very high level is ISO 27001. And this is a compliance that's given from the International Organization for Standardization, as well as the International Electrotechnical Commission. And in order to be compliant within this particular framework, you'll need to get a third-party audit, which can certify that your system is compliant within what ISO 27001 specifies. And what they specify is the requirements for establishing, implementing, operating, monitoring, reviewing, maintaining, and approving a documented information security management system within the context of the organization's overall business risk. The way to be ISO compliant is to pretty much build a system and then make sure that all of your components within that system are compliant according to ISO 27001. One thing that is great know is that AWS itself is ISO 27001 compliant. However, that does not automatically make anything that you build on top of AWS compliant within this framework. So please keep that in mind. Another well-known compliance framework is the Health Insurance Portability and Accountability Act, otherwise known as HIPAA. And this particular compliance primarily ensures the confidentiality as well as the security of healthcare information. So if you're building any healthcare applications that deal with patient data or patient health information, you'll definitely want to understand HIPAA compliance as there can be severe penalties for not complying your applications with this particular compliance. Another compliance that's good to know and understand would be the Payment Card Industry Data Security Standard. This is a particular compliance around e-commerce businesses, which accept payments over their website. And what this standard is, is basically a set of policies and procedures intended to increase security around credit and debit transactions. And the Payment Card Industry Data Security Standard was founded by major credit card companies, Visa, Discover, American Express, and MasterCard. And with Payment Card Industry Data Security Standard, there's 12 requirements that PCI requires every compliant vendor to have. And we could actually go through those now, so you can get a better understanding of them. For the 12 requirements of PCI DSS compliance, starting with #1, would be to install and maintain a firewall configuration to protect cardholder data. This would include the use of AWS web application firewalls, security group, subnets, all of the AWS features you can use to make sure that your data is secured. #2, using password protection. Do not use a vendor supplied default system password or other security parameters. Making sure you aren't just simply using admin or password as a username or password. #3, protect stored cardholder data. This means if your data is at rest within the database, making sure that that RDS instance is encrypted at rest. #4, encrypt transmission of cardholder data across open public networks. This would be requiring SSL certificates to make sure data is encrypted in transmission. Requirement #5, use and regularly update anti-virus software or programs. Requirement #6, develop and maintain secure systems and applications. Requirement #7, restrict access to cardholder data by business need-to-know. This can be done through using identity and access management, using specific roles and groups to limit access to cardholder data, and granting access based on a least privileged basis. Also, #8, assign a unique ID to each person with computer access. Requirement #9, restrict physical access to cardholder data. This one probably isn't as prevalent anymore as we don't have a lot of businesses printing out cardholder information. However, if you do, keep this particular requirement in mind. #10, track and monitor all access to network resources and cardholder data. A good use case for this would be CloudTrail, monitoring those API logs, as well as CloudWatch, a security components

such as GuardDuty-- just using all of the AWS services possible to make sure we're keeping track over who's entering the network and who may be possibly reaching your cardholder data. #11, regularly test security systems and processes. And #12, maintain a policy that addresses information security for all personnel. So those are the 12 requirements of PCI DSS v3.2. Now let's discuss several other frameworks that won't necessarily appear on your SysOps exam, but it'll be good to know. Starting with FIPS 140-2, which is a US government computer security standard used to approve cryptographic modules rated from a level 1 to a level 4, with level four being the highest security and Cloud HSM meets the level 3 security standard. Another one would be SOC1, which is a service organization control around accounting standards. SAS70, which would be a statement on auditing standards, #70, as well as FISMA, which is the Federal Information Security Modernization Act. Overall, key takeaway is just understand the compliance standards which would be required within your specific area of work. The last thing we'll discuss for compliance on AWS is the Federal Risk and Authorization Management Program, or simply stated FedRAMP, which is an initiative which empowers the federal government to obtain objectives through innovation and cloud computing. Basically, this is a compliance which allows federal government entities or entities, which is contracted by the federal government to do so safely as it relates to data and security standards. And AWS does have a FedRAMP offering. And that FedRAMP offering is AWS's GovCloud, which AWS makes its services available for these type of vendors. These particular services have more restrictions and are limited to certain regions, availability zones, as well. For our Compliance on AWS exam tips, just remember the compliance frameworks that we covered, including ISO 27001. Remember our Payment Card Industry Data Security Standard and the requirements that we discussed as well as the Health Insurance Portability and Accountability Act, or HIPAA. You won't necessarily be quizzed on these, but just understand from a compliance perspective what compliance on AWS entails. Lastly, keep in mind the Federal Risk and Authorization Management Program or FedRAMP--the initiative that empowers the federal government to obtain objectives through innovation and cloud computing. So if you're asked the question about how to help a federal government move into AWS in a more secure fashion, AWS GovCloud through FedRAMP would be a viable answer. So that's all for compliance on AWS. If you're ready to move forward, join me in the next lecture.

Understanding Distributed Denial of Service (DDoS)

Hello, Cloud Gurus, and welcome back to this lecture on understanding Distributed Denial of Service or DDoS. In this lecture, we'll help you understand DDoS through understanding first, what is a DDoS attack? We'll discuss amplification and reflection attacks as well as NTP amplification. We'll talk through application attacks and talk through some AWS services which can help you mitigate DDoS attacks. We'll also talk through AWS Shield and then we'll conclude with some exam tips. So let's go ahead and jump in. What is a DDoS attack? A Distributed Denial of Service or DDoS attack is an attack that attempts to make your website or application unavailable to end users and this can be achieved by multiple mechanisms, such as large packet floods using a combination of reflection and amplification techniques, or by using large botnets. One type of DDoS attack that you should be aware of is an amplification or reflection attack and this type of attack is generally where an attacker may send a third party server such as an NTP server or a network time protocol server a request using a spoofed IP address. And basically what this is, they send a request to a server acting as if it's your IP address and they receive a larger payload than what they send, ultimately causing damage or destruction to your applications load. And these attacks can include

NTP, SSDP, DNS, Chargen, SNMP attacks, etc. Now let's see how that works. For an NTP amplification attack. Let's say you have a hacker whose IP address is 190.1 .2 .3 and then you have a victim, or the victim's IP address, which is 245.1 .2 .3. What the hacker does is provide a spoofed source, which is the victim's IP address and sets the destination to be the NTP server. He'll send 64 bytes over to the NTP server, which will respond with 3, 456 bites which will ultimately overload the victim's IP address. Another type of application attack would be a Layer 7 attack where an attacker would send a flood of GET request over to a web server and overload the web server to where it cannot respond to all the requests that it's receiving and it would eventually slow down. There's another attack, which is very similar, which is called a Slowloris attack where an attacker opens multiple connections with the web server and never completes the request that they're sending and it eventually causes the application to max out on the number of open connections and the application then can't service authentic requests from its users. Now that we have a better understanding of DDoS attacks, let's talk through how we can mitigate them using these 5 steps. First, we can mitigate them through minimizing the attack surface area and we can do this through utilizing AWS Web Application Firewalls to make sure that we're monitoring and creating rules where possible for our web application traffic. We can also be ready to scale to absorb the attack and this is done through auto-scaling groups as well as auto-scaling plans to make sure that there's enough resources available in the case that we have an attack which is maxing out our application CPU utilization. We can also safeguard exposed resources through the use of additional AWS security services, which we'll talk through in a moment and throughout this section. We can also do this through learning our applications' normal behavior and understanding abnormal behavior, which through understanding abnormal behavior we can create a plan for the attacks. With that in mind, there's several AWS services, which helps us in mitigating DDoS attacks on our AWS account, services and resources. There's Amazon Shield, which we'll talk about a great deal within this lecture. There's also the Amazon Web Application Firewall, which has been paired with Amazon Shield and both of these service work together in helping us mitigate DDoS attacks, which monitors and filters unwanted web traffic to our AWS resources. And also Amazon CloudWatch, which provides us a greater level of visibility into our accounts and resource through reporting, alerting as well as metrics. One service in AWS that's key in helping us mitigate DDoS attacks would be AWS Shield. And it's an AWS service, which protects all AWS customers on Elastic Load Balancing, Amazon CloudFront, and Route 53. AWS Shield also protects against SYN, UDP floods, reflection attacks, as well as other Layer 3 and Layer 4 attacks. The advanced version of AWS Shield provides an enhanced protection for your applications running on Elastic Load Balancing, Amazon CloudFront and Route 53 against larger and more sophisticated attacks. However, this comes at a price tag of $3, 000 per month but also comes with additional support and protection for your applications. So for our DDoS exam tips, the main thing you'll need to remember is the technologies you can use to mitigate a DDoS attack and this is Amazon CloudFront, Amazon Route 53, Amazon CloudWatch, AWS WAF, AWS Shield, Elastic Load Balancing as well as AWS Auto Scaling and also be sure to check out the AWS DDoS White Paper as it contains a lot of great information you'll need to better understand how to mitigate these DDoS attacks within your architectures and not just on the SysOps exam. That's all for this lecture. If you're ready to move on, join me in the next lesson.

AWS Marketplace Security Products

Hello, Cloud Gurus, and welcome back to this lecture on AWS Marketplace security products. In this lesson, we'll talk through the services and products available through the AWS Marketplace, to help you introduce security to your architecture. We'll talk through the Marketplace security products, as well as the Center for Internet Security. We'll also have a brief tour of the Marketplace, and finally conclude with some exam tips. Let's go ahead and get started. The AWS Marketplace allows you to purchase security products or solutions that's been pre-baked within EC2 AMIs to help you introduce additional security within your architecture. These security solutions range from application security, which allows you to fully manage security throughout the life cycle of your applications, for example, through your CICD pipelines. Also through cloud infrastructure security, where you can discover, and inspect, and protect your workload's containers as well as cloud-native applications. These services also cover WAF and Edge security, where you can shield your applications as well as Edge devices from cyber threats that could affect your availability and security. Also, endpoint security, which help protect your endpoints from exploits, incidents, or data loss. These solutions also assist with identity and access management, where these third-party tools are allowing you to grant and manage user access to the right resources, while monitoring and enforcing your IAM policies. And finally, firewall, which helps you enhance your network security through deep packet inspection, as well as intrusion prevention and detection. Now, let's talk a little bit about the procurement process for the AWS security products, first, starting with its licensing. Licensing for vendor products can be procured through the AWS Marketplace, or negotiated directly through the vendors. And for the cost for these products, the pricing model ranges from product to product. Pricing for the security products ranges from free, hourly, monthly, annually; or in those cases where you've negotiated through the vendor directly, you can bring your own license. And lastly, for the support of the Marketplace security products, Marketplace security products offer direct vendor support if installed on-premises. However, increased support may be available through vendor offerings, so you definitely want to be sure to understand what level of support comes with the products or services you decide to use. One vendor that it's good to familiarize yourself with, which necessarily isn't knowledge you'll need for the Sysops exam, but it's good to know the Center of Internet Security. This particular vendor offers OS hardening on multiple EC2 operating systems. These includes Amazon Linux 2, Red Hat, as well as Ubuntu, and Microsoft Windows Server. And all of these images are available for you to buy within the AWS Marketplace. So, let's go ahead and take a look within the AWS Marketplace, and we'll conclude with our exam tips. From the AWS Management Console, we can get to Marketplace Products a couple ways. We can either go to it in our recently visited services, or we can type it in the search bar at the top, which I'll just type Marketplace. Then I'll click AWS Marketplace Subscriptions. And then, in the left pane here, I can open this, and go down to Discover products. And here, we can see that there's thousands of results here. We can search for a product by name. However, to get to the security products, we'll just go to "Infrastructure Software, " and then we'll go down and we see that there's 1, 797 security products at the time of this recording. So, we'll click those products. And we can scroll through the products and see them one by one, or we can continue to filter by first delivery method, based upon whether it's an AMI, or it's a Software as a Service, some cloud formation container, or professional services. We can search by vendor, the pricing plan that we discussed as well, or by operating system, the type of support they provide, or for those particular products that may have a free trial. Also, keep in mind, if you want to search for a vendor one by one, you can do that. For example, what we talked about earlier with the Center for Internet Security, here are those

AMIs which have operating systems which have been hardened. You can select them and use them if needed. And also, shameless plug; A Cloud Guru is also within the AWS Marketplace. So, that's generally the AWS Marketplace. So, we'll go back to wrap up with our exam tips. So, for our AWS Marketplace security products exam tip, you may be asked what particular option you may utilize to introduce additional security within your architecture. And just be aware and understand that purchasing a third-party security tool within the AWS Marketplace is a way to introduce additional security within an architecture. So, that's all for AWS Marketplace security products. So, if you're ready to move forward, join me in the next lecture.

IAM Refresh

Hello, Cloud Gurus. And welcome back to this lecture on IAM Refresh. In this lecture, we'll be covering - at a very high level - the basics of Identity and Access Management. If you're taking this course, you've probably already completed either the Cloud Practitioner or the Solution Architect certification, So all these concepts will be covered at a very high level. We'll talk through Identity and Access Management, IAM basics, as well as users, groups, and roles. We'll talk through an IAM policy, and we'll wrap up with some exam tips. Let's go ahead and get started. Identity and access management allows us to manage users and their level of access within AWS. This is very similar to when we check in a hotel and we're given a key with all of our permissions and abilities to enter into the amenities that we've paid for. It is important to understand IAM and how it works, both for the exam and for administering a company's AWS account in real life. Here's several IAM basics that is good to understand. First that Identity and Access Management through IAM is centralized, offering control of your AWS account from one place. It's also in Identity and Access Management that you're able to give access, which is shared, to your AWS account, as well as a place where permissions are given, granular permissions to which AWS services and which subset of functions within that services are available. And IAM also offers Identity Federation which supports well known identity providers, such as Active Directory, Facebook, as well as LinkedIn. Identity and Access Management also provides first multifactor authentication, which provides us with increased security for our AWS account settings and resources. We can also create temporary access within IAM, which provides temporary access for users, devices, and services as needed. Within IAM we can create password policies, which allows us to set up our own password rotation policy. IAM is also integrated, which allows us to integrate with many different AWS services. And it also provides us with compliance, which supports PCI DSS compliance. A key component of Identity and Access Management that you'll also need to understand is users, groups, and roles. Starting with users. We think of end users or people that we would like to grant access to. And those people are assigned to groups which are a collection of users under one set of permissions. We also have roles where we can create and assign roles to users, applications, and services to give access to AWS resources. Now let's talk about an IAM policy, which is a document that defines one or more permissions. An IAM policy can be attached to a user, group, or role and also keep in mind that policies attached to resources are called resource policies. To conclude, for IAM refresh exam tips, you'll need to keep in mind the basics of identity and access management, which are users, groups, and roles and how you can grant access to them and through them to AWS resources. Also keep in mind that IAM is centralized control of your AWS account through providing access through permissions and also configuring Identity Federation. Lastly, keep in mind these several things that Identity and Access Management provide such as multifactor authentication, temporary access, password policies,

integration with multiple AWS services, as well as compliance. That's all for this lecture. So if you're ready to move on, join me in the next lesson.

Demo: Creating Custom IAM Policies and Roles

Hello, Cloud Gurus, and welcome back to this demo where we'll be creating custom IAM policies and roles. Before we jump in, let's cover our lesson objectives and some steps you can expect to take throughout this demo. We'll first get started by creating an IAM policy for S3 access. We'll then create an IAM role for our EC2 instance, and we'll attach the policy we previously created. We'll then create an EC2 instance using the created role which will have access to our S3 bucket. And finally we'll test and update our IAM roles and policies using the AWS CLI. Feel free to follow along using your cloud playground account or using your personal AWS account. If you're ready to move forward, join me in the AWS console. Now that we're here in the AWS console, we'll need to head over to Identity and Access Management which we can do by either typing it in the search for services bar here at the top. Our Recently Visited Services, or if we expand All Services and head down to Security, Identity, and Compliance, we can see IAM here, so I'll go ahead and give that a click. And here we are on the main page of Identity and Access Management. Here in the left pane if I scroll down, I can see Policies. I'll just click that. And here we can see that there's several policies that are available to us that are already AWS managed, as well as some customer managed policy. So what I want to do is I want to click Create Policy, and I can do so using the visual editor or I can simply paste some JSON from a previously created policy in here. However, we'll just go through and use the Visual Editor. So the first thing I'll need to do here is choose a service. So I'll click choose service and I'll type in S3, and I'll click on S3. And here from S3 I can now select which S3 actions I would like to grant within this policy. I can either select here for All S3 Actions, or I can go through each access level and define the permissions that I would like. However, for this policy, what we'll go ahead and do is we'll give List and Read access. The next thing we'll need to do is select a Resource. So I'll open this here and I'll select which resources I would like to grant access to So for this particular policy, I'd like to grant this access to All Resources and here, keep in mind also, you can add request conditions, so you can attach multifactor authentication or you can limit the access to a particular source IP. But for now we won't add any of those permissions, so we'll click next. We'll bypass tagging for the moment, and we'll go to review our policy. So here we'll give our policy a name and let's just call it ACG-Test-List-Read-Policy and then I'll give it a description, "this policy grants list and read access to S3 for all resources, " or something like that and now I'll go ahead to the bottom and click Create Policy. So now within my policies, if I click on ACG-Test-List-Read-Policy which I can also search for it here if I can't find it, I'll just click on it. And now I can see that I have S3 Full List and Full Read for all resources. So my policy has been successfully created. Now we'll need to create a role for our EC2 instance and attach this policy. So just above Policies here in the left pane, we can see Roles. So I'll click Roles and I can also see here that there's several roles that's already been created. However, I'd like to go ahead and create one specifically for my EC2 instance. So I'll click Create Role, and here I can select the type of role or the trusted entity I'd like to create. I'd like to create it for an AWS Service, which is already selected, and we'll choose a common use case here which is our EC2 instance. Keep in mind, you can follow this same process for several other AWS services, but I'll just go ahead and click EC2 Instance, and I'll click Next. And now we can attach our policy, so what I'll do is I'll type in ACG, and here's my ACG-Test-List-Read-Policy. So I'll go ahead and click our check box right there, and I'll click Next. We'll bypass tagging for now. And I need to give my role a name. For the role name I will give it

ACG-Ec2-Read-List-Role, and I'll go ahead and click Create Role. So now my role has been created and if I click into the role I can see that it has attached to it our ACG-Test-List-Read-Policy for S3. So now my role has been created, and my policy has been attached. So we'll head over to EC2 to go ahead and create our EC2 instance which will use this role. To do that I'll go back to the AWS console main page and I'll expand All Services and here under Compute, I'll click on EC2. And I can just scroll down here to Launch Instance, I'll click Launch Instance and here I'll need to choose my AMI. For my AMI I'll select Amazon Linux 2 AMI and I'll click Select. Here within the Choose an Instance Type I'll go ahead and select our Free Tier Eligible instance which is already selected for me. And I'll go ahead and click Next to configure instance details, and so we'll keep all of these default configurations the same until we get to IAM role. And I'll expand this and change this to our ACG-Ec2-Read-List-Role. And I'll go ahead and click Next. We'll keep the storage the same. I'll click Next. We'll bypass tagging for now, keep our same security groups. We'll allow AWS to create new security groups on our behalf here, so we can click Next. And we're OK with these details. This is the AMI that I selected, the sizing of t2 micro, as well as our security group gives us the ability to be able to ssh in from anywhere, which we're good with that and I'll go ahead and click Launch. I'll need to go ahead and create a new key pair. So I'll create a new key pair, and keep in mind that you'll need to hang on to this key pair for the next step which will be logging in to the EC2 instance. So I'll just make this my acg-test-keys, I'll download the key pair, and then I'll click Launch Instance. We'll give that a second to finish, which it's now complete. If I click my instance ID, I can see here that the instance state is pending, so we'll give this video just a moment. We'll pause the video and allow the EC2 instance to spin up. Now that our EC2 instance has been created, let's go ahead and create our S3 bucket. Then we'll ssh or log on to our EC2 instance to be able to test our policy. So we'll go back to the AWS console, and then under All Services we'll scroll down to Storage. We'll click S3 and then under Buckets, we'll just click Create Bucket. We'll give our bucket a name, testing-acg-policy, and then a random string of numbers. I'll scroll down and I'll click Create Bucket. My bucket has been successfully created. So now let's ssh into our EC2 instance. We'll go back to EC2. We'll go to our Instances. I'll select this instance, and I'll go over to Connect. And I'm going to use the EC2 Instance Connect to connect to my EC2 instance. So I'll just click Connect, and we're logged in to our EC2 instance using AWS CloudShell. So what I'm going to do is I'm going to type aws s3 ls and I can see my bucket that I've created, testing-acg-policy. Now I'm going to create a small file that I'll try to copy over to my S3 bucket. So I'll just simply say echo "This is a test" > hello.txt and that worked. I'll do a ls, and I can see my file is hello.txt and it's there. So now I'm going to try to copy it over to my S3 bucket using this command aws s3 cp, or copy, /home/ec2-user/ hello.txt. I want to copy that to my S3 bucket, which is s3://testing-acg-policy and then my random strings. Then when I hit Enter, I should have a access denied because when we created the bucket we did not give write access. So now let's go back over to IAM and grant access and we'll retry this command. Now in a separate window, I'll go back to the AWS management console and I'll click on IAM, I'll click on Policies to go back to my policy. I'll find my ACG-Test-list-Read-Policy. I'll click on it there, and now I'll click Edit Policy. I'll expand S3, and then under Actions I'll expand the Actions here as well. I'll just simply add write policies, and then I'll review my policy. And after I've validated that I now have full list, read, and write access, I'll save those changes. And now I'll go back to my terminal to test to see if those changes have taken effect. Now that I'm back in the terminal, I'll just arrow up to get back to my command where I originally tried to copy the file over, and I'll hit Enter, and now I can see that I've successfully uploaded a file. So we can see here that our changes that we made to our IAM policy are instant and allowed us to write to our S3 bucket. Now let's go back and review the key learnings that you'll

need to take away from this into your sysops exam for exam tips. First, understand how to create custom policies using the visual editor, or JSON, also understanding that you can use the AWS CLI or the console to attach roles to the EC2 instance at any time. Also understand that the changes that you make within IAM take effect immediately. That's all for this demo. If you're ready to move on, join me in the next demo.

Demo: Enabling MFA and Reporting with IAM

Hello Cloud Gurus and welcome to this lesson, which is going to cover enabling multifactor authentication and reporting with Identity and Access Management. Before we begin the demo, let's quickly review how multifactor authentication can be used with your AWS account. Now multi-factor authentication can be enabled for access using the AWS console, and we can enable it for the root and user accounts. And when it's enabled, the console is going to prompt you to provide your MFA token at the time of sign-in in addition to your password. So it's giving you an extra layer of security when signing into your AWS account using the AWS console. So let's take a look at our lesson objectives. And we're going to get started by reviewing how to configure MFA for the root account. Next, we're going to create an IAM user without MFA enabled. And then we're going to download and understand the IAM credential report, and this is a report provided by IAM, which is going to tell us which of our users have MFA enabled. And we'll be able to clearly see which users are using MFA and which are not. Now one thing to note is that configuring MFA for the root account is not going to be available in the cloud sandbox. You will need to use your own AWS account. So you'll need to create your own free tier account in order to practice enabling MFA. And I've actually created a brand new account this morning so that I can show you how it all works. So if you're ready to get started, please join me in the AWS console. Now from the console, remember this is in your own AWS account, let's navigate to Identity and Access Management. And the one thing that we can see right away is that we've got a security recommendation to add MFA for our root user account. So the root account does not have multi-factor authentication enabled by default. So let's go ahead and add MFA. Select Activate MFA, and I've got a few different options here. So first of all, there's the virtual MFA device, which is an authenticator app installed on your mobile phone device. You can authenticate using a security key, so a FIDO security key like Yubikey. Or alternatively, use another hardware MFA device. But we're going to stick with the virtual MFA device, and I like to use Google Authenticator on my smartphone. So select Continue. If you want to a list of compatible apps for your mobile device, you can select this link here and download the app to your mobile device. Next, we need to use the virtual MFA app and our device's camera to scan a QR code. And we need to click here to show the QR code. At this point, I need to open up the virtual MFA app on my smartphone. So that's going to be the Google Authenticator. So from my app, I'm going to scan this QR code that appears on the screen. And then the next step is we need to type two consecutive MFA codes in the boxes below, and these are the codes that are being generated by Google Authenticator. Here's my first code. And now I need to wait for it to refresh and give me another code, and there's my second code. And after that, I can select Assign MFA. And if it's been successful, you should get a message like this. So now, if we head back to our IAM dashboard, we can see that our root user now has MFA. So now, if I log out of my account, select Sign out, and then log back in again, provide my password, now it's asking me to provide the MFA code from the app on my phone. Provide the code. And only after adding the code is it going to let me log into the account. So you can see, it's giving a second layer of protection for my account credentials. So now, let's go ahead and create an IAM user. So I come back to IAM, select Users,

Add user. My user is going be robbie with programmatic access. Hit Next. We'll create a group for our user. The group name is going be admin, and we'll give them administrator access and Create group. Hit Next, Next: Review and Create user. Close that down. So now, our user has been created, and we've given our user programmatic access, but we haven't enabled MFA. And now for the last thing, we'll go over to the credential report. So using the menu on the left, you'll find it here. So select Credential report, and this is a report that's going to list all of your IAM users in the account and the status of their various credentials, including whether or not MFA is enabled. So select Download credentials report, and it's going to download it in CSV format to your local machine. And here's the report. And if we check column H where it says mfa_active, we'll see that for the root account, mfa_active is true. And for our user, Robbie, mfa_active is false. So this report is really useful to help you see at a glance which accounts have multi-factor authentication enabled. And imagine you have hundreds of users, it's a really easy way to work it out. So now, let's review my exam tips. And just remember that MFA can be enabled to protect access through the AWS console and the CLI as well, and we can use the IAM credential report to find out which of our users have MFA enabled or not. So it's going to give us the MFA status of all our users in one easy report. So we created our report, and then we looked for the mfa_active field. And ideally, we want it to be set to TRUE for all of our users. So if that is it for this lesson. Any questions, please let me know. Otherwise, feel free to move on to the next lesson. Thank you.

Introducing AWS Single Sign-On (SSO)

Hello, Cloud Gurus, and welcome back to this lecture on introducing AWS Single Sign-On. In this lecture, we'll introduce AWS Single Sign-On through understanding federated identity. We'll answer the question "What is AWS Single Sign-On? ", speaking of the AWS service, and how AWS Single Sign-On works. We'll also discuss some AWS Single Sign-On on use cases, and we'll conclude with some exam tips. So let's go ahead and jump in. Before we can understand the problem that AWS Single Sign-On solves, let's get a better understanding of federated identity. So if you're anything like me, you probably have, let's say an email which has its own unique password, a social media account, which has its own unique password, and banking, which has its own unique password. And as your accounts grow, this can be kind of troublesome, having all these different passwords to log in to different accounts. However, one thing that Google solves, for example, is using Google Single Sign-on, or federated identity, which allows users to be able to sign on to all of their accounts with one single password. And that's what federated identity does. It allows us to sign on to multiple platforms with a single identity. So federated identity is the process of authenticating user identity across multiple systems and applications. And that's exactly what AWS Single Sign-On does for us within our AWS account. AWS Single Sign-On is a service which manages identity federation within AWS. So what is AWS Single Sign-On? Let's talk a little bit more about Single Sign-On. It's an AWS service, which centrally manages single sign-on for multiple AWS accounts and cloud applications. So AWS Single Sign-On integrates with AWS Organizations and it streamlines access and permissions across organizational units, as well as AWS accounts and other AWS API operations. And AWS Single Sign-On simplifies application sign-on, which allows us to easily manage sign-on to multiple AWS accounts, cloud applications, and SAML 2.0 applications. And the really cool thing about AWS Single Sign-On is it leverages our existing active directories, whether it's within AWS or within our data center, which we can use our existing corporate active directory service and sign-on credentials. It also allows us to streamline user and group creation. It allows us to create and manage user groups within the console by provisioning external identities. Now let's

discuss how AWS SSO works and can be implemented within your AWS account. You first got started by enabling AWS Single Sign-On, either within your AWS console or within AWS Organizations. After enabling this service, you'll then be able to select an identity source. Your identity source can be an active directory that you create within AWS, or an existing active directory that you manage. For example, a Microsoft Active Directory that is local within your data center, which can be connected through the AD Connector, which we'll talk through in just a moment. You then are able to centrally manage your permissions within AWS, giving your users access to not only AWS accounts, but your SAML applications, which you've enabled to utilize SSO. This also gives users single-click access to multiple AWS account, which totally simplifies the AWS account login process. Now let's talk through AWS use cases and some reasons why it will be beneficial to use AWS SSO. AWS SSO allows you to first, manage permissions to AWS accounts, making it very easy for your users to access the AWS console and have single-click access to the AWS accounts managed within AWS Organizations. This makes it easy for you to streamline your access through your users and groups already established within identity and access management. AWS SSO also allows you to streamline this access through your business cloud applications. For example, Dropbox or Slack. And there are several others. You can also give single sign-on access to your custom SAML applications, which you've built within, let's say EC2, Elastic Beanstalk, or any of the other AWS compute services offered. And the single sign-on access allows you to easily grant users access to any of those. AWS SSO also gives you the ability to integrate with your on-premise active directory, leveraging access to your own premise, users and groups, as well as on-premise Microsoft Active Directory using the AD Connector or the AD Trust, which can be connected through the AWS Directory Service. For our AWS Single Sign-On exam tips, you'll definitely need to remember what is federated identity. And federated identity is the process of authenticating user identity across multiple systems and applications and AWS Single Sign-On is the service which manages identity Federation within AWS. Also keep in mind that on-premise active directories can be connected using an AD Connector or by configuring an Active Directory trust within the AWS Directory Service. That's all for this lecture on AWS Single Sign-On. If you're ready to move forward, join me in the next lecture.

Auditing and Troubleshooting Access Issues

Hello Cloud Gurus and welcome to this lesson, which is going to cover auditing and troubleshooting access issues in your AWS account. So in this lesson, we're going to walk you through how to audit and troubleshoot any issues that you might be having relating to access in your AWS account. We'll review CloudTrail, IAM Access Analyzer, and the IAM Policy Simulator as well. We'll do a walkthrough of each of these services in the AWS console and then wrap up with some exam tips. So let's go ahead and get started. Now there are several AWS services that allow us to audit and troubleshoot within AWS, and one of the most important is AWS CloudTrail. And this allows us to monitor user activity and API usage. And since the AWS back end is primarily APIs, CloudTrail actually gives us an in-depth understanding of how those API are responding on the back end of our account. First of all, it records and captures CloudTrail events occurring within our AWS services. So whether we're uploading a file to S3 or creating a new user, group, or role in IAM, all of those API calls are going to be recorded and visible within CloudTrail. However, it's not immediate, and there may be a time lag to it actually reaching the log. And we can view CloudTrail events using the AWS console from within S3 if you set up a CloudTrail or within CloudWatch Logs. Additionally, through further configuration with CloudWatch, we can then decide to act upon

the events that are occurring in CloudTrail. So we can configure CloudWatch alarms and events to occur when events are detected in CloudTrail. And then, finally, we can review everything in CloudTrail. So we can check the recent events using the CloudTrail console, and you'll be able to see the recent events from the past 2 weeks. Or you can also review them in Amazon Athena if you have that set up. And if you remember, we did that earlier on in the course. So now, let's hop over to the AWS console to get a first-hand look at CloudTrail. So from the console, search for CloudTrail. And here, in the CloudTrail dashboard, there's several things that we can see. So first of all, we can see the list of any cloud trails that exist in the this account. And of course, you can create a CloudTrail log and have it store data within an S3 bucket of your choice. And you can just go ahead and create a new trail down here if you want to. The next thing you can do is observe your event history, and you'll be able to observe the event history of the last 2 weeks in this section of the CloudTrail dashboard. So if you select an event like the ConsoleLogin, we can see all the information relating to my particular login. So I'm logged in as a cloud_user. It's got my source IP address. Here's the event record. So it's got the event time, the user agent, which is all details about the browser that I've used to log in with, whether the login was successful or not, whether MFA was used or not. And down here, it's got an event type as well. So it's really useful to be able to understand how to read these event records if you're going to troubleshoot a specific access issue that you're having within AWS. Scrolling back to the dashboard, over here, you can set up CloudTrail Insights. And this is going to monitor your account for any unusual API activity. And then over here, if you select Create trail, it's really easy to create your own trail. You just give it a name. You can either have it create a new S3 bucket for you or use an existing one. You need to provide an alias for a new KMS key and then go ahead and create your trail. And of course, when you create a CloudTrail trail, it's going to keep that data indefinitely. So that is the basics of AWS CloudTrail. And then, another way to audit and troubleshoot access issues is through the use of Identity and Access Management Access Analyzer. And the Access Analyzer helps you to identify resources that have inadvertently been shared with an external entity, and it supports loads of different services including Identity and Access Management roles so you can check for any roles that are accessible by another account, S3 buckets so you can check which buckets are being shared. For Lambda, you can check functions and layers that might be shared with an external entity. You can check if any KMS keys have been shared with an external party. You can also do a check on AWS Secrets Manager to see which secrets are being shared, as well as SQS, so which queues are being shared with an external entity. So now, let's check out AWS Access Analyzer within the console. And just be aware that IAM Access Analyzer is not available in the cloud sandbox, and you will need to use your own AWS account. So if you would like to enable Access Analyzer, you will need to use your own AWS free tier account. And it pretty much works in three basic steps. First of all, you create an analyzer, then Access Analyzer is going to review any active findings, and then we'll be able to go ahead and take action on those findings. So let's quickly walk through this. First of all, I'm going to create an analyzer. My region is US East Northern Virginia. It's prepopulated a name for my analyzer. My zone of trust is going be the current account I'm in. Then go ahead, scroll down to the bottom, and Create analyzer. Now I'm going to give the analyzer some time to find any resources. And you can see straightaway it's already found a couple of things. So it looks like we've got a problem with this S3 bucket. Let's go ahead and click on the finding ID. And we can see straightaway is that we've got a public bucket. So it says this finding is for a resource that allows public access, and it's allowing read access for everybody. So I can go ahead and click on the resource, and I can see my bucket is publicly accessible. So let's select Permissions, come down to the block public access settings and edit those, and we're going to block all public access. And then

if we scroll down further, we can save changes. And, of course, as this is only a lab, this is okay to do. But if this was in production, then you would want to understand what is actually using this bucket before completely removing all public access just in case you break something. Confirm our settings, scroll down again, and we'll see that we've also got to a bucket policy which is enabling public access So let's go ahead and delete our bucket policy. We'll acknowledge the settings. I'll just close these messages down. And our bucket should no longer be public. So now, I'm just going to head back to IAM Access Analyzer, so search for IAM, find Access Analyzer on the left, and we should find that that bucket has been removed from the list. And if we come to the resolved findings, there it is. It's been resolved. So that's pretty much how Access Analyzer works. It's pretty cool, eh? So now let's go back and talk about the last way that we can troubleshoot access issues with AWS. So the final service I wanted to talk to you about is the IAM Policy Simulator. And this allows you to create and simulate policies attached to IAM users and groups, as well as troubleshoot and test permissions boundaries. So this tool is a great way to simulate access provisioning without actually granting it, and it allows you to test policies that are attached to AWS resources, including conditional elements like IP addresses, dates, etc. So now, let's head over to the AWS console and check it out. So from the console, we'll search for IAM. Select that. And you'll find the Policy Simulator on the right-hand side under the Tools section. So just select that. And now within the Policy Simulator, we can simulate any policy that we like. We can select a user, group, or role that we'd like to test, and I'm going to test the cloud_user. First of all, make sure that AWS Organizations SEP is deselected because we don't want that to affect our results. And then I have the option to uncheck these policies if I would like to exclude them from the simulation. Next, I'll select the service I'd like to simulate. I'm going to go for API Gateway. I'm going to select All to select all actions and then Run Simulation. And we'll see that we are getting permission denied for all of these actions, and this is because I removed these policies from our simulation. But if I add this one back in the allow_all and run the simulation again, we should see the that we're getting permission allowed. And you can actually go in and even create your own policy that you want to test right here within the IAM Policy Simulator. So you can test the access of a brand new policy, and you can just add the policy code in here. And when you do that, you are only creating the policy for the purposes of the simulation. It's only going to exist in the simulator. It's not going to be applied to your AWS account. So that is how IAM Policy Simulator works. So now let's head back over and cover our exam tips. So for auditing and troubleshooting access issues, just remember these three services and features. So first of all, there's CloudTrail, which allows us to monitor API usage within AWS. And you can view the CloudTrail events within the AWS console, in S3, as well as using CloudWatch Logs. And it also enables us to check recent events in the CloudTrail console, and we can even query them using SQL if we have it all set up with Athena. Also, keep in mind the IAM Policy Simulator, and this allows us to simulate policies that are attached to our IAM users, groups, or roles. And we can also use this information to troubleshoot and test permission boundaries to allow us to understand what our users can and can't do. And then we've also got the IAM Access Analyzer, which allows us to check several services and their resources to help identify which of them are shared with an external entity. So that is it for this lesson. If you have any questions, please let me know. But if you're ready to move on, please join us for the next lesson. Thank you.

AWS Inspector vs. Trusted Advisor

Hello, Cloud Gurus, and welcome back to this lecture on AWS Inspector versus AWS Trusted Advisor. In this lecture, we'll discuss 2 services within AWS that's used interchangeably and can be

quite confusing as it relates to your SysOps exam. We'll help you better identify them and separate them so you won't get tricked on these particular questions. So we'll talk about AWS Inspector as well as the reporting that it offers. And then we'll also discuss AWS Trusted Advisor, as well as its security checks. And then we'll put them side by side and give you a side-by-side comparison of them as well as their functionalities. And finally, we'll conclude with some exam tips. So let's go ahead and get started. First starting with AWS Inspector, it's important to note that this service is a security service and AWS Inspector is an AWS service, which provides automated security assessments on our EC2 instances. So if you have a question as it relates to AWS Inspector, if it doesn't pertain to EC2 applications, that might be a hint that that's not the answer. However, AWS Inspector assesses your application for these 3 things. First starting with common vulnerabilities and exposures, network security best practices, as well as application security best practices. AWS Inspector works very simply. You simply start by installing the AWS Agent on your EC2 instances, you're then able to run an assessment on your target EC2 instances in which these assessments check your network as well as host for security vulnerabilities, and then it produces your findings and allows you to remediate those issues. Now let's talk about AWS Inspector reporting. One thing AWS Inspector reporting shows you is your common vulnerabilities and exposures, your network security best practices. So perhaps you have some VPCs that are open to the internet. It also talks about authentication best practices, perhaps you have a password that doesn't meet the password standards. Operating system, this would be something like making sure your operating system is correctly hardened, as well as application security best practices and PCI DSS 3.0 assessment. On the other hand, we have AWS Trusted Advisor, which is a management and governance service. And this is an AWS service which offers recommendations to optimize your AWS resources according to best practices. Unlike AWS Inspector, AWS Trusted Advisor runs assessment on your AWS account and its resources to give you recommendations on how they could be optimized. And the recommendations ranged from these areas; cost optimization, so this may be idle EC2 instances or low utilize resources that you can clean up to lower your account. Performance. Perhaps you have an over-utilized resource which may be costing you. Security, as well, maybe you have an open VPC, some subnets that are too loosely opened, AWS Trusted Advisor will give you recommendations on those. It also gives you information for fault tolerance, making sure your architectures are multi-AZ, if you've selected them to be so, as well as checking your service limits within your AWS accounts. AWS Trusted Advisors starting with cost optimization helps you save money by giving you recommendations to eliminate unused and idle resources. They also instruct on your performance. It checks your service usage over 80% of the service limit threshold and monitors over-utilized instances. For security, it improves the application security by checking your security permissions as well as enabling security features. From a fault tolerance perspective, it increases your application availability through leveraging backup, auto scaling, health checks, as well as multi-AZ capabilities. And lastly, it checks for those service limits, which checks for service usage greater than the 80% of the allotted service limit. So now let's talk about, a little bit more in detail, about what AWS Trusted Advisor security checks look for. They first look for public EBS snapshots as well as RDS public snapshots, S3 bucket permissions, IAM usage, multi-factor on root account, as well as security groups with unrestricted ports. Now let's compare these 2 services side by side to help you get a better visualization of how you can distinguish the services. Starting with AWS Trusted Advisor, this service provides recommendations across multiple services. On the other hand, AWS Inspector inspects security and network accessibility of your EC2 instances, and what both these services do is perform security and compliance audits as well as generate reports. For our exam tips for AWS Inspector versus Trusted Advisor, starting with AWS Trusted Advisor, remember

that it provides these types of optimization recommendations across multiple services. Starting with security, AWS Trusted Advisor gives us security recommendations on EBS public snapshots, RDS public snapshots, S3 bucket permissions, IAM usage, multi-factor authentication on our root account, as well as security groups with unrestricted ports. From a performance perspective, we can expect to see recommendations such as high utilization of EC2 instances, excessive rules in our security groups, Amazon EBS volume attachment configurations, over-utilized EBS magnetic volumes, CloudFront content delivery optimization. From a service limits perspective--and these are just a few that you can look for--but there's many more. You can check within the service to see all of the recommendations provided. However, for service limits, you can expect to see Auto Scaling groups and launch configuration recommendations, CloudFormation stacks, DynamoDB read and write capacity, EBS throughput storage, EC2 Reserved and On-Demand leases, RDS configuration checks, and VPC internet gateways. For fault tolerance, you can expect to see recommendations on EBS snapshots, EC2 Availabilities Zone balance, load balancer optimization, Amazon RDS, multi-availability zone, S3 bucket logging and versioning, as well as Route53 records set checks. And lastly, for our cost optimization recommendations, you can expect to see these low utilization of our EC2 instances, idle load balancers, idle RDS instances, saving plans, Amazon RDS reserved instance optimizations, as well as AWS Lambdas with high error rates and timeouts. For AWS Inspector, AWS Inspector inspects security and network accessibility of our EC2 instances, and keep in mind the network assessment, which the network configuration analysis checks for ports reachable from outside of the VPC. AWS Inspector also does host assessments. And the host assessments check for vulnerable software, host hardening, and security best practices. So that's all for this lecture. If you're ready to move forward, join me in the next lecture.

Introducing AWS Organizations

Hello, Cloud Gurus, and welcome back to this lecture on Introducing AWS Organizations. In this lecture we'll define what is the AWS service AWS Organizations. As well as get a better understanding of the functionality AWS Organizations provide. We'll talk through how Organizations assist in controlling access and permissions, as well as the policies that you can define within Organizations. We'll also discuss the service integrations within AWS Organizations, as well as the feature of consolidated billing. And as usual, we'll conclude with some exam tips. So let's go ahead and jump in. AWS Organizations is a service which allows us to manage multiple AWS accounts in one place. Through AWS Organizations, we can centrally manage policies across multiple AWS accounts, as well as control their access to select AWS Services. AWS Organizations also allows us to automate account creation and management, as well as consolidate billing across multiple AWS accounts. Now let's get a better understanding of AWS Organizations. So let's imagine within your workplace or within your organization there is a CEO. And then under that CEO, we have several different business units which may include marketing, or finance, or even information technology, as well as operations. And each of those unique business entities are separated by their function or their value that they're adding to the business. Well the same works within an AWS Organization, where when you set up an AWS account you have a root account, which would be the highest level account. And then under that particular root account, you can separate the account into organizational units. Which are separated by their function or the functionality that's provided within the AWS accounts. So under these organizational units, we have AWS accounts. And the ability to control which particular services each of these accounts have access to is what we know to be as a service control policy. And a service control policy can be

attached to either an organizational unit or a singular account. Now let's talk through controlling access and permissions. So through AWS Organizations, we can integrate with AWS Single Sign-On, which configuring SSO enables you to grant access to multiple AWS accounts using Active Directory and customizable permissions based on job roles. You can also attach--as we just discussed-- service control policies. Which, service control policies can be applied to users, accounts or organizational units to control access to AWS resources, services, and Regions within your organizations. And lastly, through AWS Organizations, we can share resources across accounts. Accounts within AWS Organizations can share resources such as AWS Transit gGteways, VPC subnets, route 53 Resolver rules, and more using AWS Resource Access Manager. AWS Organization allows you to set several different policies within the service, which are then applied to the organizational units and the accounts that are contained within them. Starting with service control policies, which we previously discussed can be applied to users', accounts or organizational units to control access to AWS resources services as well as Regions. AWS Organization also allows you to set artificial intelligence opt-out policies, which enable you to control whether AWS artificial intelligence services can store or use your content. You can also set backup policies within AWS Organizations. Which allows you to deploy organization-wide backup plans which ensures compliance across organizations' accounts. And lastly, you can initiate tagging policies which is very helpful in standardizing tagging across all your accounts. AWS Organization also provides several service integrations within organizations which help you perform actions through the services directly from AWS Organizations. Starting with security, AWS Organization integrates with GuardDuty for security reporting within your accounts, as well as AWS Resource Access Manager for sharing certain resources within your account. Amazon Macie, Amazon Artifact, as well as Amazon Firewall Manager for managing firewall and security rules, AWS Directory Service, Amazon Single Sign-On, as well as Amazon Security Hub. AWS Organizations also have service integrations from a management and governance perspective. So for those configurations that you would like to see across accounts, we have AWS Config as well as AWS Systems Manager, AWS Service Catalog, AWS License Manager, as well as CloudTrail, and CloudFormation Stacks. AWS Organizations also offers consolidated billing, which enables you to set up a single payment method for all AWS accounts in your organization through consolidated billing. As well as monitor their usage through AWS Compute Optimizer, where you can understand the resource utilization of your compute resources. As well as allowing you to configure budgets and cost alarms through AWS Cost Explorers. For our exam tips for AWS Organizations, just keep in mind that AWS Organizations is the service where we can manage multiple AWS accounts in one place. So this is a central managed service where we can manage policies across multiple AWS accounts. We can also control access to AWS services to those accounts within the organizations that we create. We can also automate account creation and management, as well as consolidate billing across multiple AWS accounts. That's all for this lesson. So if you're ready to move on, join me in the next lecture.

Service Control Policies

Hello, Cloud Gurus, and welcome back to this lecture on service control policies. In this lecture, we'll introduce the concept of service control policies as well as understand how service control policies allow us to grant and deny access to different services within our accounts. We'll also discuss how this is done through allow versus deny list, as well as talk about how service control policies allows us to inherit those permission boundaries within our accounts. And as usual, we'll wrap up with some exam tips. So let's go ahead and jump in. Service control policies are policies

created within your organization which help you control and manage an account access to services. Service control policies are created within AWS organizations and they pretty much allow you to determine which services are used within select accounts. And by default, service control policies use a block list strategy, meaning they assume that all services are available and you select the services which you'd like to block. However, service control policies can be created for allow list statements. Now let's understand how service control policies implement these permissions within our accounts. Let's say I have my route account, and for the sake of this example, I've decided to go with 3 organizational units to establish my multi-account strategy. Within this multi-account strategy, I have my production workloads, development workloads, as well as security. Now the service control policies that I'll establish in this example isn't primarily what we would do within an account strategy. However, it gives you the idea of what can be done through service control policies. So let's say within my production workload, I would like to make sure that I enable-- along with other services--VPN so that users who have access can VPN to check logging and monitor other services. However within my development workload, I would like to disable all machine learning services as though this can be costly and add to my overall development budget. And also within security, I would like to block all compute resources as these do not necessarily contribute to overall security reporting. Now this may or may not be something you would be interested in. However, this idea and how we would execute and implement this through our organizational unit as well as accounts would be through service control policies. To control the services we grant permission to within our service control policies, the policies can be used as an allow or deny list. And with the allow list, we determine which services we would like to allow access to within the select AWS accounts. And this means that actions or access to other services are prohibited or not allowed by default. And within our allow list, we specify which services and actions are allowed, all other services and access will be denied. On the other hand, we have our deny list and this deny list is where all actions are allowed by default and we specify which services and actions we would like to prohibit. Now let's take a look how an allow versus deny list looks within the JSON of a service control policy. So we'll start with an allow list. Within a allow list, we simply state which services we would like to allow and which actions within those services we would like to allow as well. In this scenario, we've allowed access to only ECS (which is Elastic Container Service) as well as CloudWatch on all resources. All other services and permissions within this particular account where this service control policy is established would be prohibited. Now let's take a look at our deny list. With the deny list, we allow access to all AWS services and implicitly state which service we would like to deny access to. Within our example here, we see that all access and all actions have been allowed within the first part of the JSON. However, access has been denied to DynamoDB. So keep in mind how these allow versus deny access list work. Now let's discuss service control policy inheritance because as you develop these to secure your account, you need to know how they work, otherwise you may have access issues using certain services. So let's say within a certain organizational unit, you establish a service control policy within your root account or parent account which allows the use of these 3 services--for example, the machine learning, CloudTrail, as well as EC2. Now, these services are allowed within not only this organizational unit but the organizational units and accounts underneath it. Now within another organizational unit, you create another service control policy to allow EC2 as well as Aurora and Elastic File service. Because of the service control policies you've created, you would only be able to use EC2, however, would not be able to use Aurora and Elastic File System because you haven't explicitly stated it within your service control policy. So you'll definitely need to understand how service control policy inheritance work so you won't have any service control policy issues within accessing your

services. Far service control policy exam tips, keep in mind the 2 type of lists in which we can grant service control policies to our organizational units and accounts. Remember allow lists where actions are prohibited for all services. And we specifically state which services and actions are allowed. And on the other hand, deny list, where all actions are allowed by default. However, we specify which services and actions are prohibited. And also keep in mind the purpose of service control policies which these are policies created within your organization which help you control and manage an account's access to services. And remember, by default, services control policies use a block list strategy. However, they can be created for allow list statements. That's all for this lesson on service control policies. If you're ready to move on, join me in the next lecture.

Securing Multiple Accounts with AWS Control Tower and Organizations

Hello, Cloud Gurus, and welcome back to this lecture on securing multiple accounts with AWS Control Tower and Organizations. In this lecture, we'll discuss how to secure a multi-account strategy using AWS Control Tower and Organizations. We'll do this through getting a better understanding of what is AWS Control Tower. We'll discuss what necessitates a multi-account framework, as well as some key Control Tower terminology, such as landing zones, account factory, and guardrails. As usual, we'll wrap up with some exam tips. So let's go ahead and get started. What is AWS Control Tower? So Amazon Control Tower is a service which allows us to create, configure, as well as manage multiple AWS accounts securely through automating ongoing policy management and guardrails, and AWS Control Tower uses AWS Organizations to create accounts, apply guardrails, and utilize consolidated billing. AWS Control Tower and Organizations help implement a multi-account framework and there are several reasons why we would like to use a multiple-account framework over just putting all of our resources and users within one account. One reason would be that there are many teams within our organization, and with many teams comes different responsibilities and resource needs, and to make sure they don't overstep one another, we implement a multi-account framework. Another reason would be the limit allocation which is granted for each account. In order to make sure that our resources and applications aren't constricted by service limits, we add multiple accounts to make sure each application can function freely without having its service limits maxed out. A multi-account framework is also useful for the billing of our applications in AWS. Through separating our accounts, we can easily separate items at a billing level so we can easily see transfer charges and billing across business units, functional teams and individual users. A multi-account framework is also useful for isolation. So when we'd like to isolate resources or environments such as production, QA, or development, we can easily separate users by their function and make sure that our most important resources have the level of security that they need. Through creating a multiple account framework, we can also enforce different security controls, which AWS Control Tower offers, and we can also ensure that we isolate the data through creating them in separate accounts. Within AWS Control Tower, there's some key terminology that it's very useful to know. First, starting with landing zones, which is a framework for creating a multi-account architecture within AWS to configure default accounts, account structure, and network and security layouts. So within landing zones, we can create and configure default configurations that we'd like to create for accounts in the future. There's also another term; account factory. An account factory is an account template, which helps you provision new accounts with pre-approved standardize account configurations. There's also guardrails, which are security constraints or rules, which provide governance within your AWS environment. Guardrails are applied to Organizational Units or OUs and the accounts within them. Now let's talk a little bit more

about landing zones. Within landing zones, you can do several things. You can first build out a multi-account environment using AWS Organizations. You can also configure identity management and federated access using AWS Single Sign-On, or SSO. You can also use consolidated logging from AWS CloudTrail and AWS Config, which can be configured and stored within S3. And finally, you can create cross-account security audits using AWS SSO and Identity and Access Management. Within your AWS Control Tower landing zone, there's a default structure that's made available to you. But keep in mind, you can always add more accounts to this structure. Within your master account, there exists an account baseline as well as the AWS service catalog and organizations, which help control how you would like to create organizations in the future. You can also integrate AWS Single Sign-On which helps federate user access or simplifies the ease of users logging in to each of the subsequent accounts that's created. So within this particular core organizational unit, there's 3 accounts that's created by default. First, you have the AWS Shared Service Account, which holds the default VPC or network baseline. You also have the Log Account, which by default comes with all the logging information for the accounts, which holds the aggregate AWS CloudTrail and Config logging. You also have the AWS Security Account, which houses all the security information, including cross-account roles, security notifications, and Amazon GuardDuty by default. Keep in mind, you can configure this particular landing zone structure to have more accounts created by default, if you'd like. Account factory, which serves as our account template functionality within Control Tower, allows us to do several things. First, provision and manage accounts. It also allows us to automate account provisioning with IAM roles. So it makes it easy for us to streamline our IAM roles within these new accounts that we create. We can also configure Amazon VPC settings. And lastly, we can provision the accounts using Service Catalog. So ensuring that we can deploy our resources within the accounts we create using Service Catalog. As it relates to securing our accounts, guardrails are essential as they provide preconfigured governance rules for security and compliance for our operations. And there's 2 types of guardrails you'll need to understand. There's first, preventative guardrails, which prevent policy violations through AWS CloudFormation and service control policies, and these guardrails are always compliant, so we can understand these to be, let's say, we would like to block use of machine learning or artificial intelligence services through service control policies-- these are preventative guardrails. There's another type of guardrail, which is detective guardrails and these detective guardrails detect policy violations and alerts in the dashboard through AWS Config rules, and these particular guardrails check within the accounts within our AWS Control Tower or Organizations, and they check and let us know if we are compliant or non-compliant, which then we can later see within our AWS Control Tower dashboard. For our exam tips for securing multiple accounts with AWS Control Tower and Organizations, keep in mind that AWS Control Tower is the service which allows us to create, configure and manage multiple AWS accounts securely through automating ongoing policy, management and guardrails. An AWS Control Tower uses AWS Organizations to create the accounts, apply guardrails and consolidate billing. Also keep in mind what AWS Landing Zones are. They are the framework through which we create a multi-account within architecture within AWS to configure default accounts, account structure, and network and security layout. We also use AWS Account Factory as our account template, which helps us configure new accounts with pre-approved standardize account configurations. Also keep in mind guardrails, which are the rules, which helps us to provide governance within our AWS environment, and guardrails are applied to Organizational Units and the accounts within them. Also keep in mind the 2 types of guardrails; preventative guardrails, which are always compliant as they are enforced through service control policies as well as through CloudFormation; as well as the detective guardrails, which AWS Control

Tower detects and displays within the AWS Control Tower dashboard, if we're compliant or noncompliant. And that wraps up our lesson on Securing Multiple Accounts with AWS Control Tower and Organizations. If you're ready to move on, join me in the next lecture.

Security Token Service (STS)

Hello, Cloud Gurus. And welcome back to this lecture on Security Token Service. In this lecture, we'll get a better understanding of Security Token Service by defining it. We'll also discuss some key terms that you should know as it relates to Security Token Service. We'll discuss how STS, or Security Token Service, can be accessed through the command line interface or the AWS APIs. We'll also discuss an STS use case to help you get familiar with how Security Token Service works. And per usual, we'll wrap with some exam tips. So let's go ahead and jump right in. AWS Security Token Service is a service that allows us to create temporary security credentials that grant trusted users access to our AWS resources and the credentials that it provides are credentials for short-term use that can be active for as little as a few minutes or up to several hours. These credentials, once expired, can no longer be used to access your AWS resources. STS can be used through the AWS CLI, command line interface, and the AWS APIs. So this service is available first through end points and the end points is where our credentials are returned with 3 components, a security token, an access key ID, and then a security access key. You can also access STS through the command line interface and the AWS CLI makes it easy to request STS credentials programmatically using the CLI. So this is great for a lot of your processes and you can automate the retrieval of STS credentials, so your applications can access different components of AWS, as well as the resources you've created. STS also is good for recording API requests. AWS STS supports AWS CloudTrail and can be configured to deliver log files to S3. STS grants users limited and temporary access to your AWS resources. Starting with your enterprise identity federation, AWS STS uses Security Assertion Markup Language, which is regularly known as SAML 2.0. It also grants temporary access based on user's credentials in your active directory. It's also able to be integrated with single sign-on or your custom federation broker. STS is also great for web identity federation. And this is for use cases like Facebook, Amazon, Google, or other open ID providers, which help us log into our applications. AWS STS is also great for cross-account access, which lets users from one account access another account. So this is the way that we can give our resources access to complete cross-account functions that we've enabled within our AWS account. And finally, through STS, we're able to configure roles for EC2 and other AWS services. STS runs applications on EC2 instances with access to other AWS services without embedding credentials. Now let's discuss some key terms that it's very useful to know as it relates to Security Token Service or STS. First, starting with federation. Federation allows us to join or combine users in one domain, such as IAM, with a list of users in another domain, such as active directory, Facebook, etc. So federation would be something like single sign-on to sign in an application with your Gmail address. There's also identities. When we think of identities, we've simply think of users of a service like Facebook, for example. Everyone who has a Facebook page has an identity within a larger service, like Facebook. There's also the identity broker, which is a service that allows you to take an identity from point A and join it or federate it to point B. And lastly, there is the identity store, which are services like active directory, Facebook, Google, etc. Now let's walk through an STS use case to help you get a better understanding of how STS works within the scope of an application. Let's say we have a group of employees and the employee enters their username and password into their application. In this case, it's the enterprise reporting application. The application then calls out to the identity broker. The

broker captures the username and password. The identity broker then uses the organization's LDAP directory to validate the employee's identity. The identity broker then calls the new get federation token function using IAM credentials. The call must include an IAM policy and a duration which can be from 1 to 36 hours, along with a policy that specifies the permissions to be granted to the temporary security credentials. The Security Token Service then confirms that the policy of the IAM user making the calls to get federation token, gives the permission to create new tokens and then returns 4 values to the application: an access key, a secret access key, a token, and a duration, which includes the token's lifetime. The identity broker then returns the temporary security credentials to the reporting application. The data storage application then uses the temporary security credentials, including the token, to make a request to Amazon S3. Amazon S3 then uses IAM to verify that the credentials allow the requested operation on the given S3 bucket and key. IAM provides S3 the go-ahead and perform the requested operation. So this is how STS would be implemented within an application. Now let's discuss our exam tips for Security Token Service. Remember that this is the way that we create temporary security credentials that grant trusted users access to our AWS resources and we'd use security token for the following use cases or uses. We'd use it for identity federation, we'd also use it for web identity federation, cross-account access. And lastly, we'd use it for roles for EC2 instances and other AWS services. So that's all for this lecture on AWS Security Token Service. If you're ready to move on, you can join me in the next video.

AWS Key Management Service (KMS)

Hello, Cloud Gurus, and welcome back to this lecture on AWS's Key Management Service. In this lecture we'll get a better understanding of AWS's Key Management Service. We'll also discuss what is a CMK, or a customer master key, as well as how to access KMS. We'll also discuss some Key Management Service CLI commands as well as some AWS services that Key Management Service integrates with. And as usual, we'll wrap up with some exam tips. So let's go ahead and get started. AWS's Key Management Service allows us to create, store, and manage encryption keys within AWS. It allows us to centrally manage all of our cryptographic keys used to protect your data in AWS. It also integrates with many AWS services making it easy to encrypt your data within existing services, as well as logs key usage within AWS CloudTrail logs for compliance and audits. And the really cool thing is it simplifies encrypting data by just checking a box within select services. Now, the key component of AWS's KMS is the CMK. The CMK is a customer master key, and the customer master key is a logical representation of a master key which holds key material to encrypt data. And CMKs can only encrypt up to 4 kilobytes of data. So what is the CMK used for? It's used to generate, encrypt, and decrypt the data key. And the data key is actually used to encrypt and decrypt the data. Within your customer master key, there are several bits of data that's important to help AWS encrypt and decrypt your data using the CMK. So we'll go over those now. One of which is the alias, and the alias allows your application to refer to the alias when using the CMK. So you'll see that as a customer master key would have an alias like S3 for when encrypting data within S3 or SQS, which simply makes it easier to locate and manage your keys. Within the CMK, you also have your creation date, which is the date and time that the CMK was created. You also can give your CMK a unique description, which describes your CMK. The CMK also has key state, which shows the status of a key, which a key can be enabled, disabled, pending deletion, or unavailable. The CMK also has key material, which the CMK can either be customer-provided, which in that case is a key that you create, or it can be AWS-provided. And we'll speak about that in just a moment. And most importantly, CMKs always stay inside KMS. CMKs can never be exported. For accessing

KMS, KMS can be used within the AWS Console or CLI. And there's 2 types of keys that reside in KMS. There's the AWS managed keys, which are automatically generated AWS KMS keys for services integrated with KMS, and there's also custom keys. And since KMS only stores CMKs, all custom keys must be first created with CloudHSM. Now let's talk through some popular KMS CLI commands that is important to know if you'll be using KMS through the AWS CLI. First starting with aws kms encrypt. This is the command which encrypts plain text into ciphertext using a customer master key. To decrypt your data, you would use aws kms decrypt, and this decrypt ciphertexts that was encrypted by a customer master key. For re-encrypting data, you would use the aws kms re-encrypt CLI command and this command decrypts ciphertext and re-encrypts using a CMK that you specify. For example, when you change or manually rotate the CMK, which is really good for security purposes. And lastly, aws kms enable-key-rotation, which enables automatic key rotation every 365 days. AWS Key Management Service integrates with several AWS services which makes it very easy to encrypt your data at risk. Keep in mind these services which KMS encrypts your data as it may be seen on the Sysops exam. And it's also useful to know how to secure your data using the services. So for EBS, Elastic Block Store as well as S3, KMS is really great for encrypting data with those, as well as CloudTrail, DynamoDB, Elastic File Storage, RDS, as well as backups and AWS Lambda. It's a really good idea to understand how KMS allows you to use each of these services. For example, with your AWS backups, Elastic File Storage, as well as RDS, you can encrypt your snapshots using AWS KMS. So keep in mind which of these service functionalities are useful with KMS. For our Key Management Service exam tips, keep in mind, you'll need to understand an AWS managed CMK is an AWS-provided and AWS-managed CMK, which is used on your behalf with the AWS services integrated with KMS. Remember your data key, which are encryption keys that can be used to encrypt data, including large amounts of data. You can use a CMK to generate, encrypt, and decrypt your data keys. The customer managed key are the keys that you create, own, and manage within your CMK. And most importantly, you use KMS to encrypt your data at rest. KMS is an essential service for encrypting data across multiple AWS services. So that's all for this lecture on AWS's Key Management Service. If you're ready to move on, join me in the next lecture.

AWS Certificate Manager

Hello, Cloud Gurus. And welcome back to this lecture on AWS Certificate Manager. In this lecture, we'll understand the importance of AWS Certificate Manager and why we should use SSL or TLS certificates. We'll also discuss what happens when you enable and how HTTPS is enabled. We'll talk about AWS's Certificate Manager, as well as its service integrations. And lastly, we'll wrap up with some exam tips. Let's go ahead and get started. So why should we use SSL or TLS certificates? We use the certificates to easily identify secured websites and applications. We've generally liked our users to know that our websites are secure, and this is the way to do it through the use of the certificates, as it encrypts our information in transit. We also use the certificates because it secures our network connections. And as you can see through the images on the screen, this is how a secured HTTPS website looks that is correctly using an SSL or TLS certificate. Now, when we enable HTTPS traffic, we do so by using TLS encryption. And what happens is when a user comes to your website, the use of TLS encryption allows us to be able to encrypt their data while it is being transmitted. And the use of Certificate Manager allows us to be able to encrypt that data, not only in transit, but attach those certificates to our EC2 instances or our AWS resources. With AWS Certificate Manager, this is the service which is used to manage our certificates. Our certificates are

in charge of encrypting traffic that's going across several different platforms, not just in our AWS account. These are our Amazon EC2 instances. These can be our mobile devices as well as our on-premise services. And the overall picture here is to protect our organization's resources. And keep in mind, AWS Certificate Manager is a key method for encrypting our data in transit. So, let's talk a little bit more about AWS Certificate Manager. AWS Certificate Manager is the preferred tool to provision, manage, and deploy your server certificates. So the important role of this particular service is first, to provision certificates. And through the provision of our certificates through Certificate Manager, this service contributes to secure web presence by handling the creation, storing, and renewing of SSL and TLS certificate. You also have, within AWS Certificate Manager, the Private Certificate Authority, and this portion of the service allows you to create your own certificate authority, hierarchy, and issue certificates for authentication. Since AWS Certificate Manager is key for encrypting data in transit, AWS Certificate Manager is integrated with the following services, AWS Elastic Load Balancer, Amazon CloudFront, Amazon API Gateway, Amazon Nitro Enclaves, as well as AWS CloudFormation. For our exam tips for AWS Certificate Manager, keep in mind these things for your SysOps exam, as well as your use within AWS; the AWS recommended way to store and manage certificates is to use AWS Certificate Manager. And when you're enabling HTTPS on your website, you need to associate an SSL or TLS certificate. Just keep in mind the little lockbox that we saw or can see on our websites. That's signifying that a SSL or TLS certificate has been attached to that website and the communication with it is secure. And lastly, keep in mind the important reason why we use AWS Certificate Manager is because it is a key method for encrypting our data in transit. So that's all for this lecture. If you're ready to move on, join me in the next video.

AWS Web Application Firewall (WAF)

Hello Cloud Gurus and welcome back to this lecture on AWS WAF, or Web Application Firewall. In this lecture, we'll briefly introduce AWS Web Application Firewall. We'll answer the question what is AWS WAF? We'll talk about how WAF works, as well as the services Web Application Firewall integrates with. And we'll conclude with some exam tips. So let's go ahead and get started. What is Web Application Firewall? So let's talk about WAF. Web Application Firewall is a firewall that allows you to monitor your HTTP and HTTPS requests. So, this particular service allows you to monitor and provide mitigation against attacks that comes through your HTTP and HTTPS traffic. And so what WAF does, first, is it filters your web traffic, which provides you an ability to configure conditions, such as which IP addresses are allowed to make a request or which query string parameters are needed. WAF also offers full API, which AWS WAF allows you to create and maintain rules via API, which can be deployed through and provisioned with CloudFormation. And finally, AWS WAF provides real-time visibility, which provides metrics and details from application requests about IP addresses, URIs, geolocation, and referrers. Now let's discuss how WAF works. So WAF is managed through AWS Firewall Manager, which allows you to manage multiple AWS WAF deployments. And through AWS Firewall Manager, WAF can be configured with Amazon CloudFront, Application Load Balancers, Amazon CloudWatch, as well as Amazon API Gateway. And within WAF, you're able to create a policy or simply build rules through code or the visual rule builder. You're also able to block and filter, which protects against vulnerabilities and exploits by giving you the ability to define patterns, as well as IP addresses. And lastly, you're able to monitor using Amazon CloudWatch metrics along with log data. So, which services does WAF integrate with? WAF integrates with the following services or features, Amazon Application Load Balancers,

as well as CloudFront and API Gateway. AWS WAF does not integrate with Classic Load Balancers or Network Load Balancers. So for your exam tips, just keep in mind that WAF is an application firewall, which helps you monitor your HTTPS and HTTP traffic requests. It allows you to filter web traffic through its fully automated API and offers real-time visibility through CloudWatch metrics. Also keep in mind the services that WAF integrates with, which would be Application Load Balancers, CloudFront, and API Gateway, as well as what WAF does not integrate with, which would be the Classic Load Balancers, as well as the Network Load Balancers. So that's the brief introduction of WAF. If you're ready to move forward, join me in the next lecture.

Demo: Configuring AWS Web Application Firewall (WAF)

Hello Cloud Gurus and welcome to this lesson where we'll be configuring an AWS Web Application Firewall. And we'll begin by creating a sample API using API Gateway. Next, we're going to use AWS WAF to create a web access control list to block incoming traffic from our location. And then finally, we'll test to verify that the ACL is working correctly. So if you're ready to get started, please join me in the AWS console. So here I am in the console, and the first thing I'm going to do is search for API Gateway. So select API Gateway. And we're going to build a new REST API. So scroll down on this page until you find REST API and then select Build. Now they're prepopulated with some sample code, and it's defined using Swagger. So select OK. Our API is going be REST API. We're using the example code. And here is the example code, and it's basically just defining a very simple web application using this Swagger template. So then just scroll down to the bottom and select Import. So now that our code has been imported, we can go ahead and deploy our API. So from the Actions menu, select Deploy API. Deployment stage is going be New Stage, and we'll call our new stage Testing and hit Deploy. Now the API is ready. Here's our invoke QRL. So just click on that, and there is our new API. However, let's say that we want to use AWS WAF to block traffic from a specific location. And I'm going to block individuals from reaching this page if they reside in the UK, and that's where I currently am. So let's go ahead and create a WAF access control list to do just that. So back in the AWS console, search for WAF. There it is. And I'm going to open it in a new tab. And the first thing we're going to do is create a web ACL, so select that. And I'm going to call it Block-API-Calls-From-UK. and you'll need to use the name of the country that you are residing in. I'll use that as my description as well. Scrolling down, the resource type is going be Regional resources because that includes the API Gateway. My region is going be US East. Down here, we need to associate the AWS resources. So select Add AWS resources. The resource type is API Gateway, and it's automatically found our API down here. So select that and Add. Then hit Next. This is where we're going to add our rule. So select Add rules and add my own rules and rule group. Make sure the rule builder is selected. We'll give our rule a name. The type is going to be a regular rule, and the rule is going to apply if a request matches the statement. And the statement is going to be Originates from a country in the UK. And then down here, this is where we select our country code. So just select the country that you are currently in. And we're going to use the source IP address of the request to determine the country of origin of the request. Then scrolling down and the action is going be to block the request. So now, select Add rule, scroll down, and hit Next. We don't need to set the real priority, so hit Next. By default, it's configuring a CloudWatch metric for our rule, so hit Next. We can review all of our settings and Create web ACL. And it might just take a few moments to complete depending how quickly AWS is running this morning. And there we go. It successfully created our web ACL. So now we're ready to test if it's all working. So come back to your API Gateway to your invoke URL. Select your URL, but don't be surprised if it doesn't work

immediately. Of course, AWS is a weird and wonderful thing. And sometimes, changes just take a few minutes to take effect. So just be patient. And in a few minutes, it should take effect because the rule that we just created needs to propagate worldwide. And I actually noticed in the FAQs for AWS WAF that they acknowledge that it takes around a minute to propagate. And in my experience, it actually takes about 5 to 10 minutes. So this is a good time to step away from the screen, have a cup of tea, come back in 5 minutes. And if it's still not propagated, then give it another 5 minutes, and we should be good to go. And after another couple of minutes, I'm going to test my URL again, and there we go. We cannot access our API anymore. We're getting this message, Forbidden. And if we head back to AWS WAF, select our ACL, come to the overview and then scroll down to find sample requests, we should be able to see that we've got some sample requests that have been blocked. So AWS WAF has blocked these because the source IP is coming from the UK. So for my exam tips, just remember that AWS WAF can be used to protect your AWS resources, and it can actually be used to protect CloudFront, API Gateway, Application Load Balancer, and AWS AppSync resources. And it helps you with threat mitigation by allowing you to create rules to allow or block traffic based on lots of different criteria. For instance, it can block things like SQL injection attacks and cross-site scripting attacks. It can block your traffic based on the IP address of the request, the country of the request like we did just now, the size of the request, and based on pattern matching or string matching on part of the request. So that is it for this lesson. If you have any questions, please let me know. Otherwise, I will see you in the Next lesson. Thank you.

Differentiating Dedicated Instances vs. Dedicated Hosts

Hello, Cloud Gurus, and welcome back to this lecture on differentiating Dedicated Instances versus Dedicated Hosts. In this lecture, we'll discuss how you can introduce additional security to your architectures through the use of Dedicated Instances and Dedicated Hosts. We'll do this through defining Dedicated Instances and Dedicated Hosts, and then we'll compare them side by side. As usual, we'll wrap up with some exam tips. Let's go ahead and get started. Let's start with Dedicated Instances. Dedicated Instances are reserved physical hardware for EC2 instances, and as we can see on the server images to the left, this is when we only have a portion of the server on which our EC2 instances can run, and this provides us 2 important security features. It first allows us to be able to run in a VPC on hardware that's dedicated to a single customer, meaning other customers cannot run on this particular hardware. It also allows us the ability to be physically isolated on AWS hardware from other instances that belong to other AWS accounts. Dedicated Hosts, on the other hand, are reserved servers for EC2 instances. So, unlike Dedicated Instances, Dedicated Hosts allows us to reserve the entire server, allowing us to have complete control of the physical hardware on which are EC2 instances run. This gives us several benefits from a security perspective. It gives us visibility and control over how instances are placed on a physical server, it also allows us to deploy our instances to the same physical hardware over time, it enables us to use our existing server-bound software licensing and also allows us to address our corporate compliance and regulatory requirements. Now let's discuss Dedicated Instances and Dedicated Hosts through a side-by-side comparison. Dedicated Instances allows us to be billed on a per-instance basis. It also allows us to share hardware with instances from the same account. And, lastly, it allows us automatic instance placement. Dedicated Hosts, on the other hand, allows us to be billed per host, and it allows us to meet regulatory requirements because of the separation it provides at a server level. It also has licensing conditions, which allows us to bring our own license to the entire server, and it gives us visibility of sockets, cores, and host ID, and bring your own license. And, overall, Dedicated

Instances and Dedicated Host provides us dedicated hardware for launching our EC2 instances. It's important to understand their differences so you can understand which can provide the security that you need within your application. So for our exam tips for Dedicated Instances versus Dedicated Host, keep in mind that Dedicated Instances are reserved physical hardware for our EC2 instances and, remember, that these just gives us certain space on the physical server, but not the entire server. Dedicated Hosts, on the other hand, allows us to have the entire server for our use, which is a more secure method. Also keep in mind the side-by-side comparison of Dedicated Instances and Dedicated Host, as they both meet different security needs. That's all for this lesson, if you're ready to move forward join me in the next video.

Using AWS Systems Manager Parameter Store

Hello, Cloud Gurus, and welcome back to this lecture on using AWS Systems Manager Parameter Store. In this lecture, we'll help you get to understand what is AWS's System Manager, as well as its feature, Parameter Store. We'll talk through what Parameter Store provides, how it works, as well as compare it side-by-side to AWS Systems Manager. As usual, we'll wrap up with some exam tips. So let's go ahead and get started. AWS Systems Manager allows us to be able to streamline and manage changes across multiple AWS resources. Without this, this process would be very daunting. So AWS Systems Manager allows us to manage and control our maintenance windows, patch management, run commands, and it also includes the use of the Parameter Store. And Parameter Store allows us to first secure storage for our parameters and secrets. And it also allows us, if we would like, to use optional encryption using AWS's Key Management Service, or KMS. Within the Parameter Store, we can also control user and resource access using Identity and Access Management. And the really cool thing about AWS Systems Manager Parameter Store is that the parameters and secrets we create are referable through multiple AWS services. Now let's talk about how AWS Systems Manager Parameter Store works. So let's say you have an application like Lambda or ECS, your Elastic Container Service. Your application reaches out for a plain text parameter request to Systems Manager Parameter Store. It then checks for IAM permissions to make sure that that user or application has access to the parameters that you would like. It's then able to feed the parameter back to your application. And with an encrypted parameter request, the application also reaches out to Parameter Store, where Parameter Store then will check the IAM permissions, reaches out to KMS for decryption service, and then feeds the data back to your application as a decrypted text. Now, let's compare Parameter Store and AWS Secrets Manager. Parameter Store and AWS Secrets Manager are very similar, but offer some different features. They both allow you to store values up to 4, 096 characters. You can also encrypt values with KMS, as well as refer to each of the parameters and secrets in AWS Parameter Store and Secrets Manager from CloudFormation. One difference between these 2 services and features is that AWS Secrets Manager offers a built-in password generator, while AWS Systems Manager Parameter Store allows you to use parameter storage at no cost. However, AWS Secrets Manager is great for cross-account access and automated secret rotation. Now, let's review our exam tips for AWS Systems Manager Parameter Store. AWS Systems Manager Parameter Store securely store secrets and parameters. AWS Secrets Manager, on the other hand, works similar to Parameter Store, however, Secret Manager provides cross-account access, automatic password rotation, and password generation. That's all for this lecture on AWS Systems Manager Parameters Store. If you're ready to move on, you can join me in the next video.

Understanding AWS Service Quotas

Hello Cloud Gurus and welcome to this lesson on understanding AWS Service Quotas. In this lesson, we'll help you understand how you can view your service quotas for AWS services. We'll cover what are service quotas. We'll do a walkthrough using the AWS console. Next, we'll talk through Trusted Advisor, and we'll conclude with some exam tips. So let's get started. Now, AWS Service Quotas are formerly known as Service Limits, and they allow you to do two key things. So first of all, we can centrally manage our quotas across multiple AWS services, and we can also request a quota increase directly from within the AWS console. Now the AWS Service Quotas console has got three main functionalities. First of all, from the dashboard, we can view service quotas of our most recently used AWS services, as well as pending and recently resolved quota requests. We can also see a drill-down of the AWS services. So we can view a detailed list of all the service quotas with any AWS service. And then finally, we can view our quota request history. So we can view any quota increase requests that we've made in the past 90 days. But just be aware that the ability to configure service quotas is not going to be available in the cloud sandbox. And if you'd like to follow along with me in the demo, then you will need to use your own personal AWS free tier account. So now, let's head over to the AWS console, and I'll show you the service quota feature first hand. So here I am in the AWS console. And remember, you will need to do this in your own personal AWS account. In the search box, type service quotas. And from the dashboard, we can see the quotas across several AWS services that we've used recently. Down here, we can see any pending service quota requests, as well as any recently resolved service quota requests. On the right-hand pane, select AWS services. And it's going to give us a list of all the different AWS services, and we can see the quotas for each of these services. In the search box, let's type VPC. Select VPC. And down here, we can see the quotas that I have within this account for the VPC service. I can see that my active VPC peering connections per VPC has an applied quota value of 50 and an AWS default quota value of 50. And also, it's adjustable. And if I scroll down, I can see the same information relating to all the different VPC services. And if I want to request a quota increase, I can do this here. Select the quota name, and you can request a quota increase using this button. Down here, we can change the quota value. So let's change it to 60. And then, we can use this button to make the request, but I'm not going to do that here in this account. So I'm going to just close that down. And then finally, we can check our quota request history. So select that in the left-hand pane, and this is where we can view any open quota increase requests or requests that were closed in the last 90 days. And I haven't currently made any within this account. And you can also manage your quota requests at an organizational level as well if your account is part of an organization. And this one isn't, so I'm not going to be able to do that. So now, let's head back over to the lesson and get a view of Trusted Advisor before we wrap up with our exam tips. Now another way to keep track of the status of your service limits or service quotas is within Trusted Advisor. So Trusted Advisor offers a service limits check, and that displays your usage and limits for some aspects of some AWS services. So it's just another way to keep track of the status of your service limits. Let's take a look at it in the AWS console. Back in the console, I'll search for trusted advisor. And from the dashboard, select Service limits on the left. And there we go. It's giving us a high-level overview, if there's any action required or recommended, and it's telling me there's no problems with my limits detected. And it's basically going check for usage that is more than 80% of the limit. And so that will alert you if you need to take any action and request a limit increase. So for my exam tips, just remember that AWS Service Quotas is formerly known as Service Limits, and those terms may be interchangeable within the exam. From the Service Quotas console, You

can essentially manage your quotas across multiple AWS services. And you can also request a quota increase directly from the AWS console. And then finally, you can also check your service limits using AWS Trusted Advisor. So that is it for this lesson. If you have any questions, please let me know. Otherwise, I will see you in the next lesson. Thank you

Reviewing the AWS Shared Responsibility Model

Hello Cloud Gurus, and welcome back to this lecture on reviewing the AWS shared responsibility model. In this lecture, we'll be introducing and revealing the AWS shared responsibility model, which is a model which helps you understand your responsibility within AWS, as well as AWS's responsibility from a security and resource support perspective. We'll talk through the AWS responsibilities as well as your responsibilities as a customer and then we'll also outline some exceptions to that model. We'll discuss the exceptions from an infrastructure services, container, and abstracted services perspective. And as usual, we'll wrap up with some exam tips. So let's go ahead and get started. The AWS shared responsibility model can be simply thought of as a separation of responsibilities within AWS. To help you remember this model, compare this model to a leasing agreement when renting a house or an apartment. The landlord has their responsibilities, which would include the foundational structure, plumbing, fire alarms, etc. While your responsibility would be the contents of the home, as well as your personal belongings. And that's exactly how we should view the AWS shared responsibility model. The shared responsibility model clearly outlines what's our responsibility as a customer, which is our responsibility for security inside of the cloud. This being so, we're responsible for the safety and security of our customer data. We're also responsible for the platform applications and Identity and Access Management for our applications, making sure that only the users who we would want can access our platform applications and applications to ensure optimal security. We're also responsible for our operating systems, network, as well as firewall configuration. Several of these are probably familiar to you as from a network and firewall configuration, you're probably thinking about VPCs and web application firewalls. We're also responsible for client-side data encryption and data integrity authentication. This would be the use of Key Management Service for example, which we would use as an encryption method for our data. We're also responsible for server-side encryption, encrypting our file systems like Elastic File System or EFS, as well as networking traffic protection, encryption, integrity, and identity. This would be something like using Certificate Manager, making sure that our data is encrypted within transit. On the other end, AWS has a responsibility for the security of the cloud. This would include the AWS software that's used for our compute, storage, database as well as networking resources. AWS is also responsible for the hardware and AWS global infrastructure. They're responsible for staffing and securing their data centers as well as the route services that they offer to us from an AWS perspective. This would also include different Regions, Availability Zones, and edge locations. And for more information, feel free to check out the entire shared responsibility model that's linked below as well as within your resource sections. AWS also has responsibility for the maintenance of their global infrastructure. This would range from staffing for the security of their data center, cooling for their servers within their data centers as well. Also hardware, software, networking and facilities, as well as managed services. So any services within AWS that are managed services also fall into the AWS scope of responsibilities. Some of our customer responsibilities include infrastructure as a service. This would include any applications which uses AWS's networking features, for example VPCs, compute resources such as EC2 instance, ECS or EKS, or data storage space, as well as ensuring each of those services has their correct level of

security to protect your customer data. You'd also be responsible for security patches and updates. This would include using AWS Systems Manager or the hardening of your EC2 instances. And lastly, your responsibility would include the configuration of AWS provided firewalls, which would be simply your security groups and making sure you're protecting your application from unwanted web traffic. There are some models to this exception. The AWS shared responsibility model changes for different service types. This would include infrastructure, container, as well as abstracted. With infrastructure service, this would include Amazon EC2, Amazon Elastic Block Store, Amazon Auto Scaling, as well as Amazon VPC. With these services, you control the operating system and you configure and operate any identity management that provides access to the user layer of the virtualization stack. For example, with your EC2 responsibilities, you're responsible for the Amazon Machine Image, the operating system, the applications. You're also responsible for the data in transit, the data at rest, the data stores, as well as the credentials policies and configurations. For container services like the following: Amazon Elastic Container Service, Elastic Kubernetes Service, Amazon Elastic MapReduce, Amazon Lambda, Amazon RDS and Elastic Beanstalk, you're also responsible for setting up and managing network controls such as firewall rules, and for managing platform-level identity and access management separately from IAM. Also keep in mind the following abstracted services: Amazon S3, Amazon Glacier, Amazon DynamoDB, Simple Queue Services, and Simple Email Services. These services abstract platform or the management layer on which you can build and operate cloud applications. You access the end points of these abstracted services using AWS APIs and AWS manages the underlying service components or the operating systems on which they reside. For AWS shared responsibility model exam tips keep in mind you need to have a strong understanding of the shared responsibility model. Keep in mind your responsibilities are EC2 patching, antivirus, as well as security groups. And AWS responsibility models would be RDS operating system updates, RDS database updates as well as code updates with Elastic Beanstalk. So make sure you understand which services AWS share responsibility for and what's your responsibility within those services. That's all for this lecture on the AWS shared responsibility model. If you're ready to move forward, join me in the next lecture.

Protecting Logs within CloudTrail

Hello, Cloud Gurus, and welcome back to this lecture on protecting logs within CloudTrail. In this lecture, we'll discuss how to protect your logs within CloudTrail. First, through understanding and going through a recap of CloudTrail. We'll also talk about why it's important to protect your CloudTrail logs. We'll talk through protecting as well as encrypting your CloudTrail logs. We'll also discuss what is integrity validation. We'll talk about some best practices for securing your CloudTrail logs within S3. And as usual, we'll wrap up with some exam tips. Let's go ahead and get started. Let's get started by recapping CloudTrail. So what is CloudTrail? CloudTrail records user activity within AWS, and it records events related to the creation, modification, or deletion of resources. And this could be anything from identity and access management users, S3 buckets, or EC2 instances. And by default, you can view the last 90 days of your account activity within CloudTrail. Also, you can create a trail which will store the logs indefinitely in S3. Now let's discuss why you would want to protect your CloudTrail logs. CloudTrail logs record the who, when, what, and where, as well as the source IP and request parameters and responses related to your API calls within AWS. So for example, if someone is up to no good within your AWS account, they may try to delete or modify the logs to conceal their activities. Also, it is important to protect your CloudTrail logs from being read, deleted, or modified by unauthorized users. And by protecting

your CloudTrail logs, there's a couple of things that are in place natively, and then there's another step you can do manually to help the protection of your CloudTrail logs. Natively within AWS, server-side encryption is enabled for your CloudTrail logs, as well as log file integrity validation, and a good best practice for securing your CloudTrail logs is to secure the S3 bucket used to store your logs. For encrypting logs within CloudTrail, when you create a trail using the AWS console, server-side encrypted KMS encryption is set to enabled by default, meaning you control who can use the key for encrypting and decrypting. You can also control which users have access, where only users with permissions to use the key will be allowed to decrypt the files. So what is integrity validation? You can also secure CloudTrail logs by enabling integrity validation, and integrity validation allows you to detect if a log file was changed or deleted. So when you enable integrity validation, CloudTrail creates a hash for every log file that it delivers. Every hour, CloudTrail also creates a file which it references each log file created in the last hour and contains a hash for each file. This file is called the digest file. Digest files contains all of the logs created in the past hour and each hour thereafter. Digest files are digitally signed using a private key owned by AWS. According to AWS, this makes it computationally infeasible to modify, delete, or forge CloudTrail log files without detection. Here are some best practices for securing CloudTrail logs stored in S3. One would be to use a dedicated S3 bucket for CloudTrail logs. This way you can control the access to your S3 bucket through the use of a bucket policy. CloudTrail will need permissions to write to this bucket. Also, restricting users and giving them read access. Restrict read access permissions to only those who need it, for example, your security team or your SysOps team. For our exam tips for protecting logs within CloudTrail, keep in mind when you create a trail using the console, encryption and integrity validation are set to enabled. Also AWS encrypts our logs within CloudTrail by using server-side encryption KMS. This allows us to control which users have permission to use the key to decrypt files. Also keep in mind integrity validation, which integrity validation should make it impossible to change or delete CloudTrail logs without detections, and last but not least, remember it's always a best practice to have a dedicated S3 bucket. Use a dedicated S3 bucket and secure it with a strong bucket policy. That's all for this lecture on protecting your logs with CloudTrail. If you're ready to move on, join me in the next lecture.

Demo: Introducing AWS Security Hub

Hello Cloud Gurus and welcome to this demo on AWS Security Hub. And in this demo, we're going be introducing Security Hub, which is a service that allows you to view and manage security alerts, as well as automate security checks in your AWS account. And it provides a single centralized view of the security of your AWS account. It allows you to monitor compliance with common security standards and best practices. And it utilizes AWS Config to perform many of its security checks. But the best way to understand Security Hub is to actually use it, and that's what we're going to be doing next. And we'll begin by configuring AWS Config, and we'll use a CloudFormation template to configure Config to record the configuration of resources in our AWS account. Next, we'll review the security standards that Security Hub uses to measure security compliance. And then finally, we'll enable Security Hub, and we'll review the integrations with AWS and third-party services, and we'll review the Security Hub findings as well. So if you're ready to get started, please join me in the AWS console. So from the console, first of all, I'm going to head to CloudFormation and select that. And we're going to use a CloudFormation template that's been provided for you in the Resources section, and here it is. It's called EnableAWSConfig_CF_Template.yml. And you will need to download this from GitHub to your local machine and provide it to CloudFormation. And don't

worry about the contents of the template. It's just a very simple template, which is going to quickly enable Config so that we can get on with using Security Hub. So from the CloudFormation console, select Create stack, With new resources. The template is ready. And select Upload a template file. I'm going to upload it from my local machine. There it is. Select Open. Hit Next. We'll enter a stack name. Select Next. We don't need to change anything else on this screen, so just hit Next. We can review all of our options. Acknowledge that CloudFormation might create IAM resources and Submit. And now, I'll just give it a moment to complete. And feel free to pause the video while your resources create because it just takes about 2 minutes to complete. A few minutes later, my CloudFormation template has successfully created all my resources. And if you select Resources, you'll be able to see everything that's been created. So we've got our ConfigBucket, BucketPolicy. It created a ConfigRecorderRole. It's created an IAM role and an SNS topic for us as well. So now, if we head over to Config, we can see that Config has been enabled, and it's started doing checks on our account already. But just keep in mind it can take a moment for Config to update your compliance status and check your resource inventory as well. So don't be surprised if you're not seeing as much on the screen at the moment. So now, let's head over to AWS Security Hub to check what our findings are. And open up Security Hub. So now, select Go to Security Hub, and it's reminding us that before we can enable Security Hub standards and controls, you must first enable resource recording in AWS Config, which we've already done. So now, we can scroll down here and select which security standards we would like to apply. And I'm just going to select all of them. Then down here, under AWS Integrations, it's telling us that by enabling Security Hub, we're granting permission to import findings from AWS services that we've already enabled. And we'll be able to take a look at those integrations once we've enabled it. So now, select Enable Security Hub. And once you've enabled it, Security Hub is going to start running its assessments on my account resources, and it's going to let me know shortly what I need to know from a security perspective and what I should change or fix. But just do be aware that after you enable Security Hub, it can take up to 2 hours to start seeing the results from the security checks for the newly enabled standards. But hopefully, we should start to see some results after just a few minutes. Now if we select Security standards on the left, these are the standards that we've enabled for Security Hub. If we select Insights, this is where you can get a high-level overview of the resources with the most findings. And we can find out which S3 buckets have been configured with public write or read permission. But these are basically just the high-level insights that will give you an idea of the security posture on your account. So now, if we select Findings, these are the individual findings that Security Hub has detected. And a finding is simply a security issue or a failed security check. We've got some that are low severity, some medium, and we've probably also got some critical ones as well. But if I order by severity, and I need to click this arrow twice, then we'll get the critical ones at the top. And there's all our critical findings. And if you scroll to the right, under Title, you can see the title of each one. And then if you select one, you can drill down and get mode details. So over here, it will give you an explanation of why the check has failed, the related requirements. So this is to do with the PCI DSS check. And then, if you scroll down, you can find out which resource it's in relation to. So this is in relation to my account as a whole because this is to do with hardware MFA for the root user. And then down at the bottom under Remediation, it will give you directions on how to fix the issue. And based on the results of all these individual assessments, eventually, it's going to be able to give us a score for how compliant we are with the security standards that we enabled. So if we head back to Security standards under PCI DSS, so the Payment Card Industry Data Security Standard, down here, eventually, and it's probably going to take a couple of hours, but we will get a percentage security score, and you'll also be able to review the results of all the assessments that

Security Hub has done in relation to the PCI DSS standard. So now, if we come back over here on the left and select Integrations, you will see the that Security Hub also accepts findings from other AWS services and also third-party integrations as well. So services like AWS Config, which we've already enabled, Amazon Detective, which is used to analyze, investigate, and identify the root cause of security findings or suspicious activities, AWS Firewall Manager, which allows you to centrally manage and configure AWS WAF rules, GuardDuty, which is a threat detection service, and AWS Health, which gives you visibility on resource performance and availability of your AWS services, IAM Access Analyzer, Inspector, which is another security assessment service, and Macie, which allows you to discover and classify and protect sensitive data in AWS. And then, if you scroll down, you will also see a load of third-party integrations as well. So it's also going to accept assessment information from these tools as well. So I hope you'll agree that Security Hub is a great one-stop-shop for all of your security monitoring, aggregating information from all these other sources, as well as performing its own security assessments. And it's a great way to quickly optimize security for your AWS account. And then one last thing that I wanted to show you, and this is within my own AWS account, is my own summary of the security standards in my own account. And you'll see here that it's giving me a security score of 77% for the AWS Foundational Security Best Practices standard. So this is the kind of information that you can expect to see in a couple of hours after Security Hub has been enabled. So for my exam tips, keep in mind that Security Hub provides a single, centralized view of your AWS account security. It monitors compliance with security standards and best practices, and supported standards include the AWS Security Best Practices, CIS Foundations Benchmark, and PCI DSS, and I'm sure they'll add more soon. It's also integrated with many AWS services like Config, GuardDuty, Inspector, and Patch Manager, as well as third-party companies like Alert Logic, Check Point, and CyberArk for threat detection. And Security Hub provides a great dashboard for showing our security score for each of the standards that we've enabled, and here's an example of what it's going to look like a couple of hours after enabling Security Hub when it has completed all of the security assessments. So that is all for this lesson. Any questions, please let me know. But if you're ready to move on, please join us in the next lesson. Thank you.

Exploring Amazon GuardDuty

Hello, Cloud Gurus. And welcome back to this lecture on exploring Amazon GuardDuty. In this lecture, we'll help you get a better understanding of Amazon GuardDuty by answering the question, "What is GuardDuty? " We'll also talk about Amazon GuardDuty's features as well as how to get set up using GuardDuty. And as usual, we'll wrap up with some exam tips. Let's go ahead and get started. What is GuardDuty? So GuardDuty is a threat detection service which uses machine learning to detect malicious behavior within your AWS account. And GuardDuty does that through tracking several different resources and features within AWS. First, it looks at your API calls. Unusual API calls and API calls from a known malicious IP address. AWS GuardDuty also checks for the disabling of CloudTrail. Any attempts to disable CloudTrail logging are monitored through GuardDuty. GuardDuty also checks for compromised S3 buckets. For example, unauthorized access from a known malicious IP address. GuardDuty also checks for compromised EC2 instances, for example, evidence of Bitcoin mining, overuse of AWS resources, or communication with a known malicious IP address. GuardDuty also checks for reconnaissance activities. For example, port scanning and failed login attempts. And lastly, it checks for compromised accounts, evidence of unexpected behavior, for example, unusual infrastructure launches. Now let's talk about some of the

features that are available within GuardDuty. Starting first, GuardDuty allows you to configure alerting-- which alerts appear within the AWS console of GuardDuty. GuardDuty also provides threat intelligence feeds where it receives feeds from cybersecurity experts like Proofpoint, CrowdStrike, and AWS Security who provide lists of known malicious domains and IP addresses. GuardDuty also is great for log analysis and GuardDuty performs analysis of your CloudTrail logs, VPC flow logs, as well as your DNS logs. GuardDuty also displays its findings in the dashboard with the low, medium, or high priority. So as you can see from the screenshot captured here below, you can see your findings which GuardDuty has found, and it also reports on the severity of its finding. Now let's get a better understanding of how GuardDuty works through a given scenario. Let's say we start with a compromised instance. GuardDuty detects that one of our EC2 instances is communicating with the known malicious IP address. GuardDuty then generates a finding, which is generated and appears within the GuardDuty dashboard. It's then our responsibility to take action. For example, our action would be updating the security group to restrict access for the port that is being used. After we restrict access, the malicious actor no longer has access to our compromise instance. So that's generally how GuardDuty works. Now let's talk about setting up GuardDuty. So after enabling GuardDuty, it takes about 7 to 14 days to analyze your account and establish a baseline of normal behavior for your AWS account. And then it's able to generate GuardDuty findings. Findings will appear in the GuardDuty console only if GuardDuty detects behavior it considers a threat. GuardDuty is also available 30 days for free. After 30 days, AWS will charge you. So remember to disable GuardDuty if you are trying it out in your AWS account. GuardDuty also charges based on the amount of data analyzed, including the number of CloudTrail events and S3 events, DNS volume, and VPC flow log data. So for our GuardDuty exam tips, keep in mind that GuardDuty is all about threat detection and it uses machine learning to establish a baseline of normal behavior in your account and detect abnormal or malicious behavior. Also keep in mind that GuardDuty keeps a database of known malicious domains using external feeds from third parties. It analyzes CloudTrail logs, VPC flow logs, and DNS logs. And it also displays findings in the GuardDuty dashboard. Also, findings are displayed within the dashboard with a low, medium, or high priority. Basically, depending upon the severity of the finding. That's all for this lecture on Amazon GuardDuty. If you're ready to move on, you can join me in the next demo where we'll see GuardDuty firsthand.

Demo: Working with Amazon GuardDuty

Hello Cloud Gurus and welcome to this demo on working with Amazon GuardDuty. And in this lesson, we'll get a better understanding of how to use Amazon GuardDuty by taking a walkthrough of the service and revealing its findings. And we'll get started by enabling GuardDuty, which is Amazon's threat detection service that is powered by machine learning. Next, we're going to generate some sample findings, and GuardDuty includes a really cool feature that we can use to generate some findings. And then we'll review the sample findings so that we can understand the type of threats that GuardDuty is able to detect. So let's go ahead and get started, and I'll see you in the AWS console. From the console, search for GuardDuty, and you'll see that you can try GuardDuty for free for 30 days. And then after that, they'll charge you. So go ahead and click on Get Started. And the first thing that I need to do is grant service permissions, and they describe here the kind of permissions that GuardDuty needs. And it needs access to all of these resources in order to generate its security findings. And you can actually review the service role permissions by clicking this button, and these are the two roles that GuardDuty is able to assume in order to

perform threat detection in your account. And down here, you will see the AWS managed policy that is associated with this role. So it's getting quite a lot of permissions in order to perform its threat detection. And there are two different roles that it's allowed to assume. So just close that down and select Enable GuardDuty. Now that it's been enabled, you'll see straightaway that we don't have any findings yet. So if you select Settings on the left, here's my detector ID, which identifies this instance of GuardDuty. Down here, once again, I can view the service role permissions. Scrolling down, we can see that findings are automatically sent to CloudWatch Events, and new findings are exported within 5 minutes. And we can also modify the frequency for updated findings using the options below. And we can also export our findings to an S3 bucket, but I'm not going to do that now. Now under Sample findings, this is a really cool feature that I wanted to show you because we can just click on Generate sample findings, so click on that, and it's going to create one sample finding for each finding type. And then we'll be able to explore the kind of thing that GuardDuty is able to detect. If we scroll up to the top of the screen, you should see a message saying that it's successfully generated some sample findings. And after you've done that, head to Findings on the left. And I can see that I've got a whole list of findings that have been generated. And we can actually sort them based on a severity level. And it's this icon here that tells us what the severity level is. So the red triangle is high severity, the yellow square is medium, and the blue circle is low. So if we click on this arrow, it's going sort by finding type. And so it's sorted by the lowest first, but we want to see the highest criticality. So click the arrow again, and there we get the high severity ones at the top of the list. Now the first thing I can see here is the type of finding it is. And we can see that all these findings are relating to some kind of suspicious behavior that its detected, so maybe somebody up to no good. I can also see the resource that it's relating to, when this particular event was last seen, and how many times it's happened. So let's click on one of these findings. I'm going to click on the S3 malicious IP caller. And then over here, we can see that we've got to a finding ID. We've got an explanation. And basically, it looks like an S3 API call has been invoked from a known malicious IP, and that sounds pretty serious to me, and it's definitely something that I would want to know about if it was happening in my AWS account. And the great thing about GuardDuty is that under the hood, it's using machine learning to learn about all the tricks and behaviors that malicious actors use so that it can recognize when something suspicious is happening in your AWS account. And if we scroll down, you can see what resources were affected, any instances or IAM instance profiles that were affected or involved, and details of the remote actor as well. So now I'm just going to close that view. And it's worth having a scroll through and taking a look at the kind of findings that GuardDuty can detect. So it can detect things like, for instance, a compromised EC2 instance that's behaving unexpectedly. And in this finding, it's noticed that an EC2 instance is behaving in a manner that may indicate that it's being used to perform a DDoS attack. So once again, something you definitely want to know about if it was happening in your AWS account. And then scrolling down, it can also detect activities that are associated with Bitcoin mining. So we can see here, we've got an EC2 instance. It's querying a domain name that is associated with Bitcoin-related activity. And then down here, this is another one of my favorites, anomalous behavior from an IAM user. And this is kind of where the machine learning comes in because it's noticed that this user is performing an activity that is not typically seen from this user. So it knows how does the user normally behave, and it's able to detect when something unusual is happening so that it can alert you. So that's it for GuardDuty. Let's go ahead and review my exam tips. And for the exam, just remember that Amazon GuardDuty is a threat detection service that is powered by machine learning. It generates a finding if unusual or potentially malicious behavior is detected. And here's some of the things that we can expect to see in GuardDuty, evidence of account

compromise, unusual API calls, disabling CloudTrail, compromised EC2 instances or S3 buckets, Bitcoin mining, and reconnaissance activities. And it's well worth having a look through the sample findings so that you can familiarize yourself with the capabilities of GuardDuty. So that's all for this lesson. Any questions, please let me know. Otherwise, I'll see you in the next lesson. Thank you.

Securely Storing Secrets Using AWS Secrets Manager

Hello Cloud Gurus and welcome to this lesson on securely storing secrets using AWS Secrets Manager. And we'll begin with an introduction to Secrets Manager. We'll take a look at storing database secrets using Secrets Manager and the other type of secrets that you can store with Secrets Manager. Next, we're going to have a demo. So we're going to create a secret of our own for an RDS database. We're going to compare Secrets Manager to Systems Manager Parameter Store and finish up with some exam tips. Now Secrets Manager allows you to protect and store your secrets for AWS services, IT resources, and applications, and it allows you to centrally manage your secrets that are used to access resources both inside and outside of AWS. You can also automate the rotation of your secrets without having to deploy any additional code to do so. And it also allows us to secure our secrets using fine-grained permissions like resource policies and encrypt our secrets using AWS KMS, or Key Management Service. Now Secrets Manager allows us to protect several different types of secrets. For instance, you can store secrets for your RDS database, your Redshift cluster, DocumentDB database, other databases, as well as API keys that you might be storing inside or outside of AWS. So now let's talk about the kind of database secrets that can be stored within Secrets Manager. And with RDS, for MySQL, PostgreSQL, Oracle, MariaDB, and SQL Server, all of these allow you to automate the management of database secrets using Secrets Manager. And the kind of information stored in these secrets is going to be things like the database username and password, the server address that you use to access the database, as well as the database name and port. Secrets are encrypted using an AWS KMS key. So it uses a customer master key, and the customer master key is a logical representation of a master key. It holds the key material that's going to be used for encrypting your data. And with a customer master key, you can encrypt up to 4 KB of data. It's best practice to configure the automatic rotation of secrets. For instance, you can configure your secrets to rotate every 30, 60, 90, or a custom number of days with 365 days being the maximum. Secret rotation is performed by Lambda. And when you first create a secret, you'll get the option to either create a new Lambda function, or you can use an existing one if one exists in your account. And it's this Lambda function that's going to perform the rotation. So if it's a database password that you're storing, the function will go in and update the password for you automatically based on the rotation schedule that you define. But we are going to see all of this in action in our demo. And we'll begin by creating RDS database, and we're going to use PostgreSQL. Next, we'll create a secret, and we're going to store our database password as our secret. And then finally, we'll be able to view our secret information in Secrets Manager. So from the console, first of all, I want you to search for RDS. And then select Create database. Select Easy create. Database engine is going be PostgreSQL. Under Database instance size, select free tier. The master username is going be Postgres, and we need to create a master password. So type a master password and confirm it. And you need to remember the password because this is the password that we're going be storing in Secrets Manager. Once you've done that, scroll down to the bottom and Create database. And we'll just need to wait a few minutes for it to complete. And then once it's done, we'll be good to continue. Now that our RDS database has been created, let's go ahead and store our database password in Secrets Manager. So search for Secrets Manager and select that.

Store a new secret. The secret type is going be credentials for an RDS database. And then down here, you're going to type in the credentials that we want to store as a secret. So that's going be our Postgres username and password. And hopefully, you've remembered what your password was. By default, it's going to encrypt your password using a KMS key that is managed by Secrets Manager, or you can create a KMS key that you manage and use that encryption key instead. Next, select your database, and it should have appeared down here. There it is. Click Next. And we need to give our secret a descriptive name so that we can easily find it later. And I'll call it RDS-Prod_PostgreSQL, and I'm going to use that for my description as well. I'll go ahead and add a tag of CreatedBy and then my name so we know who created the secret. Down here, we've got the option to edit our resource permissions. And let's say that you want to give access to a specific Lambda function or EC2 instance, then you can simply paste your resource policy in here. Under Replication, this is where you can replicate your secret to another region, and I'm going to select us-east-2 (Ohio). And once again, by default, it's going to encrypt your secret in the replication destination as well. So then hit Next. And this is where I have the option to add automatic rotation. So select Automatic rotation. Under Rotation schedule, you can create your own rotation schedule using a cron expression if you select this option. But the easiest way is to use the schedule expression builder. And we'll enable rotation on a schedule of 60 days, so update my days to 60. There's also an option to set a duration window. Let's say you've got to a 4-hour window in which you would like to rotate the secret. You can set that in here. And you can also select whether you want the secret to rotate immediately after it's been stored. Scrolling down, if you remember, rotation is handled by a Lambda function, and Secrets Manager can automatically create a new Lambda function for you, or you could use one in your account if you already had one. But we're going to create a new one. We need to give it a name. And we don't need to use any separate credentials. And then hit Next. On this page, we can review all of our settings. And then down here at the bottom, we can see that AWS has provided some sample code in various different programming languages to help us access our secret later on. So this is to help you retrieve the secret when you're in your application. So that's pretty cool that they've provided that, and they've provided it in lots of different programming languages. So now, let's go ahead and select Store. And as we can see, our secret is currently in the process of being created. It does take a couple of minutes to complete everything, including getting the rotation started. After it's finished, we can see that replication has succeeded, and secret rotation is enabled. So now we can select View details, and we can see all the information about our secret, so the encryption key that was used, our secret name, its Amazon resource name, or ARN, our tags. Here's our secret value, and we can retrieve our secret value by clicking this button, and it will show us the details of the secret that we stored. And there is my very insecure database password. It's a good job that we're storing it in Secrets Manager. Scrolling down, you can also see the that rotation is enabled, and our rotation schedule is 60 days. And then down here at the bottom, we've got our replication. And finally, you can access the sample code down here as well. So that is everything that I wanted to show you on Secrets Manager, and you will also need to understand the differences between AWS Secrets Manager and Systems Manager Parameter Store, including when to use each one. So Secrets Manager is great for storing database credentials, API keys, and it can be used to automatically rotate your keys, and it's a great service that is going to help you meet any security and compliance requirements in relation to your database credentials or API keys. Systems Manager Parameter Store, on the other handh, as a wider set of use cases. And as Systems Manager generally is used to perform maintenance updates on your EC2 instances and other AWS resources, Parameter Store is designed to be used to store configuration variables and license keys. So finally, onto my Secrets Manager exam tips. And just remember that Secrets Manager is used to centrally manage

our secrets, rotate them without the need for code deployments, and secure our secrets using fine-graind permissions and encrypt them using AWS KMS. Keep in mind the various different types of secret that you can store in Secrets Manager, so RDS database secrets as we demonstrated, as well as secrets for your Redshift cluster, DocumentDB database, other databases, and API keys as well. And then lastly, make sure you understand when to use Secrets Manager and when to use Systems Manager Parameter Store. And the way that I try to remember it is that Secrets Manager is for anything that's secret or confidential; whereas, Parameter Store is all about configuration parameters. So that's it for this lesson. Any questions, please let me know. And if you're ready to move on, please join us in the next lesson. Thank you.

Section Review: Security and Compliance Summary - Part 1

Hello, Cloud Gurus, and welcome to this lecture, which is part 1one of the Deployment, Provisioning, and Automation Summary. Beginning with EC2, and just remember that EC2 is like a virtual machine hosted in AWS instead of in your own data center, and you can select the capacity that you need right now, grow and shrink whenever you need, pay only for what you use, and wait minutes to deploy your infrastructure, not months. Elastic Block Store provides highly available and scalable storage volumes that you can attach to your EC2 instance, and there are a few different types of Elastic Block Store volumes to be aware of, beginning with the solid-state disk, or SDD, volumes. So first of all, we have gp2, which is a general-purpose SSD, and this is suitable for boot disks and general applications up to 16,000 IOPS per volume. But if your application requires greater than 16,000 IOPS per volume, then you will need to go with a provisioned IOPS volume, and you've got a choice between io1 and io2. And io1 is going to become the legacy option eventually, but it is still available, and you may still see it in the exam. So this is suitable for OLTP and latency-sensitive applications, and you get 50 IOPS per GiB up to 64,000 IOPS per volume, and the provisioned IOPS SSDs are the higher performance and most expensive option. And then we've also got io2, which is the latest generation of provisioned IOPS SSD. It's also suitable for OLTP and latency-sensitive applications, but the difference with io2 is that you get 10 times more IOPS per gig, so you get 500 IOPS up to 64,000 IOPS per volume. I's the latest generation, and it is priced at the same price point as io1. And there's also HDD volumes as well, and there are two HDD options. So firstly, we have st1, also known as Throughput Optimized HDD, and these are suitable for big data, data warehouses, and ETL workloads and a max throughput of 500 MB/s per volume. And remember the HDD volumes cannot be boot volumes. Next, you have sc1, which is Cold HDD, and this has a lower throughput, so the maximum throughput is 250 MB/s per volume. And this is suitable for less frequently accessed data, so think cold data, and it cannot be a boot volume. A bastion host enables you to connect to private instances in your VPC from an untrusted network using SSH or RDP. A bastion is in a public subnet and is reachable from the internet, and you will need to remember to configure the security group associated with the private subnet to enable SSH or RDP access from the bastion. Onto Elastic Load Balancer, and there are a few different types of Elastic Load Balancer that you will need to be aware of, and you will need to understand the differences. So first of all, we have Application Load Balancers, and these load balance for HTTP and HTTPS, and you get intelligent load balancing so you can route requests to a specific web server based on the type of request. And just think about my example of a car dealership website where they're offering sales and loan agreements and service or repairs, and you can use an Application Load Balancer to route requests to a specific server based on the HTTP header. Network Load Balancers provide high-performance load balancing for TCP traffic, so these

are great for low-latency applications, and this is the most expensive option. There's also Classic Load Balancer, and this is the legacy option supporting both HTTP and HTTPS and TCP as well. And even though it's a legacy option, it may still appear in the exam. And then finally, we have Gateway Load Balancers, and these provide load balancing for third-party virtual appliances like firewalls and intrusion detection and prevention systems. And if you need to find the IPv4 address of your end user, then look for the X-Forwarded-For header. And of course, if access logging is enabled, we can also check the load balancer access logs as we saw in the demo. On to load balancer error messages. And if you do see any questions in the exam relating to HTTP codes and HTTP error messages, then I want you to ask yourself, is this a client-side error or a server-side error? And just remember that a 400 error is a client-side error and a 500 error is a server-side error. And my way of remembering it is that the number 5 looks a little bit like the letter S when I take my glasses off. And as a SysOps administrator, we are more concerned with a 500 error. And there are a few common error messages that you should just be aware of. Firstly, we've got HTTP 504: Gateway Timeout. The Elastic Load Balancer could not establish a connection to the target, so something is wrong in your application, and you will need to identify where the application is failing and fix the problem. Next, we have HTTP 502: Bad Gateway, and this means that the target host is unreachable, ao you will need to check, for example, that your security groups are allowing traffic between the load balancer and the target hosts. And then finally, we have HTTP 503: Service Unavailable, no registered targets. So just go in and check that you have registered the correct targets and that they are successfully registered. On to Elastic Load Balancer access logs, and just remember the Elastic Load Balancer access logs capture information relating to incoming requests to your Elastic Load Balancer. They are disabled by default, and you will need to go in and enable them, and they are stored in S3, so by default they're encrypted and stored in an S3 bucket, and then they are automatically decrypted when you access them. We use sticky sessions to override the load balancing algorithm, and sticky sessions use cookies to identify a user session and send requests that are part of the same session to the same target. And these are great for applications which cache session information locally on the web server, so consider shopping carts, online forms, or even a training website where we don't want to log out our customers halfway through a task. On to Elastic Load Balancer Cloudwatch metrics. And the Elastic Load Balancer sends metrics into CloudWatch by default, and by default, you get metrics relating to HealthyHostCount, UnhealthyHostCount, RequestCount, TargetResponseTime, and HTTP status codes as well. On to EC2 Image Builder. And EC2 Image Builder automates the process of creating and maintaining Amazon Machine Images and container images. And it's a four-step process. So you begin by selecting a base operating system image, you customize by adding the software that you would like to install, test the AMI, and then distribute to your chosen region, and EC2 Image Builder handles this entire process for you. And there is some terminology that you will need to be aware of. So first of all, image pipeline defines the settings and process that you are configuring within Image Builder. Image recipe is the source image and build components. And build components refers to the software that you would like to include in your AMI. So that's it for this lecture. If you have any questions, please let me know, and if you are ready for part 2, then please join me in the next lecture.

Section Review: Security and Compliance Summary - Part 2

Hello, Cloud Gurus, and welcome to this lecture, which is part 2 of the Deployment, Provisioning, and Automation Summary. Beginning with CloudFormation, and just remember that

CloudFormation allows you to manage, configure, and provision AWS infrastructure as YAML or JSON code, and you will need to remember the different sections of the CloudFormation template. So we've got Parameters, which allows you to input custom values, for example, input the name of an SSH key. Conditions allows you to provision resources based on environment. The Resources section is mandatory and describes the AWS resources that CloudFormation will create. Mappings allows you to create custom mappings, for example, mapping a region to an AMI that you would like to use in that region. And then we also have the Transform section, which allows you to reference code located in S3. For example, Lambda code to provision a Lambda function or reusable snippets of CloudFormation code. CloudFormation StackSets allow you to create, delete, and update CloudFormation stacks across multiple AWS accounts and regions using a single operation. And to do this, you will need to set up some cross-account rules if you are working with multiple accounts. So for the Administrator account, you can use the AWSCloudFormationStackSetAdministrationRole, which is allowed to assume the AWSCloudFormationStackSetExecutionRole to provision resources in the target accounts. And then once you've created resources in these other accounts, you can use Resource Access Manager to share the resources between your accounts. For example, you could share resources like EC2 instances, S3 buckets, and EC2 Image Builder images. When it comes to troubleshooting CloudFormation, you can use the CloudFormation console to view the status of your stack and any error messages. And common error messages include insufficient permissions. And if you see an error like that, you'll need to go in and add permissions for the resources that you are trying to create, delete, or modify. If you see the resource limit exceeded error, then request a limit increase, or you could also delete any unnecessary resources and retry. And if you see UPDATE_ROLLBACK_FAILED, then you will need to fix the error causing the failure and retry. For example, if you deleted or changed a resource manually, which you then try to roll back or delete using CloudFormation, then CloudFormation will throw an error because the resource is not in the state that it's expecting, so you will need to get your stack back to the state that CloudFormation is expecting in order to continue. Now if you are troubleshooting CloudFormation, then the first place to check is the CloudFormation console, and under Status reason it will give you some pretty good information that is going to give you a clue as to what went wrong. And in this example, I used an incorrect AMI, which did not exist in the region that I was operating in. On to CloudFormation best practices. You should be controlling access to CloudFormation using Identity and Access Management. Be aware of your service limits because if you hit a limit, then CloudFormation will fail to create your stack. Avoid manual updates because they're going to cause errors when you try to update or delete the stack. Use CloudTrail to track all changes in CloudFormation, along with who made them and when. And use a stack policy to protect critical stack resources from unintentional updates and mistakes caused by human error. On to blue/green deployments. And a blue/green deployment strategy is a low-risk strategy. So blue is the current version of your application, while green is the new version. And after testing is successfully complete, live traffic can be directed to the new version, and the old version will become deprecated. Rolling back is fast and easy because if something goes wrong after the new version is being used in production, then simply redirect customer traffic back to the original environment. A rolling deployment allows you to deploy new application versions and other changes in batches. It's very cost effective because you only need one environment, and you can set the batch size and the minimum number of servers to keep in service in your environment. But the tradeoff is that it comes with some added complexity because for a while you will be operating with a mixed environment with some instances running your old version and some running the new version. And in addition to

that, rolling back will involve a redeployment. When it comes to canary deployments, remember the canary in the coal mine. And just like the canary in the coal mine, you get an early warning system that can indicate that something is wrong in your application. And with a canary deployment, you deploy the new version to a small number of servers, direct a small proportion of customer traffic to the new servers, for example, 10%, and this allows you to test your application with a small proportion of real customers before you roll out to everybody. And a canary deployment is a great way to reduce risk of deploying a new application version when you are working with a production environment, for example, if you're installing new code or new packages. And to roll back, simply direct 100% of users back to the original version. On to OpsWorks. And just remember OpsWorks is an automated configuration management service compatible with Puppet or Chef, and if you see anything in the exam relating to Puppet or Chef, then I want you to think of OpsWorks. And it allows you to use a Configuration as Code approach to managing the configuration of your operating systems, application settings, and packages as well. You get managed instances of Puppet or Chef. And there is also OpsWork Stacks, which allows you to model your application as a stack consisting of multiple layers, for example, a database layer, a web server layer, application layer, and load balancer layer, etc. On to Systems Manager Run Command. And imagine you're a SysOps administrator supporting hundreds of servers, and your manager asks you to run a command to check the network configuration on each server. Well, this is something that you can really easily do using Run Command. So using Run Command, you can run a command of your choice on a group of servers simultaneously without having to log in to each one individually. And there's also Systems Manager, which we can use to patch multiple EC2 instances simultaneously. So imagine you've got hundreds of service to patch. It's going to be pretty boring and laborious if you have to patch each one of these manually, so you can just go ahead and use Patch Manager to patch a group of servers simultaneously. So that's it for this lecture. If you have any questions, please let me know; otherwise, we will see you in the next section of the course. Thank you.

Networking and Content Delivery

Section Introduction

Hello, Cloud Gurus, and welcome to this section of the course, which will cover Networking and Content Delivery. And this is domain 5 of the exam. So in this section, we are going to be covering building out your own custom VPC, VPC flow logs, VPNs and Direct Connect, DNS and Route 53, and CloudFront as well. And it's going to be very hands-on because it's really important to get as much practical experience as possible, and going into the exam, you will need to be confident that you can build out your own custom VPC from memory. So if you're ready to get started, I'll see you in the next lecture. Thank you.

Virtual Private Cloud (VPC) Overview

Hello Cloud Gurus, and welcome to this lecture, which is going to cover VPCs. Now, if you've recently completed our Solutions Architect Associate course, then you will already be very familiar with VPCs, and this is going to be a bit of revision for you. But it is a really important topic for the exam, and you will need it to know it inside out in order to pass. So, first of all, we're going to have a quick introduction to VPCs. We'll take a look at networking and VPNs. We'll explore an example network diagram for a custom VPC. Next, we'll review some of the cool features of VPCs. We'll compare a default VPC with a custom VPC. And then finally, we'll take a look at some of our exam

tips. So let's get started. So what is a VPC? Well, I want you to think of a VPC as a virtual data center in the cloud. So it's a logically isolated part of the AWS cloud where you can define your own network and install AWS resources, for example, EC2 instances. And when you create your own VPC, you will have complete control of the virtual network, including defining your own IP address range, subnets, route tables, and network gateways as well. So a VPC allows you to define your own fully customizable network, and it allows you to configure multi-layered security using mechanisms like network access control lists, or ACLs, and security groups as well, to control access to the EC2 instances inside your subnets. And if you think about a classic 3-tier architecture where you have a web tier, an application tier, and a database tier, you can create a custom VPC with all of the network segregation needed to configure this type of architecture. So we've got a web tier with a public facing subnet. We've got our application tier, and in this example, it's a private subnet which can only speak to the web tier and the database tier. And then finally, we have our database tier, which again is a private subnet, and it can only speak to the application tier. So just like a network in your own data center, you can configure all of this in your own VPC, and you can also establish a hardware Virtual Private Network connection between your own data center and your VPC, and this enables you to use the cloud as an extension of your own data center. And of course, with a VPN, you get a secure, private, encrypted tunnel between your data center and AWS. So now, let's take a look at an example of a custom VPC. So here is our Region, and it's us-east-1 and then inside our Region is our VPC, and our VPC is named "acloudguru", and it has a CIDR range of 10.0 .0 .0 /16. So that is our CIDR range or our IP address range. And then we can connect into our VPC using an internet gateway or a virtual private gateway. So remember, the internet gateway is for internet access, and the virtual private gateway is used to configure a VPN. And then when the traffic comes into our VPC, it's going to hit a router or router, and we can configure our own route tables or route tables, however you like to say it, and they are going to define how the traffic is going to be routed or routed within our VPC. So the traffic comes in through the route table, and then it's going to hit our network access control list, and then it will hit our subnets. So here, we've got a public subnet, which means it has internet access and it's accessible from the internet, and then here we have our private subnet which is not accessible from the internet. And in my example, you can only reach this private subnet using the VPN. So you've got to come in through the virtual private gateway. And then each one of our subnets is associated with a security group, and this security group is going to protect our instances. And of course, a security group behaves a little bit like a firewall, only allowing the access that you define, for example, if we enable SSH access on port 22 from only a specific IP address range. And of course, 1 subnet equals 1 Availability Zone, and a subnet cannot span multiple Availability Zones. So our public subnet is located in one Availability Zone, and our private subnet is located in another Availability Zone. And then when it comes to internal IP addresses, they actually recommend that you specify a block of IP addresses known as a CIDR range from one of the following IP address ranges. So we've got 10.0 .0 .0, 172.16 .0 .0, and 192.168 .0 .0. And do just bear in mind, the prefixes that we have here. So a /8 prefix is going to give you the largest IP address range, /12 will give you a medium-sized range, and /16 is going to give it the smallest range, but even a /16 range is still going to be absolutely massive. And there's actually a really quick and easy way to calculate how many IP addresses a specific range is going to give you. And if you head over to this website, it's called CIDR.xyz, you can input the values that you were thinking of using for your IP address range, and it will automatically calculate everything for you. So here is our net mask. Here's our first usable IP, here's our last usable IP, and here is the number of IP addresses that we get with this subnet mask. So we get 16 million IP addresses. And then what happens if we change the subnet mask to 16? That's going to

give us 65, 000 IP addresses. And then if we change it to /24, we're going to get 256 IP addresses. So that is a great site that you can use to check out your CIDR ranges when you are planning out your VPC. So what can we do with a VPC? Well, we can create subnets and then launch instances into the subnet of our choice. We can assign custom IP address ranges for each of our subnets. Configure route tables to route traffic between our subnets. We can create an internet gateway and attach it to our VPC to enable internet access, and we can control which networks can access our AWS resources, and the 2 main security mechanisms that they provide to enable us to do that are network access control lists, or ACLs, and security groups as well. And if you've done our Solutions Architect Associate course, then you will remember that you can use a network access control list to block specific IP addresses. And don't worry, throughout this section of the course, you are going to get plenty of hands-on experience with all of these technologies. So what is the difference between a default VPC versus a custom VPC? And up until now, throughout this course, whenever we've created an EC2 instance, we've just launched it into the default VPC. So you've probably been using the default VPC without even thinking about it. So the default VPC is very user friendly and it's really easy to use. All subnets in the default VPC will have a route out to the internet, and each EC2 instance that you launch in the default VPC will have both a public and a private IP address, whereas with a custom VPC, this is fully customizable. So this means that you can configure it exactly the way you want, and you can make it as secure as you need to. So you decide if subnets are going to be public or private. And of course, you can decide whether an EC2 instance is going to have a public IP address or not. And finally, as with anything customizable, it does take additional time to set all of this up. So, on to my exam tips. Just remember that a VPC is like a logical data center in AWS, consisting of subnets, route tables, network access control lists, security groups, and internet gateways, or of course, virtual private gateways as well. And remember that 1 subnet is always in 1 Availability Zone, and you cannot have a subnet spanning multiple Availability Zones. So that's it for this lecture. If you have any questions, please let me know. Otherwise, I will see you in the next lecture. Thank you.

Demo: Building Your Own Custom VPC - Part 1

Hello Cloud Gurus and welcome to this demo where we're going to be building our own custom VPC. And we'll begin by creating a new VPC. We'll name it acg-vpc, and use a CIDR range of 10.0.0.0/16. Next, we'll create a private subnet. Call it 10.0.1.0-us-east-1a-dash priv. We'll also create a public subnet called 10.0.2.0-us-east-1b-pub. We'll create an internet gateway and attach it to our VPC. And then finally, we'll create a route table so that we can route internet traffic to the internet gateway and then associate our public subnet with our route table. So if you'd like to join me in the AWS console, we'll get started. So from the console, the first thing to do is search for VPC. Make sure that you're in the us-east-1 region and select Your VPCs. And you'll notice that there's already a VPC in your account, and this is just the default VPC that they give you. However, we are going to go ahead and create our own custom VPC. So select Create VPC. We'll only be creating a VPC at this point. The name is going to be acg-vpc. CIDR range is 10.0.0.0/16. Down here, we can also create an IPv6 CIDR block, but we're not going to do that. Under Tenancy, you've got a choice either to stick with the default shared tenancy, or you've got the option to use dedicated hardware. And we're going to stick with the default. It's automatically created a tag for us. So let's just go ahead and Create VPC. So that's our VPC created. And if we select Route tables on the left, you should see the that it's already created a default route table for our VPC, and there it is. If we scroll down to Network ACLs and select that, you'll see it's already created a default network ACL

for our VPC as well. And if you scroll down to Security groups, there's also a default security group that's been created for our VPC as well. So at this point, we've created our VPC, and we've got a default security group, network ACL, and default route table as well. So what about the subnets? Well if we scroll back up to the top and select Subnets, you'll see the that there are already some subnets in here. But these are all associated with our default VPC. They're not for our new VPC. So the next thing we need to do is create our subnets. So select Create subnet. At the top, we'll select our VPC. Down here, it's reminding us of our CIDR range. Scrolling down to Subnet settings this is where we can create our first subnet. And our first one is going to be called 10.0.1.0-us-east-1a-prive. Availability zone is going be us-east-1a. The CIDR range is going be 10.0.1.0/24. It's already added a tag for us, and now we can add the second subnet. So click Add new subnet, and this is going be our second subnet. We'll call it 10.0.2.0-us-east-1b-pub. Availability zone is going be us-east-1b. And CIDR range is 10.0.2.0/24. And once you've done both of those, click Create subnet. So that is our two subnets created. So let's just check on where we are in the process. We've created our VPC with the default security group, network ACL, and route table. And we've also created our private and public subnets. But at the moment, both of our subnets are actually private because we've got no access to the internet. So, that is the next thing we're going to configure. So at this point, we are going to add our internet gateway. So using the menu on the left-hand side, select Internet gateways. You'll probably see the that you've already got an internet gateway here, and this one is attached to our default VPC. So let's select Create internet gateway. I'll call it my-igw and then select Create internet gateway. So that's our gateway created. But in order for our VPC to connect to the internet, we will need to attach this gateway to our VPC. So using the drop-down on Actions, select Attach to VPC. Select our VPC, and you'll notice that it's only going to allow you to attach this internet gateway to our new VPC. We won't be able to attach it to the default VPC. And that's because you can only attach one internet gateway per VPC, and the default VPC already has one. So select your VPC and Attach internet gateway. So now, in addition to attaching our internet gateway to the VPC we also need to create a route table, and that's going route our internet traffic using the internet gateway. And we'll need to associate that new route table with our public subnet. So head to Route tables on the left. You'll notice here that we've already got a route table for our VPC, and it's that one. And this is just the default route table associated with our VPC. And by default, whenever you create a new subnet, it's going to use this default or main route table. Now we don't want every single one of our subnets to be accessible over the internet, So let's create a new routing table and attach it only to our public subnet. So select Create route table. I'll call it MyPublicRouteTable. Select your VPC and Create route table. Next, we need to add our route to the internet. So select Actions. Edit routes. By default, there's already a route in here, and this route simply allows all of the subnets in your VPC to route traffic to each other by default. So select Add route. The destination is 0.0.0.0/0, which is the IPv4 notation, meaning the entire IPv4 address space, in other words any server on the internet. The target is going be our internet gateway, so select Internet Gateway. And it should find your internet gateway in the there. So now, select Save, and the next thing we need to do is to configure the subnet association for this routing table. So from the Actions drop-down, select Edit subnet associations. Here's our public subnet. So select your public subnet and Save associations. So now, at this point, we've got our VPC, we've got our private and public subnets, we've configured an internet gateway, and we've created a route table with a route to the internet and associated it with our public subnet. So that is the end of part 1. And at this point, we're actually ready to go ahead and test that everything is working, and that's exactly what we're going to do in the next lesson. So if you'd like to join me for that, I will see you in the next lesson. Thank you.

Demo: Building Your Own Custom VPC - Part 2

Hello and welcome back, and this is part 2 of Building Your Own Custom VPC. And at this point, we've already created our VPC. We've configured private and public subnets, added an internet gateway and our own route table as well. So the next thing we're going to do is launch two EC2 instances, one in each of our subnets. We'll then configure a security group, allowing SSH and HTTP to our instance in our public subnet. And then, finally, we're going to try and log into our public instance from our local machine using SSH. So if you'd like to join me in the AWS console, we will continue. So from the AWS console, we are going to create our public instance. But before we do that, there's one thing that I want to change, and that's a setting in our public subnet. So search for VPC, select Subnets, select your public subnet, select Action, Edit subnet settings, and we're going to modify the auto-assign IP settings. And this is going to allow us to auto-assign a public IP address for any EC2 instance that gets created in our public subnet. And by default, this is not enabled. So once you've enabled that, scroll down and hit Save. So now, we're ready to launch our instance. So search for EC2. Launch instance. We're going to call it MyPublicInstance. We'll use the Amazon Linux operating system. Instance type is going to be t3.micro. We'll create a new key pair so that we can log into our instance. I'll call it nvkp and Create key pair, and it's going to download to my local machine. Under Network settings, select Edit, select your VPC, select your public subnet. Check that Auto-assign public IP is set to Enable. Under Security group, we're going let It create a new security group, but I want to change the name of the security group so that I can find it later. And I'm going to call it MyWebDMZ. By default, it's going allow SSH on port 22 from anywhere. We'll accept all the rest of the defaults and Launch instance. So that is our public instance launched. Let's go ahead and launch a private instance as well. So come back to EC2, select Launch instance, call it MyPrivateInstance, use Amazon Linux 2. Instance type is t3.micro. We'll select the same key pair that we just created. Under Network settings, select Edit. Select your VPC. And this time, we're going to use the private subnet, so make sure that that one is selected. Auto-assign public IP is going to be set to Enable because this is our private subnet and, by default, it's not going to give us a public IP. We'll create a new security group, rename it to MyPrivateSG. And I'm going to remove the SSH access because we don't want to enable SSH access for the world because this is our private subnet. So just go ahead and remove that. And I'm going to leave the security group without any rules at the moment. Once you've done that, all the rest of the defaults are fine. So go ahead and Launch instance. Then, select Instances. We can see that one of our instances is ready, and our private instance is still initializing. And it might just take a few moments for everything to be ready. So just be patient. And in a minute or so, we should have a couple of EC2 instances. After a few moments, let's refresh using this button. And there we go. They are both complete. So now, we're ready to log into our public instance. Select our instance, select Connect, and I'm going to select my SSH client. And this is going to give you some instructions if you ever forget how to connect to your EC2 instance using SSH. Now first of all, we need to change the permissions on our private key. And this is the command that we're going to use, so we can just copy it from here. Come to a terminal on my local machine. And I'm in my Downloads directory, and this is where my SSH key was downloaded to. So now, I can paste the command that I just copied and hit Enter, and that's going to set the permissions for my SSH key pair. Then, the second command is to connect your instance using its public IP, and here's the command here. So I'm going to copy that. Then back in my terminal window, just paste my command in there and hit Enter. We can answer yes to this question, and there we are. We're on our EC2 instance. So we've connected to our EC2 instance using SSH over the internet. I'm just going to clear the screen. And to really confirm that this

instance is connected to the internet, we can run a sudo yum update -y. And this will update any operating system packages to the latest version. So there we go. It's completed successfully. Our instance has reached out to the yum repository over the internet, downloaded, and installed all the latest operating system updates. But how are we going to log into our private instance? Well let's come back to the AWS console, come back to our EC2 instances, select our private instance, select Connect. And first of all, it's warning us that we don't have port 22 open to allow SSH connectivity over the internet. But it's also telling us that if we want to connect using SSH, we will need to use the private IP address because there's no public IP address on this instance. If we come back to the instance itself, we can see that there is no public IP address. However, we do have a private IP address. So right now, this instance is completely isolated from the internet, and we've got no way to SSH into it from our local machine. So in order to SSH onto this instance, we can add a bastion host, and that's exactly what we're going to be doing in the next lesson. And that's going to allow us to SSH from our local machine. So, let's just recap on this lesson. We started off with our VPC, and we had a private and public subnet configured. We also had an internet gateway and a route table configured with a route to the internet. Then, we launched two EC2 instances, one in each of our subnets. We created a new security group called MyWebDMZ. And then finally, we connected to our public instance using SSH from our local machine. And in the next lesson, we'll be creating a bastion host, and we'll use the bastion host to connect to our private instance. So if you'd like to continue, please join me in the next lesson. Thank you.

Demo: Building Your Own Custom VPC - Part 3

Hello Cloud Gurus and welcome back, and this is part 3 of Building Your Own Custom VPC. So the starting off point from the previous lesson is that we have our own custom VPC named acg-vpc. We've got our private and public subnet. We've also got an internet gateway with a custom route table and a security group, allowing SSH from our local machine. So the next thing we're going to do is configure a bastion host in our public subnet. We'll then log into our bastion host using SSH from our local machine. We'll then configure a security group, allowing SSH from our bastion host to our private instance. And then finally, we're going to create a NAT gateway and configure a route to the internet for our private instance, so it will have outgoing access to the internet. And with the NAT gateway, it's going to be outbound access only. It's not going be inbound access to our private instance. So if you're ready to get started, then I'll see you in the AWS console. So from the console, first of all, I'm going to search for EC2 and Launch instance. The name of my instance is MyBastionHost. We'll use the Amazon Linux 2 AMI. Instance type is going be t3.micro. We'll use the same key pair that we created earlier. Under Network settings, select Edit. Select your VPC. And we need to select our public subnet. Make sure that Auto-assign public IP is set to Enable. Under Security groups, we want to be able to SSH into this bastion host, but we also want our bastion host to be able to SSH into our private instance. So I'm going to create a new security group. I'll give it a name so that I can Identify it later. By default, we've got SSH access, and that's what we need. And now we're ready to go ahead and launch our instance. Let's come back to our EC2 instances view and check the progress. And as soon as it's finished initializing, we are good to continue. As soon as your instance is running, select your instance. Select Connect. Come to SSH client, and I'm just going to be lazy and copy the command from here. Then, come to my terminal window on my local machine. Make sure you're in your Downloads directory where your SSH private key is saved. Paste in your command and hit Enter. We'll answer yes to this question. And there we are. We are on our bastion host. Now the next thing we're going to do is configure access

between our bastion host and our private instance. And to do that, we need to update the security group of our private instance. So head back to the AWS console. Come back to Instances. I'm going to select my private instance. Select Security. And here's my security group that is associated with my private instance. So select your security group, and we need to edit the inbound rules. At the moment, there are no rules in the this security group. So select Edit inbound rules, Add rule. The type is going to be for SSH. Protocol is TCP, port range 22, and the source is going to be our bastion host security group. So in this search box, if you type sg, it will bring up all of your security groups. And this is the one we want, MyBastionSG, so select that and Save rules. Now at this point, we should be able to connect to our private instance from our bastion host. So come back to our instances. We'll select EC2, Instances. Select your private instance. Hit Connect. Make sure you're in the SSH client. And because this is the private instance, you need to connect to your instance using its private IP. So this is the command that we're going to run, this one down here, using the private IP of our instance. However, in order to connect from our bastion host, we need to create this private key on our bastion host. So from a new local terminal window, so I'm on my local machine, I'm going change directory to Downloads. Then run cat and the name of my key pair and hit Enter. And there is my private key. So now, I need to copy this private key. So copy everything and just make sure that you've got everything. I'm going to come back to my bastion host. Type vi, the name of our key pair, then hit Enter. Then hit Escape, i for insert, and then we're going to paste in what we just copied. So that will be Ctrl+V or Cmd+V depending on your machine. Then, hit Escape colon, w to write, q to quit, and then bang or exclamation mark and hit Enter. So that is our SSH key copied across. The next thing we need to do is set the permissions. So run chmod 400 the name of your key pair and hit Enter. And then we can test it using this command. So this was the command to connect to our private instance. So paste that and hit Enter. We'll answer yes to this question, and there we are. We are connected to our private instance. Now that we're connected to our private instance, let's see what happens if we run a sudo yum update -y. I'll clear my screen. Well, it's not going to work, and that's because we do not have an internet connection from this system. So in order to run a yum update, we'll need to add a NAT gateway, and that's exactly what we're going to do next. So let's head back to the AWS console. Let's search for VPC. And then, on the left-hand side, find NAT gateways and select that. Select Create NAT gateway. I'll call it MyNATGateway. We need to create it in a public subnet, so make sure you've selected your public subnet. Connectivity type is Public. We need to allocate an elastic IP, so select that option, and Create NAT gateway. Now it can take a moment for your NAT gateway to finish initializing. So while we're waiting, let's set up our route table. So let's head across to Route tables on the left. Create root table. We'll call it MyPrivateRouteTable. Select our VPC and Create route table. Next, under Routes, select Edit root. And we're going to add a new route. The destination will be 0.0.0.0, and the target is going to be our NAT gateway. So select NAT gateway, and it should find your NAT gateway. So select that and Save changes. Next, we need to associate this route table with our subnet. So from Actions, select Edit subnet associations, and we're going to associate this route table with our private subnet. So select your private subnet and Save associations. So now, we should be good to go, but let's just check on our NAT gateway and make sure it's finished initializing. So on the left, find NAT gateways. It should be in a state of available. So now we can head back to our terminal window, and I'm back in my session on my private instance. I'm going to clear the screen. And then we can hit the up arrow. I'm going to hit it twice to get back to my sudo yum update command and hit Enter. And if it's all worked, you should see your machine starting to do the sudo yum update. And there we go. It's reached out over the internet to the yum repository, it's downloaded all the various packages, and it's done at the installation. So there we go. We've

added outbound internet access for our private instance. So now, let's just recap on everything that we've completed. And if you remember, we started off this lesson with our VPC, which had our public and private subnets. We added a bastion host in our public subnet, and we configured a security group, allowing SSH from the bastion host to our private instance. We then added a NAT gateway, and we created a route table allowing outbound access to the internet from our private instance. And we verified the outbound internet access by successfully running the yum update command. So that is it for this lesson. If you have any questions, please let me know. Otherwise, please join me in the next lesson. Thank you.

Demo: Configuring Network ACLs - Part 1

Hello Cloud Gurus and welcome to this demo where we're going to be getting our hands dirty configuring network ACLs. And first of all, we're going to review our default VPC. And with the default VPC, a default network ACL is automatically created along with a default security group and a default route table with a route to the internet. The next thing we'll do is launch an EC2 instance in a public subnet, and we'll be configuring this EC2 instance as a web server. Next, we'll create our network ACL, and we'll configure it to allow HTTP and SSH access to our public subnet. And we'll test that we can access our host using SSH and HTTP. And then finally, we're going to configure our network ACL to deny access from our own IP address. So if you'd like to join me in the AWS console, we'll get started. So in the AWS console, search for VPC. On the left, select your VPCs, and here is my default VPC. On the left, if you scroll down to the Security section and select Network ACLs, here's my default network ACL, and it's associated with all of the subnets in my default VPC. So let's select our network ACL. And if you scroll down, you will see the that there are separate rules for inbound connections and outbound connections as well. And if you come to your inbound rules, you should see the that we get a rule that allows all traffic on all protocols, all port ranges, and from all sources. So by default, all traffic is allowed. If you take a look at the outbound rules, there is a corresponding rule for all outbound traffic. So all traffic, all protocols, all port ranges, and all destinations is allowed. And just notice here that you can also create deny rules as well. And then over here under Subnet associations, it shows you all the different subnets that are associated with this network ACL. So that is the default network ACL. So now, let's go ahead and create an EC2 instance. From the top of the screen, we can search for EC2. Select Launch instance. The name is going to be MyWebServer. Select Amazon Linux as the operating system. Instance type is going be t3.micro. We're going to create a new key pair because we want to be able to log into this instance using SSH. Under the Network settings, select Edit. Make sure that your default VPC is selected. Under Subnet, I want you to select the subnet that is in the us-east-1a availability zone. So make sure that one is selected. Make sure that Auto-assign public IP is set to Enable and then scroll down to the firewall security groups settings. We're going to create a new security group. I'll call it MyWebDMZ. By default, of course, you get SSH, but we're also going to add HTTP as well. So select Add security group rule. Type is HTTP. Protocol is TCP on port 80. The source type is going to be Anywhere. And then once you've done that, we are ready to launch the instance. Once it's initiated, select the instance ID up here. And in a few moments, our instance should be ready. I'll refresh my screen, still initializing, and there we go. Our instance is ready. So now, let's log into our instance. Select it using this checkbox here on the side. Select Connect. And this time, we're going to use EC2 Instance Connect to connect. So just hit Connect, and this is just a really easy way to get a session on our EC2 instance. So it's going through the AWS console, but we're still using SSH. So there we are. We are now on our instance. Now first of all, we're going to run sudo su to become

root. Next, we'll run yum update -y to update the operating system. And this just updates our operating system in case any packages need updating. I'll clear my screen. And then the next thing we need to do is a yum install httpd -y, and this is just going to install httpd. Once that's finished, I'm going to change directory to the var/www/html directory. And now we're going to create a really simple web page. So type echo open quote, and it's going to be an HTML page. Here's our heading. Then, we're going to close our heading, close our HTML, close quote, and we're going to redirect everything to a new file named index.html and hit Enter. And there is our newly created file. So now we're ready to start up our web server. So type systemctl start httpd and then systemctl enable httpd. And this command simply makes sure that httpd is going to start up every time we reboot our instance. So now, we're ready to test that our web server is running properly. So you can close your tab for EC2 Instance Connect. And over here, we can grab our public IP address of our EC2 instance. So copy the IP, come to a new browser tab, paste in your IP, and hit Enter. And there we go. That is our simple website. So now, we are ready to go ahead and create our own network ACL. And that is exactly what we're going to be doing in part 2 of this lesson. So that's the end of part 1. And if you'd like to continue, I'll see you in part 2. Thank you.

Demo: Configuring Network ACLs - Part 2

Hi and welcome back. And this is part 2 of Configuring Network ACLs. And in part 1, we created an EC2 instance. And now we are ready to configure our network ACL. So come back to the AWS console. We're going to search for VPC. On the left-hand menu, scroll down and select Network ACLs. We're going to create a new network ACL. I'm going to call it my-nacl. Select your default VPC. Scroll down to the bottom and Create network ACL. So there is our network ACL. I'm going to select it up here. And in the bottom half of the screen, you can see the details of your network ACL. So here's my inbound rules, my outbound rules, and subnet associations. And by default, every new network ACL that you create yourself is going to deny everything by default. So in terms of inbound rules, we only have this single rule and then the same again for the outbound rules. So by default, it is denying everything. So let's go ahead and associate this network ACL with our subnet. Select Subnet associations and Edit subnet associations, and we're going to associate it with the subnet that's in our us-east-1a availability zone. So make sure that you select that one and Save changes. So now, if we come back to our web page and paste in the public IP of our EC2 instance, you should see the that you can no longer access the web page. And we can see that the browser is just spinning up here. And eventually, we're going to get a timeout message. And there we go. After a few moments, I received my Gateway Timeout message. So we can no longer access our web page. So let's head back to our network ACL and see if we can fix this. If we head to Inbound rules and Edit inbound rules, we can add a new rule. The rule number is going to be 100. Select HTTP because we want to allow HTTP traffic. Protocol is TCP. Port range is 80. Source is Anywhere, and we're going to allow this traffic. And if you select Info over here next to Rule number, it tells you that the rules are evaluated, starting with the lowest-numbered rule. And as soon as a rule matches traffic, it is applied regardless of any higher-numbered rule that might contradict it. So this is going to be the first rule that gets evaluated and applied, and any other higher-number rule that appears after it is going to be ignored if the traffic matches this rule. And then one last thing I wanted you to notice is that we can also deny traffic as well. So network ACLs let us create rules to deny traffic, as well as allow it. So if you're happy with that, hit Save changes. Now we also need to create an outbound rule as well. So select your outbound rules, Edit outbound rules, add a new rule. The rule number is 100. Type is going be HTTP. Destination is anywhere. We're going to allow the traffic

and save changes. So now let's come back to our web server and see if we fix the problem. So reload your page. And unfortunately, this still has not fixed our problem. So what could be going wrong here? Well, let's come back to our console and take a look at our rules. So here's our inbound rule, and here's our outbound rule. But the problem here is that I've specified this outbound rule using port 80. However, that is not going to work, and I'll explain to you why that is. Because even though the incoming request is coming in using port 80, the outbound reply from our web server, it does not reply back on port 80. So this outbound rule is never going to work. So select Edit outbound rules. We'll need to change the type to Custom TCP, and then it will allow us to change the port range. But what do we need to change it to? So we need to provide a different port range that includes the ports that our web server is going respond back to, and these are known as the ephemeral ports. And if show you in the AWS documentation, you'll see exactly what I mean. And it's actually the end user client initiating the request that chooses the ephemeral port range. And the ephemeral port range will vary depending on the client's operating system. So this is all about the web client. And there are a few different port ranges. You don't need to learn these ports for the exam. You just need to be aware that there are these ephemeral ports, and they give a really good example down here. So for example, if a request comes into a web server in your VPC from a Windows 10 client on the internet, your network ACL must have an outbound rule to enable traffic destined for these ports. However, in practice, to cover all the different types of clients that might initiate traffic to your public-facing instances, then you can open up this set of ephemeral ports. And it's 10242-65535. So I'm just going to copy that port range, come back to my management console, and I'm going to paste in the that port range. So this is the range of ports we're going to use, and it's actually a massive range, but it's going to catch all of the ports that we expect our web server to be replying on. So once you've done that, hit Save changes, and then we can come back to our web page. I'm going to refresh, and there we are. It should all be working. So we've configured our network access control to allow access. And the last thing I want to show you is how to deny access using our network access control list, and we're going to deny access to our website from our local machine. Now in order to do that, I need to find out the IP address that my machine is using, and there's a website we can use for that. It's called whatismyip.com. And if you go to that address, it will tell you what IP address your system is using. So I'm just going to copy my IP address, come back to my management console, select my inbound rules, edit inbound rules. We're going to add a new rule. Rule number is 200. It's going to block HTTP access. And the source is going be my IP address followed by forward slash and 32. We'll set the rule to Deny because we're going be denying traffic from this IP and save changes. So now, let's come back to our web page. Hold down Shift and Refresh, and it hasn't worked. We've still got access to our website. Why do you think that is? Well, the reason it hasn't worked is because of our rule numbers, and I wanted to show you how the rule numbers actually work. So if you come back to our inbound rules, here's our first rule, rule 100, which allows everything. That's the first one that's getting evaluated, and that's the first one that's getting applied. The second rule to be evaluated is 200, but we've already applied this rule. So anything that comes after rule number 100 is not going to get applied. So if we want to apply this rule, we need to give it a rule number of below 100. So let's go ahead and do that. So edit your inbound rules. We'll change the rule number to 99 for our deny rule and save changes. If you select your inbound rules, you'll notice that its reordered the rule, and our deny rule is showing at the top. So now, come back to your web page. I'm going refresh. We can see the circle up here showing that it's doing something. And there we go. We've got our gateway timeout. So you can see that just a few moments after we make the changes, they will take effect. But do be aware that the numbering of the rules determines how AWS is going to interpret them. And if you have more than one rule,

the lowest-numbered rule that matches is going to be applied, even if there's a higher-numbered rule that might contradict it. So, onto my exam tips. And just remember that your VPC automatically comes with a default network ACL, which allows all inbound and outbound traffic. When you create a custom network ACL, by default, it's going to deny all inbound and outbound traffic until you go in and add your own rules. And if you would like to block a specific IP address just like we did in this demo, then you will need to use a network ACL, and you cannot do this using a security group. So if you see anything mentioned in the exam around blocking a specific IP address, and they're asking you can you do it in a security group or do you need a network ACL, then I want you to remember you can only do this using a network ACL. When it comes to subnet associations, a subnet can only be associated with one network ACL at a time, and a network ACL consists of a numbered list of rules. These rules are evaluated in order, beginning with the lowest-numbered rule first. Remember that we have separate inbound and outbound rules, and each rule can either allow or deny traffic. And then finally, remember that network ACLs, are stateless, and this means that you must configure both the inbound and outbound element of the rule, unlike security groups, which, of course, are stateful So with a security group, if you remember, you only need to configure one rule, and it's going to work for both the inbound and outbound traffic for that particular protocol. So that is it for this lesson. If you have any questions, please let me know. Otherwise, I'll see you in the next lesson. Thank you.

Demo: Connecting to an EC2 Instance Using Systems Manager Session Manager

Hello Cloud Gurus and welcome to this lesson where we'll be connecting to an EC2 instance using Systems Manager Session Manager. And we'll begin by creating an IAM role, and we're going to create this role with permissions to communicate with both Session Manager and CloudWatch. And we'll attach a couple of AWS manage policies that will give our role the permission it needs. Next, we're going to launch, an EC2 instance, and we'll associate this instance with the Identity and Access Management role that we just created. Next, we're going to configure a CloudWatch Log group, and this is going to allow, our EC2 instance to log, our session to CloudWatch, and we'll actually get keystroke logging. So whatever commands you run on the EC2 instance are going to be logged to CloudWatch, and we'll be able to go in and view those logs. And then finally, we're going connect to our EC2 instance using Session Manager. So instead of using SSH, we're going to use the Session Manager console to run a secure session to our EC2 instance. And then once we've done that, we'll run a few commands. And then we'll terminate our session and head over to CloudWatch where we'll be able to review the session logs and see the commands that we ran. So if you're ready to get started, please join me in the AWS console. So from the console, search for IAM. Select Roles on the left-hand side and Create role. Make sure that AWS service is selected. Under Common use cases, select EC2 and hit Next. At this point, we need to add our two policies, and the first one that we're going to add is the AmazonSSMManagedInstanceCore. So search for AmazonSSM, and this is the one that we're looking for, AmazonSSMManagedInstanceCore. The next one we're going to search for is the CloudWatchAgentServerPolicy. So clear your filter, search for CloudWatchAgent, and this is the one we're looking for, the CloudWatchAgentServerPolicy. Then hit Next. We give our role a name, check that we've got both of our policies attached, and this role is going to enable our EC2 instance to communicate with Session Manager, and it's also going to allow it to send logs into CloudWatch. So if you're happy with that, scroll down and Create role. So now, we are ready to create our EC2 instance. So search for EC2. Launch instance. Give your instance a name. Make sure the Amazon Linux AMI is selected. Under Instance type, we'll use the

t3.micro. When it comes to selecting an SSH key pair, we don't actually need one. So select Proceed without a key pair because we are not going to be using SSH to log into this instance. Under Network settings, select Edit. Make sure that Auto-assign public IP is set to Enable. Under the security group, I'm going to remove this rule for SSH because we don't need it. Scrolling down, select Advanced details. And then, under IAM instance profile, this is where we need to select the role we just created. So select your role and then Launch instance. So now, while our instance is launching, we can head across to CloudWatch. So select CloudWatch. On the left-hand side under Logs, select Log groups, and we're going to create a new log group. So select Create log group. Give it a name. Down here, we can provide a KMS key ARN if we would like to encrypt our log group, but we're not going to do that right now. So, go ahead and hit Create, and there is our log group. So now, I'm going to head to Systems Manager. And Systems Manager is actually a suite of lots of different capabilities that make it easier to manage your EC2 and application resources in your AWS account. And if you take a look at the menu on the left-hand side, you'll see the that there are loads of different services included, and we're going to cover a lot of these later on in the course. So, services like Run Command, which allows you to run commands on multiple EC2 instances at once. And there's also Patch Manager as well, which allows you to apply operating system patches to multiple EC2 instances simultaneously. But the one we're interested in right now is Session Manager. So select Session Manager, select Preferences, and this is where we need to enable logging. So select Edit, and it's down here under CloudWatch logging. So select Enable. Our preferred option is to stream our session logs. So this means that your session data is going to stream continuously to CloudWatch during your session. Encryption is selected by default, but we didn't encrypt our log group, so we need to deselect this option. And then down here, this is where we can select our CloudWatch log group where we want to send the session logs. And here's our log group that we created in the previous step. So select your log group. Down here, we've got the option to send our session logs to S3, and we can encrypt them as well. And there's also the option to specify environment variables or commands that we want to run when the session starts up. And you've got options for the Linux shell and Windows shell as well. So for now, just scroll down to the bottom and hit Save. And if it's all worked, you should get a message telling you that your preferences have been successfully updated. And once we've done that, we are ready to try and connect to our EC2 instance. So select Sessions, Start session. We can optionally provide a reason for our session. I'm just going to put Admin as my reason. And then down here, we can select a target instance. So select your instance and click Start session. And there we go. We've got an interactive session on our EC2 instance without using SSH. So now, I'm just going to type a few commands so that we can get some information sent into our session log. First of all, I'll run whoami, and we're actually logged in as this user called ssm-user. So let's just run a few more commands. What about pwd, or we can run ls. And I'll run ps -ef to see what processes are running on this system. And we can run df -k to take a look at our disk. And then, once you've run just a few commands, we can go ahead and exit to terminate our session. And now, if you select Session history here, is the session that we just ran. And on the reason, you will find your reason that you typed in earlier. And if you scroll all the way to the right-hand side under Output location, you will see the that the output location of our session log has been sent to CloudWatch Logs. So select CloudWatch Logs. And if it's all worked, then this is what you should see. Under timestamp, there's a timestamp for each event in our session. And if we select the drop-down for one of the events, you'll see it's recorded our session log. So we've got an event time, a region, the target, which is our EC2 instance. Here's my user identity that I'm logged into the console with. The commands I ran were run as the ssm-user. Here's my sessionId. And then under sessionData, you will find the

command that we ran and the output of the command. Let's close that down and look at the next event. And then onto sessionData, here's the next command that we ran and the output. So that is Session Manager, and I hope you can see that It's a really easy way to log into your EC2 instances without using SSH, and it's also a really quick and easy way to configure keystroke logging, send your logs into CloudWatch, and if you wanted to, you could also encrypt the logs and save them in S3 as well. So for the exam, just remember that you can use Systems Manager Session Manager for a secure remote login to your EC2 instances. It's browser-based, and it allows you to run an interactive session using either PowerShell or Bash. And we used Bash because we were working with an EC2 instance, and you can invoke Session Manager using the AWS console, the CLI, or SDK. It provides a single solution, allowing you to manage either Windows and Linux instances using the same tool, and you can also use it with your on-premises physical or virtual systems as well. And the really great thing about Session Manager is that you don't need to use the RDP or SSH protocols, and there are no SSH keys to manage. And if you remember, we didn't have to create an SSH key pair, and we didn't have to configure a security group rule either. So no SSH keys required, no bastion hosts required, and no security groups either. And then finally, it is super secure. So under the hood, it's secured using TLS encryption. Your session logs, can be sent to CloudWatch and S3, and you can also encrypt, your session logs using a KMS key. So that is it for this lesson. If you have any questions, please let me know. Otherwise, please join me in the next lesson. Thank you.

Demo: Introducing VPC Endpoints

Hello Cloud Gurus and welcome to this lesson. And in this lesson, we'll be configuring a VPC endpoint, which can be used to provide a private connection between resources in your VPC and supported AWS services. So first of all, we're going to launch an EC2 instance, and we'll attach a role allowing access to S3. Next, we'll create a VPC endpoint for S3. So this is going to enable resources within our VPC to securely access the S3 service without sending any traffic over the public internet. Next, we're going to review our route table, and we'll check that an updated route to S3 has been added into our route table. And then finally, we can test our S3 access. So we'll log into our EC2 instance and check that it can still access S3. So if you'd like to get started, I'll see you in the AWS console. And from the console, just search for EC2. Select Launch instance. Call it EC2 S3 Access. We'll use Amazon Linux. Instance type is t3.micro. We don't need a key pair. Scroll down to Advanced details. And this is where we're going to add an IAM instance profile. So first of all, click on Create new IAM profile, and it will open up in a new tab. And we need to create a new IAM role. So select Create role, make sure that AWS service is selected, scroll down to Common use cases, and select EC2. Hit Next. Under Permissions, we'll search for S3 and select AmazonS3FullAccess because we want our EC2 instance to be able to create S3 buckets. So now scroll down and hit Next. We'll give our role a name. I'll call it MyEC2S3Role. Scroll down to the bottom and Create role. Now that our role has been created, we can head back, to our EC2 launch wizard. So select the first tab. Under the IAM instance profile section, just refresh using this button, and then you should be able to Select the role we just created, MyEC2S3Role. And once you've done that, you can go ahead and Launch instance. Now while we're waiting for our EC2 instance to finish initializing, we can go ahead and create our S3 endpoint. So in the search box, type vpc, and I'm going to open it up in a new tab. And if you take a look at the left-hand screen, this is where you will find endpoints. Onto the Virtual private cloud section, so select that and Create endpoint. Now, a VPC endpoint enables you to securely connect your VPC to another service, and they've got three

different types of VPC endpoint. So first of all, you've got interface endpoints, which provide you with an elastic network interface that you can use to access supported AWS services. We've also got Gateway Load Balancer endpoints, and these are for Gateway Load Balancers, which provide load balancing for third-party virtual appliances. And then you've also got Gateway endpoints, and this is the kind of endpoint that we're going to be creating today because Gateway endpoints are the ones that are supported by S3, and they provide you with an endpoint target that we can use for all of our S3 traffic. So first of all, I'll give it a name, and I'm going to call it my-s3-endpoint. Service category is going be AWS services. And then down here under Services, we need to select the service that we want to create the endpoint for. So search for S3, and this is the one we're looking for for the S3 service, and it's a Gateway endpoint. Scrolling down, select your VPC, and it's going to be my default VPC. Under Route tables, select the one that's the main routing table. And this is the route table where the new route to S3 is going to be added, and it's going to automatically configure the route table for us. So a rule with the destination of our Gateway endpoint is going to be automatically added to our route table. And then down here, it gives us this warning, which basically says that when you use an endpoint, the source IP address of your instances in your affected subnets for accessing this AWS service, in this case it's S3. So the source IP address is going to be a private IP address and not a public IP address. And any existing connections from your affected subnets to the AWS service that use public IP addresses may be dropped. So it's basically telling us that it's going to switch from sending S3 traffic from our VPC over the internet using public IP addresses to using private IP addresses, and all the traffic will remain on the Amazon network instead of going over the public internet. And it's just warning us that if you've got any existing connections to S3 that are using the internet, then those connections are going to be dropped. Down here is our endpoint policy, and this is where you can configure the access that you want to allow. But we need to allow full access to S3 because we want to to be able to create S3 buckets. So I'm just going to stick with the default, which is full access. Then, come down to the bottom and Create endpoint. So that is our VPC endpoint for S3 created. And now, let's review our route table. So select Route tables from the left-hand menu. It's up here. So we're looking for our main routing table. Select routes. And at the top here, this is the root that's been added. And if you select the destination, I'm going to open it up in a new tab, our destination is to S3 in the us-east-1 region. So now, come back to our previous tab and select the target. I'll open it in a new tab, and the target is our VPC endpoint. So now, let's test that our EC2 instance is still able to access S3. So I'm going to come back to the EC2 console, select my instance that we launched in the beginning, select it using the checkbox here. Hit Connect, Connect, and there we are on our EC2 instance. And the first thing I'm going to do is type aws s3 ls and hit Enter. And this is going to list any buckets that are currently in our account. Now if you don't have any S3 buckets in your account at the moment, and I don't, then we can just create one by typing aws s3 mb for make bucket, then s3:// and the name of a bucket. And it needs to be a unique name, so just add some random numbers on the end and hit Enter. And now, when you type aws s3 ls, there's the name of our bucket. So, there we are. We can definitely access S3 safe in the knowledge that all of our S3 traffic from our EC2 instance is not going over the internet because we're talking to our S3 endpoint, and all traffic is staying within the Amazon network. Onto my exam tips for VPC endpoints. And just remember that when you use a VPC endpoint, you get a private connection between your VPC and supported AWS services like S3. And all network traffic will be going over the AWS PrivateLink. And AWS PrivateLink simply provides private network connectivity between your VPC and AWS services. Using a VPC endpoint is a really secure way to handle the communication between your resources in your VPC and AWS services because it means that the traffic between your VPC and the other service does not leave the

Amazon network, and it will never be exposed to the internet. And for this reason, you can use a VPC endpoint for your private subnets to enable resources in a private subnet to access resources like S3 or DynamoDB without having to use the public internet. And then finally, when you are setting up your VPC endpoint, remember the routing table because after the endpoint is created, you will need to make sure that a route to the service you want to access is present in the correct routing table. And if you remember when we created our endpoint in the console, it did this for us automatically, but it is worth checking that you've selected the correct routing table because if you haven't selected the right one, then it will not work as expected. So that is it for this lesson. If you have any questions, please let me know. Otherwise, I'll see you in the next one. Thank you.

Understanding VPC Peering

Hello Cloud Gurus, and welcome to this lecture, which is all about VPC peering. And in this lecture, we'll cover what is VPC peering, how it works, we'll take a look at an example VPC peering scenario, and we'll also cover our exam tips. So let's get started. So what is VPC peering? Well, VPC peering allows you to route traffic between 2 VPCs using private IP addresses. So it enables EC2 instances in one VPC to communicate with instances in another, and when you set up a VPC peering connection between 2 VPCs, then your instances will behave as though they were on the same private network. And for that reason, your 2 VPCs must have different IP address ranges or CIDR ranges, because you cannot have overlapping IP address ranges, just the same as you cannot have duplicate IP addresses on the same network. And one of the really cool things about VPC peering is that the VPCs don't even need to be in the same AWS account or in the same AWS Region. So, how does it work? Well, imagine we have a VPC called VPC A, and then we configure VPC peering with VPC B, and then we also want to set up peering with VPC C, and VPC D, and VPC E as well. But what about if we want VPC B to be able to communicate with VPC C? Well, in order to do that, we will need to explicitly configure VPC peering between VPC B and VPC C. And VPC C and VPC D cannot communicate with each other, and even though they can both communicate with VPC A, they cannot communicate with each other by sending traffic through VPC A. So let's take a look at an example that you might actually see in the real world. Imagine you are looking after a number of different environments, and let's say you want to separate your workloads into different VPCs. So you've got one VPC for your development environment, and let's say your development environment, it needs to communicate with another VPC used for your proof of concept environment, and it also needs to communicate with your test VPC and staging as well. So your development environment is able to communicate with all of these other VPCs using the VPC peering connection. And remember, these VPCs are communicating using private IP addresses. So they are not communicating across the internet. But what about your production environment? Well, we might want our staging environment to be able to communicate with production. Let's say, for example, we want to copy some production data to our staging environment so that we can work with some real world data. Well, we can set up VPC peering between our staging and production VPCs. And when we do that, it means that only the staging environment can communicate with production. So that means we can still keep network segregation between our production environment and the development testing, and proof of concept VPCs. So that is just one way that you might use VPC peering in the real world. So let's take a look at some of our exam tips for VPC peering. And just remember, by default, the resources in one VPC cannot communicate with the resources in another VPC. And VPC peering is what we can use to configure inter-VPC communication. So that means it enables EC2 instances in one VPC to

communicate with instances in another VPC. And remember, the traffic is going over the private network and it is using only private IP addresses. And finally, you can configure VPC peering across different AWS accounts and across different AWS Regions, so the VPCs do not need to be within the same AWS account or the same AWS Region. So that's it for this lecture. If you have any questions, please let me know. Otherwise, I'll see you in the next lecture. Thank you.

Securely Connecting to a VPC Using a Virtual Private Network (VPN)

Hello Cloud Gurus, and welcome to this lecture, which is going to cover VPNs in a little bit more detail, and we'll be focusing on site-to-site VPNs, because that is what comes up in the exam. First of all, we'll cover what is a site-to-site VPN? We'll take a look at an example use case, and then we will cover my site-to-site VPN exam tips. So what is a site-to-site VPN? Well, it's a virtual private network, or VPN, and a VPN provides a secure connection between your on-premises networks, your remote offices, and your client devices, and your AWS resources. And an AWS site-to-site VPN creates an encrypted tunnel between your network and your VPC using Ipsec, and IPsec is simply an industry standard protocol, and it's used for encrypting communications between 2 devices. So it allows you to communicate securely using the internet. And in order to configure a site-to-site VPN, you will need a virtual private gateway, and you configure this virtual private gateway in your VPC, and you also need a customer gateway device on the remote side of the VPN connection. So this customer gateway device will typically be located in your data center or in your remote office. So let's take a look at an example use case. And here we have our VPC in AWS, and here is our corporate data center. We'll add a virtual private gateway to our VPC, and we also need what AWS refers to as a customer gateway device. And the customer gateway device can be a physical device or it can also be a virtual software appliance. And VPN devices are available from companies like Cisco, Fortinet, Palo Alto, Juniper, and Netgate as well. And it's basically any security device which supports the IPsec protocol, and these gateways are used to establish a secure tunnel between your data center and your VPC, and they also perform a cryptographic processing. So all of the network packets that pass between these devices will be encrypted. So this is a really secure way to communicate between your corporate data center and resources in your AWS VPC. So on to my exam tips. Just remember, a VPN provides a secure, encrypted connection between your on-premises networks, your remote offices and your client devices, and your AWS resources. You'll need a virtual private gateway, and you configure the virtual private gateway in your VPC, and you will also need a customer gateway. And you configure the customer gateway in your on-premises network. And we use these 2 gateways to establish a secure tunnel between your data center and your VPC. They perform cryptographic processing so that all the network packets that pass between them will be encrypted. So that's it for this lesson. If you have any questions, please let me know. Otherwise I will see you in the next lecture. Thank you.

What Is Direct Connect?

Hello Cloud Gurus and welcome to this lesson, which is going to cover Direct Connect. And we'll begin with what is Direct Connect? Why is it cool? We'll take a look at an example Direct Connect architecture at a high level, and we'll also cover my exam tips as well. So what is Direct Connect? Well, imagine you have a VPC with your AWS resources over here, and here, is your corporate data center over here. And you want your servers in your corporate data center to be able to communicate with your EC2 instances and RDS databases and all of your AWS resources in your VPC, but you don't want that communication to use the internet. Well, that is where Direct Connect

comes in. So it enables communication between your data center and your VPC over your own private connection and without using the internet. So, why is it cool? Well, Direct Connect is an enterprise solution, and you get a dedicated network connection from your premises into the AWS network. It's a private connection, which does not use the internet. And it's also a fast connection designed to increase your network throughput. And you can get a dedicated Direct Connect connection in speeds of 1 Gbps, 10 Gpbs, or 100 Gpbs. So you can select the network speed that is appropriate for your organization. And the great thing about Direct Connect is that it is consistent. So you get a more consistent network experience than you would with an internet-based connection. And what I mean by that is it's not going to be fast one day, and then slow another. And with Direct Connect, you can expect to experience consistent network speeds. So if you're currently using a site-to-site VPN to connect your corporate data center to your VPC, and let's say you're experiencing some fluctuations in your network speed or maybe your finding your network is very unreliable or it's slow, then you should definitely consider Direct Connect. Now I come from an infrastructure background, and I always want to understand how everything fits together from an infrastructure perspective. So how does it all hang together? Well, let's take a look at an example setup. Here's our VPC, and here's our corporate data center, and AWS provide a number of Direct Connect locations all around the world. And this is where you can establish a link from your data center into Direct Connect. And there are loads of different locations, and they're basically colocation facilities where AWS has a presence. And let me just show you the website. It's aws.amazon.com/directconnect/locations, and this is where you will find a list of all the different Direct Connect locations that are available. And if you select the geographical location that you're operating in, it's going to show you the different locations that are available in your region. And they are basically colocation centers like Equinix, Cologix, and CoreSite. And if you've ever worked with collocation partners before, then a few of these names will probably be familiar to you. So we've got our Direct Connect location, and it's basically a physical data center where AWS is renting some space to host their Direct Connect router. And you will need to install your own customer router in the same location, and this router is going to communicate with the Direct Connect router. Next, you will need to establish a network connection between your data center and your router at the Direct Connect location. And there are various AWS delivery partners that are going to help you with this connection. And usually, it's a service provider, for example like Colt, BT, Equinix, Telstra or Zayo, all depending on which country you're in. And once again, there's a web page that you can come to to view all the different Direct Connect delivery partners. And if you scroll down, you'll see all their logos down here. And you can just select which region you're operating in, and hopefully you are going to recognize a few of these names. So, at a high level, that is how Direct Connect hangs together. And just be aware that when you are engaging third-party companies to help provision these network links from your data center to the Direct Connect location, it can sometimes take a few weeks to get this all configured, and it's not usually something that you could get done really quickly in just 1 or 2 days, for example. So just do be aware if you are planning to configure Direct Connect yourself. So onto my exam tips. And for the exam, you will just need to understand Direct Connect at a very high level. So we can use Direct Connect to provide a dedicated private connection. So it directly connects your data center to your AWS VPC without using the internet. It's great for high-throughput workloads with lots of network traffic, and you can select the network speed that you prefer depending on the requirements of your company. So you can go for 1 Gbps, 10, or 100 Gbps dedicated link. And finally, Direct Connect is designed to be secure and stable. So it provides a stable, reliable, consistent, and secure connection to your

VPC from your on-premises networks. So that is it for Direct Connect. If you have any questions, please let me know. Otherwise, I will see you in the next lesson. Thank you.

Understanding VPC Flow Logs

Hello Cloud Gurus, and welcome to this lecture, which is going to cover VPC flow logs. And we'll begin with what are VPC flow logs? When are VPC flow logs useful? What type of traffic is logged? And we'll look at a couple of example flow log records, and finally, we'll cover my exam tips. Now according to the AWS documentation, VPC flow logs are used to capture information about the traffic going to and from network interfaces in your VPC. So imagine you have a VPC. Well, you can use flow logs to capture information about network requests coming into your VPC. For example, whenever a user connects to your web server, if one of your SysOps Administrators runs an SSH session, and if one of your on-premises systems connects to your RDS database. And flow logs will also capture information about requests between one VPC and another. So basically, all network traffic flowing into and out of your VPC can be captured. And you can configure your flow log to capture the traffic that was allowed as well as any connections that were blocked. So this is going to be great for troubleshooting. Now, flow log data can be published to CloudWatch or S3. And flow logs can be created at the VPC level, at the subnet level, or at the network interface level. So you can configure flow logs wherever you need to understand the flow of network traffic within your VPC. And flow logs are great for getting a really good understanding of what is going on in your network. So they are great for diagnostics, and especially for diagnosing overly restrictive security group rules, and finding out where connections are getting blocked. They're also great for monitoring the traffic which is reaching and not reaching your instances. And they are great for understanding the direction of traffic going to and from your network interfaces. So what type of traffic can we log? Well, you can configure your flow log to record only accepted traffic, so only the connections which successfully reach their destination. Or you can configure it to record only the rejected traffic. Or we can record everything. So we can configure it to record all accepted and rejected traffic. So let's take a look at an example flow log record, and this is an example of a connection that was accepted. So here's our destination elastic network interface. This is the source IP address of the request. This is the destination IP address for the request. Here's the source port. And this is the destination port, which is port 22, so that's SSH. The number 6 here refers to the protocol, and in this case, it's TCP. This long number here beginning with 141, this is the start time, and then the number that directly follows it is the end time, so that's the start and end times of the request. And it says ACCEPT here because the traffic was allowed, so this was accepted traffic. And then right here at the end, OK simply means that this flow log record was successfully logged. So now let's take a look at a flow log record for rejected traffic. So here is our destination elastic network interface. Our source IP address. Destination IP address. Here's our source port. The destination port is 3389, which is RDP. Once again, the protocol is TCP. Here's our start and end time. And this time the traffic was rejected. But just notice, it says OK on the end here, and don't get confused thinking that this was a successful connection. The traffic was definitely rejected, because it says REJECT here, and OK just means that it was successfully logged to flow logs. So, on to my exam tips. Just remember, VPC flow logs are used to log information relating to IP traffic going to and from your VPC and the interfaces within your VPC, and we can record the accepted traffic, the rejected traffic, or all traffic. Flow logs can be stored in either CloudWatch or S3, and they are really great for troubleshooting network connectivity issues and understanding the flow of network

traffic to and from the network interfaces in your VPC. So that's it for this lecture. If you have any questions, please let me know. Otherwise I will see you in the next lecture. Thank you.

Demo: Using VPC Flow Logs

Hello, Cloud Gurus, and welcome to this lecture, which is going to be a demo and we are going to configure VPC flow logs for an EC2 network interface. So first of all, we're going to create an identity access management role because we'll need a role with permissions to send flow logs into CloudWatch. Next we'll launch an EC2 instance and we'll associate our instance with the identity access management role that we just created and we'll install HTTPD. Next we'll create a log group in CloudWatch. We'll configure our flow log from our EC2 instance and we're going to create a flow log for the elastic network interface that is attached to our EC2 instance. And then finally, we'll test that everything is working so we'll test our access to our EC2 instance using SSH and HTTP and then we can observe the traffic in our flow log. So if you'd like to get started, please join me in the AWS console. So here I am in the AWS console, and I'm going to head to Services, scroll down to Security, Identity and Compliance, and select Identity and Access Management. We'll create a new IAM role. So select Create Role. It's going to be for an EC2 instance. Hit Next, Next, review. We'll give our role a name. I'm going to call it Flow_Logs_Role and Create Role. Then search for your role up here, select your role, and we're going to add an inline policy. Select JSON. I'm going to delete what's in here. And I've already prepared a policy that we are going to use and it's located in this GitHub folder, which is attached as a resource to this lesson. So select the file named Flow Log Policy, and we'll use this policy to allow our EC2 instance to create logs in CloudWatch. So make sure you've selected everything, copy, and we're going to paste this into our JSON window. Scroll down and review. We'll give our policy a name. I'm going to call it Flow_Logs_Policy, and Create Policy. Next, we'll create a trust relationship, and this is going to allow a trusted entity to assume our role. So select Edit Trust Relationship. Let's get rid of the existing policy document, come back to our GitHub folder, and we'll go to the top-level directory, select Flow_Log-Trust_Relationship and this is the policy that we are going to add. So this allows the VPC Flow Log service to assume our role. So head back to identity and access management and paste that in here, and hit Update. So that is all our permissions set up. The next thing we're going to do is head over to CloudWatch. So come to Services and then you'll find CloudWatch under Management and Governance. So just select that, select Log Groups, and we're going to create a new log group and this is where our flow logs are going to be stored. So I'm going to name it Flow_Logs_LG, scroll down to the bottom and Create. So now we're ready to launch an EC2 instance. So head to Services, and then you'll find EC2 under compute. So select EC2, scroll down, and Launch Instance. We'll use the Amazon Linux 2 AMI. t2.micro is fine. So hit Next. Under identity and access management role, we're going to select the Flow_Logs_Role that we just created in the previous step. So select that, scroll down to the bottom to advanced details and this is where we'll add our bootstrap script to install HTTPD. So if you come back to our GitHub folder, come to the top-level folder for the lesson and select the bootstrap script. Here is our bootstrap script and it basically just does a yum update to update the operating system, installs HTTPD, creates a simple webpage just displaying Hello, Cloud Gurus. It will start HTTPD and then enable HTTPD to start at boot time. So just copy that, come back to your launch wizard and paste that in here. Then hit Next, Next, Next. We're going to stick with the default security group. Do not add anything else in here. So just hit Review and Launch and Launch. We'll create a new key pair. I'll call it my Northern Virginia key pair and download and Launch Instance. So now select your instance and it does normally just take a minute

or so to come online because remember, it's doing the yum update and it's also installing HTTPD. And if we just hit Refresh, then in a few minutes, you should be good to go. So now we can set up our flow log. And if you come to the left-hand menu, scroll down and select Network Interfaces. Here is the elastic network interface associated with our EC2 instance. So select the ENI, come to Actions, and then scroll down and Create Flow Log. We'll call it my-flow-log. We want to capture all traffic so we can see what's been accepted and what's been rejected. I'm going to change my maximum aggregation level to every one minute. Our destination will be CloudWatch logs. And then down here, we can select our log group. So select your log group. We also need to select our identity access management role. So search for flow, and you'll find your flow logs role. Scroll down and Create flow log. So that is our flow log created. So now we are ready to go ahead and test our network connectivity. So head across to your EC2 dashboard. Select Instances, select your instance ID, and we're going to grab the public IP address. And I've got 2 tests that I want to do with this EC2 instance. First of all, we're going to SSH onto our EC2 instance. And then secondly, we will try to access the web server. So first of all, from my terminal window, I'm in my downloads directory and I need to change the permissions on my key pair. So type chmod 400 and the name of your key pair and hit Enter and now we can try to SSH. So type ssh ec2-user@ and then the IP address of your EC2 instance and -i and the name of your key pair and hit Enter. Answer yes. And there we are, we've logged in. So that's our first test. And now I'm going to head back to my browser, open a new browser tab and paste in the IP address of my EC2 instance. So now we're trying to access the webpage that we installed on our EC2 instance. And if you've gone through the steps exactly the same as me, then you will not be able to access your instance, because if you remember, we didn't enable HTTP access inside within our security group. So now let's head back over to our EC2 console. So now let's head back to our AWS console and I'm going to head to services and then select CloudWatch under Management and Governance. Select Log Groups, select your Flow_Logs log group, and then down here under log streams, this is where you will find your flow log. So select the log stream, and here are all the events associated with our elastic network interface. So we can open up some of these events and take a look at the flow log entries, but right now I'm only interested in the flow log entries that relate to the IP address of my local machine. So I'm going to open a new browser tab, and I'm going to search for whatismyip.com. And this is a website that you can use to get your public IP address. And there we go. There's the public IP address of my local machine. So I'm going to copy that IP address, come back to my flow log, and I'm going to filter events based on my IP address. So hit Enter and these are all the events that are associated with my IP address, 90.206 .22 .4. So if we take a look at some of these events and if you remember, the fields in the flow log logs, here's my source IP, here's my destination. This is my port number and it's port 80. So HTTP. This is the protocol and number 6, that just means TCP. These two numbers here are a timestamp. And then here is the status and this traffic was rejected and okay just means that we successfully logged the information to CloudWatch. So let's take a look at some of these other entries. Once again, we've got port 80. Here's an entry for port 22. So this is when we connected using SSH. And here it's showing that this connection was accepted. So onto my exam tips. And just remember that you can use flow logs to understand the traffic that is coming into and out of your VPCs, subnets, and EC2 instances. And here's an example flow log record. So here's our source IP address, our destination IP address, here's our destination port, and in this example, it's HTTP, and in this case, our traffic was rejected and OK at the end simply means that the data was successfully logged to our flow log. So that's it for this lesson. And if you're working in your own AWS account, please remember to delete any EC2 instances and you can also go in and

delete your log group as well. If you have any questions about this lesson, please let me know. Otherwise, I will see you in the next lecture. Thank you.

Introducing Domain Name System (DNS)

Hello Cloud Gurus, and welcome to this lecture, which is going to introduce DNS. And DNS stands for Domain Name System, and it can also sometimes stand for Domain Name Service. Now, first of all, we'll cover what is DNS? We'll take a look at some of the top-level domains. We'll cover domain registrars, and we'll also cover my DNS exam tips. So what is DNS? Well, DNS is used to convert human-friendly domain names like http:// acloud.guru into an IP address, like 82.124 .53 .1. So DNS is kind of like the phone book of the internet, where you've got a domain name, and you want DNS to tell you the IP address that is associated with that domain name. And of course, IP addresses are used by computers to identify each other on the network, and typically, most of us are familiar with IPv4 addresses. And typically, most of us are familiar with IPv4 addresses. And the IPv4 address base has over 4 billion unique IP addresses, and way back when the internet was first conceived, they had no way of knowing how successful it was going to become. And back then when IPv4 was first developed, and we're talking the 1980s, you can imagine that they probably thought that 4 billion IP addresses was going to be enough for many years to come. But fast forward to now, and with the dramatic growth of the internet over the last 20 years, it is no surprise that we are now running out of IPv4 addresses. And of course, if we run out of IP addresses, then we will not be able to connect new devices to the internet. So that is where IPv6 comes in, and IPv6 was created to provide a larger IP address base. And this is the number of IP addresses that you get with IPv6, and it's one giant number. And the easiest way to say it is 340 undecillion addresses, and that's basically 340 with 36 zeros on the end. But the idea is that we shouldn't run out of IPv6 IP addresses for a very, very long time. So the role of DNS is to translate friendly domain names to IP addresses. So let's take a look at the kind of top level domains that are available. And when we say top level domain, what we mean is the last word in a domain name is the top level domain. So with google.com, the top level domain is com. With a domain name like a cloud.guru, the top level domain is guru. And what about a domain name like this bbc.co .uk? Well, in this case, uk is the top level domain and co is the second level domain. Now these top level domain names are controlled by the Internet Assigned Numbers Authority, and they are stored in a root zone database, which is just a database of all the top level domains. And the Internet Assigned Numbers Authority, this is just an organization which manages and coordinates the global DNS top level domains, including domains like com, guru, uk, edu, gov, and org, etc., and you can actually view this database online by visiting www.iana .org /domains/root/db. And here it is. So this is where you can find all of the top level domains that exist. Now, as you are probably aware, all domain names must be unique. Duplicates are not allowed, and there is only one acloud.guru. Domain names are registered with a domain registrar, which is an authority who is allowed to assign domain names directly under one or more of the top level domains. And each domain name, including contact and registration information, is registered in a central database and it's known as the WHOIS database. And there are loads of different domain registrars out there, and popular ones that you may have heard of are companies like domain.com, Go Daddy, domains.google, eNom, and AWS, and you can even purchase your own domain name from inside the AWS console, and they've actually made it really easy for you to do that. So on to my exam tips, and don't worry, you are not going to be tested on how global DNS works. This lesson is really just to give you some background which is going to help you when we start using Route 53, which is the AWS DNS service. So just remember that DNS

is used to convert human-friendly domain names like http:// acloud.guru into an IP address like 82.124 .53 .1. The last word in a domain name is the top level domain. For example, com, edu, guru, gov, and org, and domain names are unique, and they are all registered using a domain registrar. For example, a company like GoDaddy, eNom, or even AWS. So that's it for this lecture. If you have any questions, please let me know. Otherwise, I will see you in the next lecture. Thank you.

Introducing Route 53

Hello Cloud Gurus, and welcome to this lecture, which is going to introduce Route 53 or Route 53, however you like to say it. We'll start off with what is Route 53? We'll take a look at Route 53 hosted zones, common DNS records, Route 53 alias records, we'll compare alias records with CNAME records, we'll take a look at TTL or Time to Live, and then we'll go through my Route 53 exam tips. So what is Route 53? Well, Route 53 is Amazon's DNS service, and it allows you to register and purchase your own domain name, and it allows you to map a domain name that you own. For example, ilovecloud.com to EC2 instances, elastic load balancers, and S3 buckets. Now, when you first register a domain name using Route 53, a hosted zone will be created for your domain name. And a hosted zone is simply a container for DNS records in Route 53, and by default, a hosted zone will always include an SOA record and an NS record. Now an SOA record stands for 'start of authority', and the SOA record provides important basic information about the domain, and the NS record is the name server record, and this defines the Route 53 name servers that are authoritative for your hosted zone. And when you first create a hosted zone, AWS will configure all of this for you. So let's take a closer look at these 2 record types in more detail, along with some of the other common DNS record types that are available with Route 53. So within our hosted zone, we've got our SOA record, or start or authority, and this contains important domain information. And the SOA record provides important information about your domain. For example, the Route 53 name server which created the SOA record, contact details, so the email address of the administrator responsible for the domain, and important settings, so things like refresh, and retry, and Time to Live settings. Within our hosted zone we also have the name server record, and this is used to identify the DNS server names that are going to be authoritative for your domain, and these are the name servers which are going to resolve your domain names. For example, these are the servers that will resolve ilovecloud.com. And the way it works is, let's say we are trying to connect to a website, so we type the name of the website into our browser, and when we do this, we are basically asking our internet service provider to translate this domain name to the correct IP address and take us to the website. Now, if our ISP doesn't know the IP address, for example if no one else has requested this website recently and the IP address is not held in the cache, then your ISP will need to go out and find it from somewhere else. So it forwards the request to the top level domain, and in this case, the top level domain is com. So it's going to ask the top level domain for the NS record for ilovecloud.com, and the top level domain is going to find the NS record for us. And remember, the NS record is not going to give us the IP address that we're looking for, but it is going to tell us who is authoritative for ilovecloud.com, and if AWS Route 53 is authoritative for our domain name, then the top level domain will respond with a name server record like this. And this is basically the Route 53 name server which is going to perform the host name resolution. So the request is then sent to the Route 53 name server, and Route 53 will respond with the IP address. Now you might be wondering, where does the name server find this IP address? And the answer is the A Record. So what is an A Record? Well, the clue is in the name, and the A in A record stands for address, and it's

the A record which is used to translate the website domain name to an IPv4 address. For example, www.acloud .guru might resolve to an IP address like 123.10 .10 .80. Now another common type of DNS record that you might be aware of is CNAME, and the C in CNAME stands for canonical. And in this context, canonical means according to the rules. So what is a CNAME record used for? Well, it's used to resolve one domain name to another. For example, you may have a mobile website with the domain name m.acloud .guru, which is used when users browse to your website using their mobile devices. And let's say you also want the domain name mobile.acloud .guru to resolve to the same address. Well, we can do this using a CNAME record. But there is one limitation with CNAME records, and you cannot create a CNAME record for the zone apex, and the zone apex, or domain apex, is basically the root level of your domain name. For example, acloud.guru. acloud.guru is also an apex. Instead, it must always be a sub-domain, like mobile.acloud .guru. So what about if you do want to do this? Well, that is where alias records come in, and alias records are AWS specific, and you can use an alias record to map DNS records in your hosted zone to AWS resources like Elastic Load Balancers, CloudFront distributions, S3 buckets, or any other record in the same Route 53 hosted zone. And if you think this sounds fairly similar to a CNAME record, then you'll be right, because an alias record allows you to map one DNS name to another. However, the great thing about alias records is that you can create an alias record for your zone apex or your domain apex. So that means you can map www.example .com to a target DNS name like the name of your Elastic Load Balancer, and this is not possible using a CNAME record. And alias records are the AWS preferred option, so if you do get a choice, and if you do see a choice in the exam, if possible, you should go for an alias record over a CNAME record, especially if you need to create an alias record for your zone apex. Now for the exam, it is really important to understand the differences between aliases and CNAMEs. So just remember, you can create an alias record with the same name as your hosted zone, also known as the zone apex or the domain apex, for example acloud.guru, and also, Route 53 does not charge for alias queries to AWS resources. Whereas with a CNAME, you can't create a CNAME record with the same name as your hosted zone, also known as the zone apex or the domain apex, and Route 53 does charge for all CNAME queries. So given the choice, always go for alias over CNAME. Moving on to TTL, and TTL stands for Time to Live, and this is the amount of time, in seconds, that you want the DNS resolvers to cache information about a particular Route 53 DNS record. And the lower the Time to Live, for example, if we set our Time to Live for 60 seconds, the faster changes to DNS records will propagate throughout the internet. And with a longer TTL, this will reduce the number of calls that the DNS resolvers need to make to Route 53, which can of course reduce your bill for the Route 53 service. However, just be aware that with a longer Time to Live, this means it will take longer for any changes that you make to propagate around the internet. So if you're making a lot of changes, and you want the changes to propagate quickly, then you should select a shorter Time to Live so that the information stays in the cache for a shorter time. But just be aware that that will increase the number of calls made to Route 53, which could increase your bill. Moving on to Route 53 exam tips. And the main thing that you will need to be aware of is the different DNS record types. So just remember the start of authority record provides important basic information about the domain, including the email address of the domain administrator. We then have name server records, or NS records, and these are the DNS server names that will be authoritative for your domain. And it's these name servers which are going to resolve your domain names. We've then got A records, or address records, and these are used to map a website name, like acloud.guru, to an IPv4 address. And there's also alias records and CNAME records. And the main thing to remember is that an alias record allows you to map one DNS name to another. For example, if you wanted to map www.example .com to the DNS name of

your Elastic Load Balancer. And you can also create an alias record that has the same name as the hosted zone also known as the zone apex or the domain apex. Whereas when it comes to CNAMEs these are used to resolve one domain name to another, for example, if you want m.acloud .guru and mobile.acloud .guru to resolve to the same address, you can use a CNAME to set that up. However, remember the limitation with CNAMEs, which is that you cannot create a CNAME record which has the same name as your hosted zone or the zone apex. So that's it for this lecture. If you have any questions, please let me know. Otherwise I will see you in the next lecture. Thank you.

Exploring Route 53 Resolver

Hello Cloud Gurus, and welcome to this lecture, where we're going to take a look at Route 53 Resolver. First of all, we'll look at what is Route 53 Resolver, we'll take a look at resolving domain names outside your VPC, integrating with other DNS resolvers, and we will also cover my exam tips. So what is Route 53 Resolver? Well it resolves DNS queries coming from resources in your VPC. For example EC2 instances, Elastic Load Balancers, and DNS records in private hosted zones within your VPC. So here's our VPC. And for DNS queries within the VPC, the query comes in to Route 53 Resolver, and Route 53 Resolver will resolve the query to the appropriate destination, for example, an Elastic Load Balancer or an EC2 instance. So it's resolving based on a domain name like this, and translating the domain name to the appropriate destination. So it's translating the domain name to the correct IP address. But what about resolving domain names outside your VPC? Let's say the DNS query is for a destination outside the VPC. Route 53 Resolver is able to query a public DNS server to get the correct IP address for the domain name that it is trying to resolve. For example, an address like acloud.guru, which is outside your VPC. So Route 53 Resolver will perform what is known as a recursive lookup against the public name server. So that basically means it is going out to the public name server to ask the public name server to resolve that domain name. And Route 53 Resolver can also integrate with other DNS resolvers. For example, another VPC that you have a peering connection with, or even a DNS server located in your own data center, accessed using a VPN. So you can integrate Route 53 Resolver with another DNS resolver on any network that is reachable from your VPC. For example, another peered VPC or an on-premises network. So let's take a look at my exam tips for Route 53 Resolver. Just remember, it's used to resolve DNS queries for resources within your VPC. For example, EC2 instances, Elastic Load Balancers, and DNS records in hosted zones within your VPC. And it can also deal with lookups for resources outside your VPC. So for DNS queries to other domain names outside your VPC, Route 53 Resolver will perform a recursive lookup against a public name server to resolve that external domain name. And finally, you can integrate Route 53 Resolver with any other DNS resolver on any network which is reachable from your VPC, and that could be another peered VPC or a DNS server in your on-premises network. So that's it for this lecture. If you have any questions, please let me know. Otherwise, I will see you in the next lecture. Thank you.

Understanding Route 53 Routing Policies

Hello, Cloud Gurus, and welcome to this lecture, which is going to cover the different Route 53 routing policies that are available. And we'll start off discussing what is a routing policy, and then we'll go through each routing policy in turn, beginning with the simple routing policy, failover routing, geolocation, geoproximity, latency-based routing, multivalue answer, the weighted routing policy, and we'll also cover my exam tips. So, what is a routing policy? Well, when you create a DNS record in Route 53, you select a routing policy and it's this routing policy, which determines

how Route 53 will respond to DNS queries. And the most basic routing policy there is, is the simple routing policy. And this is where you are routing to a single resource. For example, a single web server hosting your website. Next, we have the failover routing policy and this is the one to use when you want to configure active-passive, failover for your website. So, we have our primary active server and if something goes wrong with our active server then Route 53 is going to start resolving all of our requests to the remaining server, which will then become the active server. Next, we have the geolocation routing policy, and this allows you to route traffic based on the location of your users. So, let's say we have a user based in the USA, and we have web servers located in us-east-1, and eu-west-1 as well. With geolocation-based routing, all of our users requests will be sent to our web server in us-east-1. There's also geoproximity routing, and this allows you to route traffic based on the physical distance between your users and your AWS resources. So, let's say you have a user and they're located in France and you've got web servers in eu-central-1, which is Germany, and eu-west-2, which is London. With geoproximity routing, Route 53 can route traffic to the nearest location of our users. So, if our user in France is closer to London, then their request will be sent to eu-west-2. And if they're in a part of France, which is closer to Germany, then their request will be sent to eu-central-1. And with geoproximity routing, you can also configure a bias to route more or less traffic to a given resource. For example, you can route 90% of your traffic to eu-west-2, and the other 10% to eu-central. And this is great if you don't have an even distribution of resources. Let's say for example, you've got 20 web servers in eu-west-2, but you've only got 5 in eu-central-1. You might want to configure a bias to route most of your traffic to eu-west-2. On to latency-based routing, and this allows you to route your traffic based on the best latency. So, basically, the lowest latency, and this is great if you want to provide the best performance for your users. So, let's say we've got a user and they are based in the USA and you have web servers configured in us-east-1, us-east-2, and us-west-1. With latency-based routing, Route 53 is going to route traffic based on whichever of these regions is providing the best latency, or the best performance for your user. We also have multivalue answer routing, and this is where Route 53 responds with up to 8 healthy records selected at random. So, when a user browses to your website, Route 53 will return up to 8 healthy records, and each of those records will be selected at random. And then, finally, we have the weighted routing policy, and this allows you to route traffic to multiple resources based on a numerical weight that you define, and the weight can be anything between zero and 255 with the total weight adding up to 255. For example, imagine you have a large fleet of EC2 instances located in eu-west-1, and then you've got the same again in us-east-1, you can configure a weighted routing policy to send a majority of user requests to these 2 regions where you have the greatest capacity. And then, you can send a smaller proportion of requests to the third region, eu-central, where you have fewer web servers and less capacity. So, that is a quick run through of all the different routing policies available. Let's take a look at some of my exam tips. Now, for the exam, you only need to understand each of these routing policies at a very high level, but you will need to be able to differentiate between each one and understand the different use cases. So, just remember, with the simple routing policy, that is the most basic one that you can get, and this allows you to route traffic to a single resource. For example, a single web server running your website. With the failover routing policy, this allows you to configure an active-passive failover. Geolocation allows you to route traffic based on the location of your users. Geoproximity is based on the physical distance between your users and your resources. So, just think the proximity of your users and it also allows you to configure a bias to route more or less traffic to a given resource. Latency-based routing allows you to route traffic based on the best latency to provide good performance for your users. The multivalue answer routing policy is the one to use if

you want Route 53 to respond with up to 8 healthy records selected at random. And then, finally, we have the weighted routing policy, and this one allows you to route traffic to multiple resources based on the numerical weight that you define. For example, routing the majority of your traffic to a region where you have greater resources. So, that's it for this lecture. If you have any questions, please let me know. Otherwise, I will see you in the next lecture. Thank you.

Demo: Route 53 Simple Routing Policy

Hello Cloud Gurus and welcome to this lesson. And in the this lesson, we're going to configure a Route 53 record with a simple routing policy. We'll begin by launching an EC2 instance, and we're going to install httpd on our instance. Next, we'll take a look at the Route 53 hosted zone that is provided in the cloud playground, and we'll create an A record with a simple routing policy, and that's going to allow us to map a friendly domain name to our web server. And then finally, we're going to test that everything is working by attempting to access our website using our own domain name instead of the IP address of our web server. Now, within the A Cloud Guru Sandbox in the cloud playground, we have provided a Route 53 hosted zone for you to use, and it is already configured with the DNS registered domain name that we own. So this is the hosted zone that you're going to use for this demo. So if you'd like to join me in the AWS console, we'll get started. And the first thing that we're going to do is search for EC2 and Launch instance. Instance name is My Web Server. We'll use Amazon Linux. Instance type is going be t3.micro. We're going to proceed without a key pair. Edit your network details. And under the security group, we can name our security group Web DMZ. And I'm going to update the security group rule because we don't need SSH, but we do need HTTP on port 80. And now scroll down until you get to Advanced details and open that up. Scroll down to the bottom until you get to user data. There it is. And this is where we're going to add a Bootstrap script, and you will find a link to this Bootstrap script in the resources for this lesson. Here's our script. I'm just going to copy it and paste it into our User data section. So this is just a really simple script. It's going to start off by telling the operating system to use the Bash interpreter. Yum up update -y is going to update the operating system. We're going to install httpd because it's going to be a web server, create a simple web page that just says Hello Cloud Gurus, and the web page will be saved to this location. And then finally, we'll start our web server using the systemctl start httpd command. And then this last command is going to configure the web server to start every time the instance boots. So once you've added this script, you can select Launch instance. And while it's launching, let's search for Route 53 and open it up in a new tab. If you see an error message like this, don't worry about it. It's not going to impact anything that we're doing. You can just close it down. And now head to Hosted zones, and you should see something similar to this because we've already purchased it and registered a domain name that you can use. And here is our hosted zone name. So once you've identified your hosted zone in here, we are ready to continue. So select your hosted zone. And then down here, in the hosted zone, you can see it's already created our start of authority record, our SOA record, and it's created name server records here as well. And the next thing we need to do is create our A record for our website. So select Create record. This is where you can create a record name. And if we just want to route traffic to the name of our domain, we can leave this section blank. And it says here keep blank to create a record for the root domain, and this is our root domain. But if you wanted to create a subdomain, for instance, you could put it in here. So you could put in a subdomain, for instance, like sales. But we're just going to keep this blank. The record type is going to be an A record. But if you select this drop-down, you'll see all the different types of DNS records that are supported by Route 53. So

there's CNAME, which allows you to route traffic to another domain name and some AWS resources as well. There's MX record, which is used for mail servers. Down here is the NS record, which defines the name servers for your hosted zone. And those are the main record types. Along with A record, those are the main ones to be familiar with. And for this demo, we are going to be creating an A record. Now if you remember, an a record allows you to route traffic to a specific IPv4 address, and the IPv4 address that we're going to use is the public IP address of our web server. So let's come back to our EC2 instance tab, select our instance ID. There's our instance. If you click on the instance ID, we're going to grab the public IPv4 address, come back to Route 53, and we need to put the IP address into this box. So paste that in the there. Then scrolling down, down here, we can define a time to live in seconds. And this time to live just defines the amount of time in seconds that you want the DNS resolvers to cache information about this record. And we're going to just stick to the default of 300 seconds. And then finally, this is where we can configure our routing policy. And if you select the drop-down, you'll see all the different types of routing policy that are supported. So we've got weighted, geolocation, latency-based, failover, multivalue answer, and IP-based as well. So if you're happy with that, just select Create record. Here's our record. And the message up here just lets us know that Route 53 propagates your changes to all of the Route 53 authoritative DNS servers within 60 seconds. And we can use this View status button to check the propagation status. And the status is showing as INSYNC. So now, come back to your hosted zone, and now we are ready to test if everything has worked. So copy the name of your A record. Here it is. And then in a new browser tab, paste in your A record and hit Enter. And if everything has worked, this is what it should look like. So it's taken me to the website that is running on my EC2 instance. So for my exam tips on configuring Route 53 with a simple routing policy. Firstly, we created an A record in our hosted zone. And remember, an A record allows us to route traffic based on an IPv4 address. We selected the simple routing policy to route traffic to the IPv4 public IP address of our EC2 instance. And then we tested everything by accessing our website using our domain name instead of the IP address. So that is it for this lesson. Any questions, please let me know. Otherwise, I will see you in the next one. Thank you.

Demo: Route 53 and Weighted Routing Policy

Hello Cloud Gurus and welcome to this lesson where we'll be configuring a Route 53 weighted routing policy. We'll start off by launching two EC2 instances. Next, we'll install httpd. We'll then configure a routing policy, so we'll create an A record using a weighted routing policy. And we're going to be sending the majority of the traffic for our domain to our first web server, WS1. So we're going to set our weight to 200. And then the remaining traffic is going to be sent to our second web server, WS2, and we're going to set that weight to 55. Then finally, we'll test everything. So we'll test that it all working correctly and that our requests are being routed as expected. And the cool thing about Route 53 is that these servers could even be in two different regions. They don't both need to be in the same region. It would still be supported. However, for this demo, we're going to be doing everything in us-east-1. So if you'd like to join me in the AWS console, we'll get started. And from the console, first of all, search for EC2 and Launch instance. Call it WS1. We'll use Amazon Linux. Instance type is t3.micro. We can proceed without a key pair. Under Network details, we need to edit. And we're going to change our security group name. Just call it Web DMZ. Under Security group rules, we need to change this rule because we don't need SSH. And instead, we want HTTP on port 80 with a source type of Anywhere. Then scroll down to the bottom to Advanced details. Open the drop-down. Scroll down right to the bottom again until you get to the User data

section, and here it is. We're going to add a Bootstrap script in here, and there's a link to this script in the resources for this lesson. I'm just going to copy everything and paste it into the User data section. It's just a really simple script, which is going to start off by telling the operating system to use the Bash interpreter. Yum update -y is going to update the operating system. Next, we're installing httpd, creating a really simple web page that just says Hello Cloud Gurus, and we're calling our web page index.html. Then, we're starting httpd and enabling it to start every time the system boots. So once you've done that, select Launch instance. Once that's initiated, we're going launch our second instance. So come back to EC2, Launch instance. This time, we'll call it WS2. It's going be Amazon Linux, t3.micro. We don't need a key pair. Under Network settings, select Edit. And under Security group, we'll select the one we just created, Web DMZ. Next, come to Advanced details and scroll down to User data and paste in your Bootstrap script. Then, Launch instance. And now if we select Instances, there's our two instances, and they are still initializing. So while that's happening, let's search for Route 53. I'll open it in a new browser tab. If you see a message like this, you can just ignore it. It's not going to affect what we're doing. So close that down and select Hosted zones from the left-hand menu. Here's my hosted zone. I'm just going to open it up, and you should have the name server record and the start of authority record. And if there are any A records lurking in here from previous lessons, then you can just go ahead and remove them. And to do that, you just select the record. And then, you should be able to delete it if it's an A record. So you want your hosted zone to look similar to this just with the name server record and the start of authority record as well. So now, let's go in and create a weighted routing policy. So select Create record. We'll keep the record name blank because it's going be for our root domain, and that's our root domain. Record type is going be an A record. It's not going to be an alias. The value is going to be the public IP address of our web server 1. So if you come back to Instances, select web server 1, select the public IP and copy that, and paste that into here. Time to live is 300. Routing policy is going be weighted. And this is where we can set our weight. So for web server 1, the weight is going to be 200, so type 200. Under record ID, I'm going to give it a record ID of 1. Health check is optional, so I'm going to leave that as blank. And we need to add another record. So select Add another record. Most of the options are going to be the same. So it's a record type of A record. It's not going to be an alias. This time, the value is going to be the public IP of web server 2. So come back to our instances, select web server 2, grab the public IP. Back in Route 53, paste in the IP address. Time to live is 300. Routing policy is weighted. And this time, the weight is going to be 55. We'll leave the health check ID as blank, and the record ID is going be 2. So now, we're ready to create our records. And this message is just reminding us that Route 53 propagates your changes to all of the Route 53 authoritative DNS servers within 60 seconds. And you can use the View status button to check on the propagation status. So if we select the View status button over here, we can see the status is its INSYNC. So now come back to your hosted zone page, and there's our new A records. So that is our weighted routing policy configured. And now, let's just test it by selecting our domain name. Copy that, paste into a new browser tab, and hit Enter. And there we go. It has hit our EC2 instance, and we should be sending most of our traffic to web server 1. And then, the minority of the traffic will be going to web server 2. So, onto my exam tips. And just remember that we can use a Route 53 weighted routing policy to route traffic for the same domain name to multiple destinations. And the destinations could be in the same region or even in a completely different region depending on your use case. We use weights to define how much traffic is going to be routed to each destination, and the weight is a numerical value between 0 and 255. So that is it for this lesson. Any questions, please let me know. Otherwise, I will see you in the next one. Thank you.

Demo: Route 53 Latency Routing Policy

Hello Cloud Gurus, and welcome to this lecture, which is going to be a demo, and we'll be configuring a Route 53 latency-based routing policy. So first of all, we're going to launch 2 EC2 instances, and I'm going to launch them in different regions, 1 in us-east-1 and 1 in eu-west-1. Now, if you're working in our AWS sandbox, you will only have access to launch instances in us-east-1, so you can just launch both of yours in us-east-1. We'll install httpd on each of our instances, and then we are going to configure our routing policy. So we'll create an A record using a latency-based routing policy, and we'll test that everything is working and that requests are being routed as expected. Now I'm based in the UK, of course. So I am expecting Route 53 to route my request to my server in eu-west-1. But depending on your location, if the latency between your location and us-east-1 is less, so it's based on the least latency, then your request will be routed to us-east-1 instead. So if you'd like to join me in the console, we'll get started. So here I am in the console, I'm going to head to Services, come to Compute, and select EC2. I'll launch my first instance in us-east-1. So scroll down and Launch Instance. Select Amazon Linux 2, Select Next, Scroll down to Advanced Details, and over here is my bootstrap script, and you'll find this in the Resources section of the course. I'm going to add my bootstrap script. I'm going to modify the message on the website (keyboard clicks) to say, "This is web server 1 - us-east-1". Hit Next, Next. We can add a tag if we want. Next. We'll add another rule for our security group to enable HTTP. Review and Launch, Launch. If you have an existing key pair in that region, then you can select it, or you can create a new one, and Launch Instance. Next, I'll launch my second instance, and I'm going to launch it in eu-west-1. So head back to Services, EC2, change my region to eu-west-1, which is Ireland, scroll down, and Launch Instance. Select the Amazon Linux 2 AMI. t2.micro is fine, so hit Next. We'll scroll down and paste in the same bootstrap script. For the website message, I'm just going to update that to, "This is "Web Server 2" and it's located in eu-west-1. Hit Next. Can add a tag if you want. Next. We'll add our security group rule for HTTP, Review, Launch. I'll create a new key pair. I'm going to call it irkp, Download my key pair, and Launch Instance. So now, let's head over to the Route 53 Console. I'm going to select Services, scroll down to Networking & Content Delivery. I'm going to select Route 53 and open it in a new tab. Open up the menu on the left, and we'll come to our hosted zone. Select your domain name, and if you have any A records from the previous lecture, then we're just going to delete these records. So this is what your hosted zone should look like. You should just have your NS record with your name servers and the start of authority record as well. So select Create Record, the record type is going to be an A record., the routing policy is going to be Latency, select the region, and it's going to be us-east-1. The value is going to be the IP address of our EC2 instance in us-east-1. So let's come back to EC2, make sure you're in the us-east-1 region, let's find our web server 1, and select the public IP address, come back to our Route 53 console, and paste that in here, I'll create a record ID of 1, and Create Record. So now, we need to create our second record. So select Create Record, the record type is going to be an A record, the policy will be Latency, the region is eu-west-1, and we'll need to grab the IP address of our web server 2, so let's come back to EC2, change region to eu-west-1, select web server 2, and copy the public IP address, come back to the Route 53 console, and paste that in here, we're going to give this a record ID of 2, and Create Records. So that is our 2 A records configured using the latency routing policy. So now, let's test that everything's working. So I'm going to copy my domain name, open a new browser tab, and paste, and there we go. It sent my request to eu-west-1, and hopefully if you are working in a location that is closer to the US, or if you're in the US, your request will have been sent to web server 2, because that is the web server that is going to give

you the least latency. So on to my exam tips. Well, just remember that a latency routing policy is great to use when you have resources located in multiple regions and you would like to send traffic to the region that is going to ensure the best latency and performance for your users. So that's it for this lecture. If you are working in your own AWS account, please remember to delete your EC2 instances once you've finished. If you have any questions, please let me know. Otherwise, I will see you in the next lecture. Thank you.

Demo: Route 53 Failover Routing Policy

Hello Cloud Gurus and welcome to this lesson, which is going to be a demo. And, well, in this demo, we are going to create a Route 53 failover routing policy. First of all, we'll launch two EC2 instances, and we'll configure them as web servers. So we'll install httpd. Next, we'll configure a Route 53 health check, and this will enable Route 53 to automatically check that the website is up and running on our primary instance. So we're going to have an active and passive or primary and secondary setup for our two EC2 instances. And then we're going to configure failover-based routing. So we'll create an A record using a failover routing policy. And then, we're going to stop our active instance and test that everything is working. And if it's working as expected, then Route 53 will begin sending requests to our remaining instance. So if you'd like to join me in the AWS console, we'll get started. So from the console, first of all, search for EC2 and Launch instance. Call it WS1. We'll use Amazon Linux. Instance type will be t3.micro. We can proceed without a key pair. Scrolling down to Network settings, select Edit. And under our security group, we'll change the security group name. And I'm going to change it to Web DMZ. Next, we'll change our security group rule to allow HTTP instead of SSH because we don't need SSH. Port is 80, and source type is Anywhere. Then scroll down to Advanced details and open the drop-down. Scroll right down to the bottom until you get to the User data section. And we're going to add a Bootstrap script in here. And you will find a link to the Bootstrap script in the resources for this lesson. So I'm just going to copy everything, come back to my EC2 launch screen, and paste that in here. And this is a really simple script, which is going to start off telling the operating system to use the Bash interpreter. Then yum update -y is going to update the operating system. We'll install httpd and create a simple web page. Then we'll start httpd and enable httpd to start every time the system boots. But I'm just going to change one thing in the this script. I just want it to display, the name of my web server on the HTML page. So after Hello Cloud Gurus, just type This is web server 1 and Launch instance. Next, I'm going to launch my second web server. So come back to Instances, select Launch instance, call it WS2. It's going be Amazon Linux. Instance type is t3.micro. We're going to proceed without a key pair. Under the network settings, select Edit. And we're going to select our existing security group. And there it is, Web DMZ. Next, scroll down to Advanced details and then scroll down again to the bottom to the User data section and paste in your Bootstrap script. And once again, we're going to update our script. This time, I'm going to write This is web server 2 and Launch instance. Then at the top of the screen, select Instances. And there is our two instances. And we'll just wait a few moments for them to finish initializing. And then once your EC2 instances are both up and running, let's just check that our web servers have installed successfully. So first of all, we're going to check web server 1. So select the instance ID, copy the public IP address, open a new browser tab, and paste in the IP. Hit Enter and there we go. This is web server 1. And now, we'll go back and do the same for web server 2. So select the instance ID. Copy the public IP address. We'll open a new browser tab, paste in the IP, and there we go. This is web server 2. I'm going to close those browser tabs down so that they don't confuse me later on. So now, let's search for Route 53. I'm going to

open it in a new tab. We can ignore this message. I'm just going to close it down because it's not going to impact us. Select Hosted zones from the menu on the left, open up our hosted zone, and you should have a name server record and a start of authority record. And if there are any A records in here from previous lessons, then just make sure you go ahead and remove them. And you can just select the record and select Remove. But I only want you to remove A records. Don't remove your name server record or your start of authority. So if it's all looking something like this, we are ready to continue. So now, let's create our health check. Select Health checks from the left-hand menu and select Create health check. Now this health check is going to be used to determine whether our web server is healthy or not. So first of all, we'll give it a name. Call it Web Server 1 Health Check. We're going to monitor our endpoint, and the endpoint will be specified by using an IP address. Protocol is HTTP, and the IP address is going to be the public IP of web server 1. So head back to your EC2 instances, make sure you've selected web server 1, and grab the public IP address. Back in Route 53, we'll provide the public IP over here. Hostname is going to be the name of our hosted zone, so it's going be our domain name. To find that, we'll head to Route 53. I'm going to open it up in a new tab, and here's my hosted zone. So I'm just going to copy the name of my hosted zone and paste it in here. It's going to use port 80, and the path is just going be index.html. So this health check is basically going check whether or not Route 53 can successfully access index.html on this web server. Then under Advanced configuration, this is where you can set the request interval and a failure threshold as well. And I'm going to change the request interval to Fast so that it's checking every 10 seconds. And then the failure threshold just defines the number of consecutive health checks that an endpoint needs to fail in order for Route 53 to change the status from healthy to unhealthy. So I'm going to change that to 1 because I want things to happen pretty quickly. So now, just scroll down to the bottom. We can accept the rest of The defaults. Hit Next. Over here, we can configure CloudWatch to send an SNS notification, for instance to send us an email, if the health check changes to unhealthy. But I'm not going to do that. I'm just going to go ahead and Create health check. So that is our first health check created for web server 1, and we need to do the same thing for web server 2. So select Create health check. Call it web server 2 health check. We're monitoring an endpoint. We'll specify our endpoint using an IP address. Protocol is HTTP, and we're going to use the IP address of our web server 2. So head back to Instances, deselect web server 1 and select web server 2, grab the public IP address. Then back in Route 53, provide the public IP address. Hostname is going be our domain name from our hosted zone, so paste that in here. Path is index.html. Then under Advanced configuration, set the request interval to 10 seconds and the failure threshold to 1 just to make everything happen quickly. Then go ahead and hit Next and Create health check. Now it does just take a few minutes to configure everything and get both of these health checks into a healthy state. So now is a good time to take a break for a few minutes, step away from the screen, have a nice cup of tea, and then come back in about 5 minutes. And then we should be ready to continue. After a few minutes, we can refresh, and everything should be showing as healthy. So now, we are ready to create our Route 53 A record with failover routing. So select Hosted zones. Select your hosted zone. Within your hosted zone at the moment, you should only see your name server record and your start of authority record and select Create record. We'll keep the record name to blank because we're creating a record for our root domain, which is this. Record type is going be an A record. And then the value that we add in here is going to be the public IP address of our web server 1, which is going to be our primary web server. So back in the EC2 management console, select your web server 1 and copy the public IP. Back in Route 53, paste in the public IP. Time to live is currently 300 seconds, but I want to change it to 10 seconds because I want everything to happen quickly. Routing policy is going be Failover routing policy. And failover

record type is going be Primary because this is our primary or active web server. Under Health check ID, select our web server 1 health check, and record ID is going be 1. Now before we go ahead and create our records, we need to add another record. So this is going to be the second part of our failover routing. So add another record. We'll keep the record domain name as blank. Record type is A. Under Value, this will be the public IP address of our web server 2. So from the EC2 management console, deselect your web server 1 and select web server 2. Grab the public IP and paste it into here. Time to live is going be 10 seconds. Routing policy is Failover. Failover record type is Secondary. Health check is our web server 2 health check. And record ID will be 2. And once you've done that, you can select Create records. And it's just letting us know that it's going to propagate the changes within about 60 seconds. So now we should be able to access our website using this domain name. So I'm going to copy my domain name, open a new browser tab, and paste it in the there. And if it all works successfully, this is what you should see. So it should take you to web server 1. And now, it's time to test whether our failover routing policy has been configured correctly. And the way that we can do that is by stopping this web server. So we'll go in and stop our web server 1 instance. And in the theory, Route 53 is going to fail over its routing and route our requests to the secondary web server. So head back to the EC2 console, deselect web server 2, and Make sure you've got web server one selected. Then select Instance state and Stop instance. So now, my instance state has changed to Stopping. Now let's head back to Route 53. I'm going to select the health check on the left-hand menu. And it can take Route 53 a few minutes to notice that our EC2 instance is down. So if we just be patient, wait a few minutes, and use this button to refresh, and there we go. After a few minutes, our web server 1 health check has changed status to Unhealthy. And if we select our web server 1 and select Monitoring, scroll down, we can see that something has gone badly wrong here. So now, let's come back to our website. I'm going to refresh. And there we go. If it's all worked correctly, you should have been redirected to web server 2. So finally, let's take a look at my exam tips. And just remember that when you're configuring Route 53 failover routing, first of all, you need to launch your instances and create a Route 53 health check for each instance. Next, we created an A record with a failover routing policy. And then finally, we were able to test our setup by stopping our primary active instance, and then Route 53 began to send requests to our remaining instance, the web server 2. So that is it for this lesson. Any questions, please let me know. Otherwise, I will see you in the next lesson. Thank you.

Demo: Route 53 Geolocation Routing Policy

Hello Cloud Gurus and welcome to this lesson where we'll be configuring a Route 53 geolocation routing policy. And first of all, we'll launch two EC2 instances, and we're going to launch them both in us-east-1 because we're going to be working in the AWS sandbox in the cloud playground. But in real life, if you were working with a production environment, then you'd probably have instances that are located in multiple different regions. Next, we'll install httpd, and then we'll configure a routing policy. So we'll create an A record using a geolocation-based routing policy. And then finally, we're going to test that everything is working and that our requests are being routed to the correct location, and your location where you are right now will determine where Route 53 is going to route your requests. So for example, if like me you're located in the UK, then your request is going to be routed to web server 1. But if, for example, you're located in USA, then your request will be routed to web server 2. And don't worry if you're not located in UK or USA, that's just my example, and you will be able to select your own country or continent. So now, if you'd like to join me in the console, then we'll get started. So from the console, search for EC2. Launch instance. Call

it WS1. We're going to use Amazon Linux, t3.micro, proceed without a key pair. Under Network settings, select Edit. We'll change the security group name and change the security group rule to be HTTP instead of SSH. Port range is 80, and source is Anywhere. Then scroll down to the bottom to Advanced details. Expand the advanced details and scroll down to the bottom until you get to the User data section. And we're going to add in a Bootstrap script, which you will find in the resources for this lesson. So here's my script. I'm going to copy that, paste it into User data, and you've already seen this script a few times before so you know what it does. I'm just going to modify the message that's going to appear on the website. So I'm modifying my website to say this version of the website is for customers who are located in and I'm going to put the UK, and I want you to type the name of the country that you are located in. So this is the version of the website that you should see when you log into the website. So once you've done that, select Launch instance. Next, I'll launch my second instance. So head back to Instances, Launch instance. The name is WS2. We're using Amazon Linux. Instance type is t3.micro. Proceed without a key pair. Edit your network settings. Scroll down and select Existing security group. It's the Web DMZ security group. Select Advanced details. Scroll to the bottom to the User data section, paste in your script, and we'll update the message on our website. So this version of the website is for customers who are located in, and I want you to type a country that is not the one that you're in. So if you're in USA, then you can just type India or Canada or pick a country. And I'm just going to type Barbados because that's where I'm going on holiday soon. And once you've done that, we can Launch instance. So now, let's search for Route 53. I'm going to open it up in a new tab. We can Ignore this message. It's not going to impact us, so just close that down and select Hosted zones. Click on your hosted zone and then scroll down. And down here, if you have any A records from the previous lessons that we've done, then go in and delete them. And this is what your hosted zone should look like. You should just have your NS record, which is for your name servers, and the start of authority record as well. So now, select Create record. We can keep the record name as blank. Record type is an A record. Down here, the value is going to be the public IP address of our web server 1. So back in the EC2 management console, select Instances. Select your web server 1. Here's our public IP address. I'm going to select that and copy it. Back in Route 53, paste in the value. Time to live can be 300. Routing policy is going to be Geolocation. And down here, the location refers to the location where the requests are coming from. And we can select either by continent or by country. And I want you to select the country that you are located in. So for me, that's going to be the United Kingdom. So I'll select United Kingdom, and you select your own country. We'll leave the health check as optional, and record ID is going to be 1. Next, we're going to add another record. The record name is going be blank. Record type is an A record. This time, the value is going to be the public IP address of our web server 2. So back on our EC2 console, select Instances. Here's our web server 2. Select the public IP address. And then back in Route 53, paste the value. Time to live is 300 seconds. And routing policy is Geolocation. Then for the location, select the country that you added in your web page. And for me, that was Barbados. So this is the country that you are not in. There we are. So any requests that come to my website from Barbados are going to be sent to this IP address. Health check ID is blank, and then record ID is going be 2. And Create records. So there are our two records. And you might be wondering, what if I get users logging into my website that are not from either of these countries? How do I create a default record? Well, you can also do that using geolocation routing. If I just show you by creating a record. We can paste in a value of our web server 2. Set it to Geolocation routing. And then under Location, if I just select Default, so if I make that a record ID of 3 and create the record, and there we have our default record. So if any user is visiting our website and they're not from the UK and they're not from Barbados, then their

traffic is going to be routed to this web server. So now, we are ready to test. And to do that, I'm just going to copy my domain name, open a new browser tab, paste the name in here, and hit Enter. And if it has all worked correctly, then Route 53 should have routed your request to the correct server. And for me, because I'm located in the UK, I should see a message like this. And my request has been routed to web server 1. And that is geolocation routing. So for my exam tips, just remember that geolocation routing allows you to route traffic based on the location of the requests that are coming from your users. For instance, you might want all the requests that originate from Europe to be served by web servers that are also located in Europe, and these web servers can be configured with the correct local language and with prices that are displayed in Euros as well. And that's just one example of when you might use geolocation routing. So that is it for this lesson. Any questions, please let me know. Otherwise, please join me in the next one. Thank you.

Demo: Configuring Route 53 Alias

Hello Cloud Gurus and welcome to this lesson where we'll be configuring a Route 53 alias. And we'll begin by launching an EC2 instance, and we'll configure our instance as a web server. So we'll install httpd. Next, we'll create an application load balancer, and we'll be configuring our EC2 instance as a target for our load balancer. And then we are going to configure a Route 53 alias, and I'll use this alias to map a friendly domain name that we own to our application load balancer. And this means we'll be able to access our website using our domain name instead of using the URL of our application load balancer. So, if you'd like to get started, then please join me in the AWS console. And from the console, search for EC2, Launch instance. Call it my web server. Use Amazon Linux. Instance type will be t3.micro. Scroll down, proceed without a key pair, edit your network settings, and select the subnet that is in us-east-1a. Next, I'm going to update the name of my security group so I can find it later. Call it MyWebDMZ. Scroll down to the rules, and we don't need a rule for SSH. So I'm going to change that to rule for HTTP on port 80. And the source is going be Anywhere. Then scroll down until you get to Advanced details and open that up. Then scroll down until you get to the User data section, and there it is right at the end. And this is where we'll add our Bootstrap script. And you'll find this Bootstrap script in the resources for this lesson. So I'm just going to copy everything. Come back to my EC2 instance launch screen and paste everything in here. And this is what it should look like. So we're just creating a very simple web server. So now we're ready to go ahead and Launch instance. And while we're waiting for our EC2 instance to initialize, we can go ahead and create our load balancer. Open up the menu on the left. Then scroll down until you find Load Balancers. And I'm going to open this up in a new tab. Select Create load balancer. And if you remember, there are a few different types of load balancer available. We've got the Application Load Balancer, which works with HTTP and HTTPS. There's the Network Load Balancer, which operates with TCP and UDP, and it also does TLS offloading, and this is the ultra high-performance option. Then there's the Gateway Load Balancer, which, if you remember, is for third-party virtual appliances. And then, scrolling down, we've also got the Classic Load Balancer, which is the previous generation option. But we are going to go with the Application Load Balancer, so make sure you've selected that one and select Create. First of all, I'll give it a name. I'll call it my-alb. It's going be internet-facing using IPv4. Scrolling down, make sure you've selected your default VPC. Under Mappings, we need to select at least two availability zones. And the first one will be us-east-1a because that's where our instance has been created. And I'm going to select us-east-1b as well. Scroll down. And the security group is going to be the one we created when our instance launched. So select the one that you just created called MyWebDMZ, and

I'm going to delete the default security group because we don't need it. Scrolling down to Listeners, the listener is going be HTTP protocol on port 80, and this is where we can create our target group and register our load balancer target. So select Create target group, and it's going to open up in a new tab. Target type is going to be an EC2 instance. Target group name is my-tg. Protocol is HTTP on port 80. The VPC should be your default VPC. Protocol version is HTTP version 1. Health check protocol is HTTP. And the health check path is going be index.html. Then scroll down to the bottom. Click Next. This is where we can add our target. It's already identified our available instances. There's my web server. So select your web server. Then you need to use this button here saying include pending as below. We can then review our targets, and your target should have appeared down here. Then go ahead and Create target group. Once you've done that, you can come back to your previous tab where we are creating the load balancer. Use this button to refresh the list of the target groups. Then, from the drop-down, you should be able to select your target group that we just created, and there it is. Once you've done that, scroll down to the bottom and Create load balancer. Now it can just take a few minutes to finish creating the load balancer. So if you feel like having a break, have a cup of tea or coffee. And by the time you are done, your load balancer should be ready. And you can view the progress if you select View load balancer. After a few minutes, we can refresh using this button. And eventually, it will show a state of active like this. So now, let's just check that our load balancer was configured correctly. Select your load balancer. Here's the DNS name of our load balancer. So copy that to your clipboard. We'll open a new browser tab. Paste our elastic load balancer DNS name into the browser tab and hit Enter. And if it's worked, then this is what it should look like. So now, we're ready to go ahead and create our Route 53 alias. Coming back to our AWS console, search for Route 53. And if you see an error message like this, you can just close it down. It's not going to affect anything we're doing here today. So now, on the left-hand menu, select Hosted zones. And you should have a public hosted zone already in your account. So select your hosted zone, and this is how your hosted zone should look. So you should have your NS record, which is the name server record, and the start of authority record, the SOA record. So now, we're going to go ahead and create our alias record. So select Create record. Our record name is going to be the same as our domain name, so we don't need to type anything in here. We'll just keep it blank. The record type is going be an A record, and this is going be an alias. So make sure you've selected this Alias radio button. Scrolling down, this is where we're going to choose the endpoint that we're going to be routing our traffic to. And with a Route 53 alias, we can create an alias to another record within our hosted zone. It could be an API gateway, CloudFront endpoint, or Elastic Beanstalk environment. And it could also be an elastic load balancer or a website hosted in S3. But we are going to be using our application load balancer, so select that. Next, we need to choose the region, and the region is us-east-1. And then finally, we'll choose our load balancer, and there it is. The routing policy is just going be the simple routing policy. Evaluate target health is set to Yes. And then scroll down and Create record. So now that has been created, we can just copy our domain name and paste it into a new browser tab and hit Enter. And if it's all worked, this is what you should see. So there we go. We've mapped our domain name to the domain name of our application load balancer. And of course, this is a much more user-friendly way to tell people about our website if you can provide them with a domain name that means something to you rather than giving them this really long cumbersome domain name that is associated with the application load balancer. And for the exam, just remember that a Route 53 alias allows you to map a user-friendly domain name to an AWS resource like an application load balancer. And it allows you to use your own domain name. So you can use a domain name that you own to access your website instead of using that rather long and cumbersome elastic load balancer domain name. And a

Route 53 alias is AWS-specific, and it allows you to route traffic to an Elastic Load Balancer, a CloudFront distribution, or an S3 bucket, as well as any other record in your hosted zone. So that is it for this lesson. If you have any questions, please let me know. Otherwise, I will see you in the next one. Thank you.

Understanding Route 53 Failover Alias Records

Hello Cloud Gurus and welcome to this lesson, which covers Route 53 failover alias records. And we'll begin with what is a Route 53 failover alias record? We'll take a look at an example use case, and we'll cover my exam tips as well. So what is a Route 53 failover alias record? Well, if you remember, an alias record is a special type of record in Route 53 that enables you to route traffic to AWS resources like CloudFront, S3 hosted websites, and elastic load balancers. A failover routing policy is a policy that tells Route 53 to route traffic to a secondary resource when the primary resource is unhealthy. So bearing that in mind, what do you think a failover alias record is? Well, imagine an alias record and a failover routing policy had a baby. So with a failover alias record, this allows you to create an alias record where you configure one AWS resource as primary and another as secondary. But let's take a look at a simple example. Imagine you have a website running on EC2 behind an elastic load balancer. You've got a Route 53 hosted zone with an alias record that resolves to your elastic load balancer. And if you remember, a hosted zone is simply a container for your DNS records. Well, in this scenario, you can use a failover alias record to route requests to a static website hosted in S3 if there is ever a problem with your EC2 website infrastructure. So if your EC2 instances are not reachable for whatever reason, Route 53 will route requests to the secondary record, which resolves to a static version of your website hosted in S3. So you've got a primary record resolving to the elastic load balancer and a secondary record that resolves to S3. And it's these two records that make up the failover alias record. So for the exam, just remember that an alias record routes traffic to AWS resources like CloudFront, S3 hosted websites, and elastic load balancers. A failover alias record configures one AWS resource as primary and another as secondary. So you can have the primary failover alias record routing to your elastic load balancer and the secondary failover alias record pointing to a static version of your website in Route 53. And that is the use case that we talked about. So you can route requests to a static website in S3 if there is ever a problem with your EC2 hosted website. So that is it for this lesson. Any questions, please let me know. Otherwise, I'll see you in the next one. Thank you.

Demo: Configuring Route 53 Failover Alias Records

Hello Cloud Gurus and welcome to this lesson where we'll be configuring Route 53 failover alias records. And we'll begin by launching an EC2 instance and installing httpd. Next, we'll create an S3 bucket with static website hosting enabled. And then we're going to configure our failover record set. The primary target will be our EC2 instance, and the secondary target is going to be our S3 hosted website. And then we can test that everything is working by stopping our EC2 instance. So now, if you'd like to join me in the console, we'll get started. From the AWS console, search for EC2 and Launch instance. Call it WS1. We'll use Amazon Linux. Instance type is t3.micro. We can proceed without a key pair. Under Network settings, we're going to allow HTTP traffic from the internet, and we don't need SSH. Scroll down to Advanced settings. Open it up. And then scroll down to the bottom to the User data section, and this is where we're going to add our Bootstrap script. And you will find a link to all of the resources in the Resources section for this lesson. So I'm going to go into the EC2 folder, and here's my Bootstrap script. I'm just going to copy everything

and paste into my User data section. Now you've already seen this script a few times before, so you already know what it does. So now we can go ahead and Launch instance. Next, I'll create my S3 bucket. But for this to work, the bucket needs to have the same name as our Route 53 hosted zone. So first of all, search for Route 53. Open it up in a new tab. We can ignore this error message. It's not going to affect us. So close that down and come to Hosted zones. Here's our hosted zone, and I just need to copy the name of my hosted zone so that I can use it to create my S3 bucket. Next, search for S3. I'll open it in a new tab. Select Create bucket. Paste in the name of your bucket, and it must be exactly the same. Region must be us-east-1. Scroll down. Disable the block public access settings and acknowledge. Then scroll to the bottom and Create bucket. Next, we need to configure our bucket to host a website. So let's upload our website file, and you'll find it in the resources for this lesson. So in the S3 folder, this is the file we're looking for, index.html. Select the raw view, then right-click, select Save As, and we're going to save it to our local machine. And make sure you save it as index.html, and it's going to be saved to my Downloads folder. Once you've saved your file, come back to the S3 console, select your bucket, Upload, Add files, select your file, and upload the file. Next, we need to add a bucket policy. So close that down, select Permissions in your bucket, scroll down to Bucket policy, and select Edit. And I've provided a policy that you can use in the resources for this lesson. So let's go back, back. And under the S3 folder, you will find the bucket policy. And there it is. So just copy everything, come back to your S3 bucket, paste in the policy, and this policy is going to allow public access on all of the objects in our bucket. And all we need to do is replace our bucket name here. So I'm going to copy my bucket name here and replace it down here. And when you finish, this is what your policy should look like. And don't make the mistake of pasting over this /* because we need this to make sure that the bucket policy applies to all of the objects within the bucket. Once you've done that, you can ignore this error and hit Save changes. And there we go. That is our bucket policy configured. And if you scroll up, you will see the that your bucket is now publicly accessible. Once you've done that, we can now go ahead and configure static website hosting. So now, select Properties. Come down to the bottom, find Static website hosting, and select Edit. Select Enable. Down here, we need to specify the home page, which is going be index.html. Then scroll down and Save changes. We can test that everything is working by scrolling down in the Properties section. Here's our bucket website endpoint. Right-click and open it in a new tab. And there we go. If it's all working, this is what you should see. But let's also check that our EC2 instance has come up correctly. So come back to EC2 and select your instance. Select the instance ID, grab the public IP address. I'll open a new browser tab, paste in the IP, and hit Enter. And there we go. So both our EC2 instance and our S3 website are working correctly. So now, let's head to Route 53 and create our health check. I'm going to close down some of these pages so I don't get confused and come back to the Route 53 console. In the left-hand menu, select Health checks. Create health check. The name will be My Health Check. We'll specify an endpoint using the IP address, and it's going to be the IP address of my EC2 instance. So paste in the IP address of your instance there. Host name we're going to leave blank. Port is going be 80. And path will be index.html. Then scroll down. Hit Next. We're not going to create an SNS notification or alarm and just Create health check. And it does just take Route 53 a few minutes to establish the status of your health check, so don't worry if it says Unknown for now. But now, we can go ahead and create our failover record set. Come to your hosted zone. Select the hosted zone. This is what it should look like. And you should only have two records in your hosted zone. So you should have the NS record for the name servers and the start of authority record. If you have any a records in here from previous lessons, then go ahead and delete them. So now, select Create record. The record name will be blank. Record type is an A record. The value is going to be

the IP address of our web server. Under time to live, I'm going to change it to 10 seconds because I want things to happen as fast as possible. Routing policy is going to be a failover routing policy. Record type is Primary. Health check is the one we just created. And record ID is 1. Next, we'll add another record, and this is the other part of our failover record set. Record name is going be blank. Record type is an A record. This time, we'll be using an alias because we're routing traffic to our S3 website. Under select endpoint, Select alias to S3 website endpoint. Select our region, which is us-east-1. And then, if you click underneath, it will automatically find your S3 endpoint, so select that. Routing policy is failover. Failover record type is Secondary. We're not going to use a health check this time. And set Evaluate target health to No, and record ID is 2. Once you've done that, Create records. And that is our failover record set created, the first one routing to our EC2 instance, and the second one is an alias routing to our S3 website. So now we are ready to test if it all works. And first of all, let's check if our domain name resolves properly. So I'm going to copy my domain name, open a new browser tab, and paste in my domain name and hit Enter. And there we go. If it's all worked, your page should be being served from EC2. So far, so good. But what's going to happen if my instance goes down? Well, let's test it. Come to the EC2 console. And within your instance, we are going to stop our instance. I'm going to come back to my website and refresh. And eventually, I'm getting a Gateway Timeout. And you might see this kind of error or something similar. Now it's good to be aware that in practice Route 53 failover is not instant, and it can take a few minutes for it to fail over the DNS record. So just be patient and give it 5 or 10 minutes or so. And after that, we should be good to go. And there we go. It did actually take about 2 or 3 minutes for it to happen. So don't be surprised if it does just take a few minutes to do the failover. So for my exam tips, just remember that a Route 53 failover alias record allows you to configure one AWS resource as a primary and another as a secondary. A health check is used to determine the health of the primary target. And if the health check fails, then Route 53 will route requests to the secondary target. So that is all for this lesson. Any questions, please let me know. Otherwise, I will see you in the next one. Thank you.

Overview of CloudFront

Hello Cloud Gurus, and welcome to this lecture, which is all about CloudFront. And CloudFront is Amazon's content delivery network, or CDN, and this is just a theoretical lecture to introduce the main concepts so that you've got a good understanding of everything before we go ahead and build our own CloudFront distribution later on. So, we'll be covering what is a content delivery network, CloudFront terminology, how you can improve your website's performance using CloudFront, Time to Live, or TTL, and also, my exam tips. So, what is CloudFront? Well, it's a content delivery network, which is a system of distributed servers which deliver webpages and other content, and it's an easy and cost-effective way to distribute content with low latency and high transfer speeds. So, what does that mean? Well, this is all about making your webpages faster, and CloudFront speeds up the distribution of your static and dynamic web content. So think HTML, JavaScript, image files, videos, and web applications. So basically, any web content that you can think of. And it's a much more efficient and performant way for geographically distributed users to access your content. So, how does it all hang together? Well, imagine you are running a website located in the UK, and you have many users based all around the world, and when a user browses to your website, that request is routed through many different networks to reach your server, and the number of network hops and the distance the request needs to travel has a significant impact on the performance and the responsiveness of your website. And the network latency is, of course, going to be different for each

different location. And for users who are further away, they are going to experience greater latency and worse performance. So for example, users based in Australia, they might experience worse performance and greater latency, and they might think that the website seems less responsive than users based in India, for example, which is geographically closer to London. So, how can we improve performance and make it more consistent for everybody? Well, this is where CloudFront comes in. So, imagine you are running your website in London, and here are all your users. Instead of each user accessing the website directly from the server in London, instead, we introduce this concept of Edge locations, and an Edge location is simply a collection of servers which are in geographically dispersed data centers. And these Edge locations, they are used by CloudFront to keep a cache or copy of your objects. So copies of your HTML files, your images, and all the content that you are serving. So this means that instead of requesting content from your server located in London, users can access the content from the Edge location instead, which is physically much closer to them than the main server in London. And the way it works is that once the request is made, the Edge location forwards the request on to the server located in London, and it downloads the files requested and caches them locally in the Edge location. So this means that the next time the user requests the same file, or if another user comes along and they want to access the same file, they can access directly from the Edge location, and this provides a much faster response time. And it means that all your requests are only going to the local Edge location, and they are not going all the way to London. So let's review some of the CloudFront terminology that you will need to know for the exam. So, first of all, we have CloudFront Edge locations, and this is the location where the content is cached, and it's separate to an AWS Region or Availability Zone. Then, we have the CloudFront Origin, and this is the origin of all the files that the distribution will serve, and this can be an S3 bucket, an EC2 instance, an Elastic Load Balancer, or Route 53 address, and it can also be your own server in your own data center. And then finally, we have the CloudFront Distribution itself, and this is the name given to the origin and the configuration settings, for the content that you wish to distribute using the CloudFront content delivery network. And if we head back to our diagram, here's our Edge location, here is our Origin, and the CloudFront Distribution is the name given to the origin and the configuration settings for the content that we're distributing. Now, the reason that we use CloudFront is to improve the performance of our website for remote users, and you can deliver your entire website using CloudFront, and AWS operates a global network of over 200 Edge locations. And with CloudFront, requests for your content are automatically routed to the nearest Edge location. So this means that your content is delivered with the best possible performance, no matter where your users are located in the world, and this allows you to optimize performance for your users accessing the website from all around the world. And here at ACG, we actually use a CloudFront ourselves, and we have a lot of users from Asia, Africa, North America, South America, Europe, and Australia, and by using CloudFront, this helps keep performance consistent for everybody. Now, CloudFront is optimized to work with lots of other AWS services, and it's really well integrated with services like S3, EC2, Elastic Load Balancer, and Route 53 as well, and that makes it really easy to configure, and it also works seamlessly with any non-AWS origin server, which stores the original, definitive versions of your files and your website. Now with CloudFront, objects are cached, and they are cached for a period of time, which is their Time to Live. And the default Time to Live is 1 day, and when the TTL is up, the object is automatically cleared from the cache. Now you can clear an object from the cache yourself before the Time to Live is up; however, you will be charged for doing that. Let's say, for example, you've had a new version of an object, say a new image or a new version of one of your webpages, you can actually go in there and clear the cache yourself, to make sure that when your users try to access the

file, they are going to see the latest version. So, let's take a look at some of our exam tips. So, we've got Edge locations, and this is the location where the content will be cached, and it's separate to an AWS Region or Availability Zone. We also have the Origin, and this is the origin of the files that the distribution will serve, and that can be an S3 bucket, or an EC2 instance, an Elastic Load Balancer, or Route 53 address, and it can also be a server in your own data sensor. And the CloudFront Distribution is the name given to the origin and the configuration settings for the content that we are distributing. Objects are cached for the Time to Live, and you can clear cache objects, but just remember you will be charged for doing that. And the reason why CloudFront is so great for improving the performance of your website, is that requests for your content are automatically routed to the nearest Edge location to your users. And this allows you to optimize performance for your users accessing your website, no matter where they are in the world. So, that's it for this lecture. If you have any questions, please let me know. Otherwise, I will see you in the next lecture. Thank you.

Demo: Configuring Amazon CloudFront

Hello Cloud Gurus and welcome to this lesson where we'll be creating a content delivery network using CloudFront. We'll begin by creating an S3 bucket, and I'll be creating my S3 bucket in a location that is as far away from me as possible. But if you are working in our AWS sandbox, please just do everything in us-east-1. Next, we'll upload an image, and we'll try and access the image from our S3 bucket, and I'll show you the kind of latency that you can incur when trying to access an object that is located in a bucket that is not geographically close by. Then, we'll create a CloudFront distribution, and then we're going to access the same image using CloudFront after its cached the image. And it will automatically then send our request to the closest edge location, and we will be able to compare the response time between accessing our files using CloudFront and accessing it directly from an S3 bucket that is far away. So from the console, search for S3 and select Create bucket. I'm going to call it my-cf-origin and then add some random numbers on the end. AWS region, for you, it's going to be us-east-1. This one. But I am using my own AWS account, so I'm going to create my bucket in Asia Pacific. And I'm going to select Sydney. But if you're using our AWS sandbox, please create everything in us-east-1. Scrolling down, I'm going to deselect the Block public access settings, acknowledge, and then scroll to the bottom and Create bucket. Now we also need to add a bucket policy. So I'm going to select my bucket, select Permissions, scroll down. Under Bucket policy, select Edit, and this is where we can add our bucket policy. And you will find a sample policy in the resources for this lesson. It's called bucket_policy.json. Here it is. I'm just going to copy everything and paste it into my bucket policy here. Under Resource, we need to update this with our bucket name. Here it is. So I'll just copy the bucket name and paste it in here. And when you've finished, your policy should look like this, and it's just going to enable read access to everything in our bucket. And make sure you still have these last two characters here, so the forward slash and the asterisk after your bucket name. Then scroll down and Save changes. Next, we'll upload a file. So I'll select Objects, Upload, Add files, and I'm going to add an image file, and it's this one. So I'm selecting a fairly large image file. It's about 6.3 MB in size. So select Open and Upload. And the reason I want to use an image file is so that we can actually see it loading from the other side of the world, and you will see how slow it is. And if you're looking for an image file, you will find this file in the resources for this lesson. So go ahead and click Upload. And you can actually see the that it's pretty slow to upload. Now, let's try accessing this file in S3. I'll close that down. Select the object. And then using this object URL, right-click and open the link in a new tab.

And it might be loading faster for you, but check out just how slow it is for me because my request is going all the way from London to Sydney, Australia. And there we go. It's just a picture of my little dog, Ralphie, when he was a really cute puppy. Now if I'd selected London as my region for this S3 bucket, then I could expect the image to appear pretty much instantaneously. And just imagine you're running a website with loads of different images on there, do you really want your customers seeing images loading up as slowly as that? Probably not. And that is where CloudFront comes in. So now, let's go ahead and set up our CloudFront distribution. In the console, I'm going to search for CloudFront. I'll open it in a new tab and Create distribution. We'll select an origin, and this is going to be our S3 bucket. There it is. Origin path refers to different folders within your origin. Say, for example, you've got different folders for images, videos, and HTML files. Then you can specify the folder name you want to use here. And we don't have any, so I'll just leave that as blank. It's already populated the name for our origin. Under Origin access, there's the option to restrict access to only CloudFront. So with this option, there would be no direct access to the S3 bucket. Let's just stick with the default for now. Moving down to Default cache behavior settings. Path pattern just determines which requests the following settings will apply to. And by default, it's going to apply to all requests. There's the option to compress objects automatically. Then, under Viewer protocol policy, we can set the supported protocols. So you've got the choice to allow both HTTP and HTTPS, Redirect HTTP to HTTPS, or allow HTTPS only. And I'm going to select Redirect. So any request that comes in as HTTP will automatically be converted to HTTPS. Then, under Allowed HTTP methods, this is pretty cool because it's not just about reading and accessing and downloading files. You can also put and post, which means that you can upload files using the CloudFront edge location, and AWS will manage the transfer of those files from the edge location into your S3 bucket or whatever you've got configured as your origin. Restrict viewer access allows you to restrict access to only users who use CloudFront signed URLs or signed cookies. So this allows you to choose whether you want CloudFront to require users to access your content only if they have a signed URL or a signed cookie. And just imagine you have a website and some of the content on your website is paid for content. What you can do is then restrict the access to the paid for content to only the users with the signed URL. Now the users can't share their URL because it's individual to them, and you only release a signed URL to users who've actually paid to view the content. And that's actually a technology that we've used here at A Cloud Guru because we have some content on our website which is for members only, and we automatically generate a signed URL to allow our members to access the paid-for content. Moving down here, we've got the cache policy. And if you select View policy, this is where you will find the time to live settings. And TTL is the minimum amount of time in seconds that you want your objects to stay in the CloudFront cache. So basically, how long is the object going to be cached for before it expires and gets wiped from the cache? And the maximum is 31 million seconds or 365 days. And the default is 86,000 seconds, which is 24 hours. Now the thing to be aware of here is that you might have some files, which get uploaded much more frequently than 24 hours. Just imagine you had files that were getting updated on an hourly basis. Well, this time to live is not really going to work out for you, and your files are going to become out of date pretty quickly. So in that case, you're going want reduce your time to live and think about how frequently your data is changing and set a time to live that is appropriate for you. So that is time to live. I'm going to close that down and head back to the console. Under Origin request policy, if I select Create policy, this is where we can configure CloudFront to forward HTTP request headers and cookies. And CloudFront can actually serve different versions of our objects based on the content of the request header or cookie, and this is where you can set that up. So you can tell CloudFront to include request headers and cookies in the

origin requests, and I'm just going to cancel that, close that window, and come back to my original page. Then scrolling down to Function associations, you can actually associate a function that is invoked by CloudFront, and this allows you to write your own code to customize how CloudFront is going to respond to HTTP requests. And you don't really need to know about this for the exam, so we're not going to talk about that much further. Scrolling down to Price class, and this allows you to control which locations are going to serve your content. And you can restrict to only certain continents, for instance Northern America and Europe or Northern America, Europe, Asia, Middle East, and Africa. But do be aware that if you decide to restrict and use one of these options, you can save money, but you may not get the best performance. Moving down, we also have configuration for web application firewalls, and you can use the AWS WAF to provide traffic filtering for well-known attacks like SQL injection attacks and cross-site scripting attacks as well. And the Web Application Firewall is very tightly integrated with CloudFront. So you can choose your web ACL or AWS WAF configuration, you can choose it directly from within CloudFront. Down here you have the option to use your own registered domain name if you want to use that as the URL for your CloudFront distribution. And if you want to do that, you can just add it in here. Now if you remember, at the beginning, we selected the option to redirect all HTTP to HTTPS, and CloudFront actually has its own SSL certificate that it is going to use by default. But then down here, there is also the option to associate our own certificate from AWS Certificate Manager. Then down here, we've got the supported HTTP versions. We can specify a default root object to return if we want to. I'm just going to leave that blank. We can enable logging, and it will log to an S3 bucket. I'll keep that to the default. And IPv6 is supported by default. So now, let's go ahead and Create distribution. And it does just take a few minutes for it all to get created because CloudFront is provisioning to all the different edge locations around the world, and it's replicating your distribution setup to all of those edge locations. And as you can imagine, that's a lot of locations, and it's just one of those things in AWS that can take a little bit of time. So now is a good time to have a cup of tea or coffee. Take a break for a few minutes. And when you come back, it should be ready. So now let's take a look at our distribution. Here's our distribution domain name. Then under Geographic restrictions, this is where you can prevent users in selected countries from accessing your content, and you can actually create an allow list and a block list as well. So you can select the countries that you would like to allow, and this is great if you are restricted from operating in a particular country because you can go in and block the countries that you are not allowed to operate in. So now, let's go back to our general settings and see if we can access our file using CloudFront. So just copy the domain name. And in a new browser tab, paste your domain name. But before we hit Enter, we need to add our file name on the end. So type forward slash and the name of our file. So the name of our file is cute_dog.jpg and hit Enter. And there we go. And it wasn't instant because this is the first time we're accessing the file, and CloudFront needs to cache it from my origin. But now, because this file is going to be cached in my local edge location, it should be much faster to access next time. So I'm going to try it using a different browser. So this time, I'm going to try it using Safari. I'll copy my CloudFront URL, paste it into Safari, and hit Enter. And there we go. It should just load a little bit faster. And then one last thing I wanted to show you is invalidations. So if you select the Invalidation tab and select Create invalidation, this is all about removing files which are cached in our CloudFront edge location, for instance if the file has changed in your origin, but you've still got the old copy cached in CloudFront. And you can use an invalidation to remove that object from the cache immediately. So if you provide the object path, then you can create an invalidation, and it's going to remove that object from the cache. But do be aware, this is a manual process. And every time you do it, you are going to be charged a fee. And for the exam, just remember that CloudFront

is a content delivery network, and it can be used to speed up the delivery of your static content to viewers all around the world. The origin is the location where your original content is stored, and users access your content from a local edge location instead of directly accessing the origin, which may not be geographically close to them. Any questions, please let me know. Otherwise, please join me in the next lesson. Thank you.

Demo: Configuring Amazon CloudFront with Origin Access Control

Hello Cloud Gurus and welcome to this lesson where we'll be configuring CloudFront with origin access control. And this is used to restrict access to our content and force all requests to come through CloudFront instead of going directly to an S3 bucket. And we'll begin by creating an S3 bucket. We'll enable public access and upload an image to the bucket. Next, we'll create our CloudFront distribution and test that everything's working. After that, we'll update the bucket policy to remove public access and to only allow CloudFront to access the contents of the bucket. And we'll also reenable the block public access settings. After that, we are going to create an origin access control, and then we'll update our CloudFront origin to add the origin access control. We'll then need to wait for it all to finish deploying before we can test everything. So if you are ready to get your hands dirty with origin access control, then please join me in the console. From the console, search for S3 and Create bucket. Call it my-cf-origin and then add some random numbers on the end. Region is us-east-1. Scroll down and remove the block public access settings for this bucket and acknowledge your settings. Then scroll down to the bottom and Create bucket. Select your bucket name, select Permissions, and we're going to update our bucket policy. So scroll down to Bucket policy, select Edit, and you will find the policy to use in the resources for this lesson. And it's this one, public_bucket_policy. Here's our policy. I'm just going to copy everything and paste it into my policy. Now the one thing we need to update is our bucket name here. So we want to replace this with our bucket name. Here it is. So I'm just going to copy it and paste into the policy. And make sure you don't get rid of this trailing forward slash and asterisk because this means that the policy will apply to all the contents of the bucket. Once you're done, scroll down to the bottom. We can ignore this message, it won't affect what we're doing, and Save changes. Next, we're going to upload an image file, and you will find a JPEG file to use in the resources for this lesson, or you can use your own image file. So select Objects, Upload, Add files. Here's the file I'm going to add. Scroll down and Upload. After you've uploaded your file, you can just close that page. And now we are going to create our CloudFront distribution. So search for CloudFront. I'll open it in a new tab and Create a CloudFront distribution. We'll select the origin, which is going to be our S3 bucket that we just created, and we can accept all the rest of the defaults. So just scroll down to the bottom and Create distribution. Now it does just take a few minutes to finish deploying. So now's a good time to have a cup of tea or coffee. And in 5 minutes, we should be good to continue. So that is our distribution created. And once it's enabled, you can just select your distribution domain name. Open a new browser tab. Paste in the domain name. But before we hit Enter, remember to add the name of our file. So add /ryan_faye.jpg and hit Enter. And you must get the file name right or it won't work. So far, so good. But we've still got public access enabled on our bucket, and we want to force all access to come through this CloudFront URL. So I want you to head back to your S3 bucket, select Permissions, scroll down to your public access settings, Edit, and we're going to select Block all public access and Save changes. Confirm your settings. And then next, we're going to update the bucket policy. So scroll down to Bucket policy, select Edit. I'm going to remove this policy. And in the resources for this lesson, there's a new policy that we're going to add. And it's this one. It's called

bucket_policy.json. Here's our policy. I'm just going to copy everything. Come back to S3. Paste in the new policy. Now there are two things we need to change in this policy. First of all, we need to update our bucket name. So there's our bucket name. And then the second thing we need to update is the ARN of our CloudFront distribution. So in CloudFront, select the ARN, and you want to copy the ARN. It's this one here. Back in your policy, just paste in the ARN of your distribution. And this policy is basically going to allow get object access for our bucket for only our CloudFront distribution. And once you've finished, this is how it should look. So now scroll down. We can ignore this error. It's not going to impact us and Save changes. And those two changes on our S3 bucket have removed public access from our bucket. So next, we need to configure our origin access control. So come back to CloudFront, open the left-hand menu, and then select Origin access on the left. Then select Create control setting. Give it a name and a description. I'm just going to call it My OAC. We'll accept the default settings and Create. Next, we'll add the origin access control to our origin. So come to Distributions on the left, select your distribution, select Origins, select your origin using this radio button, and Edit. Then scroll down to Origin access and select Origin access control settings, and this is the recommended setting, which means that the bucket will restrict access to only CloudFront. Under Origin access control, select the origin access control you just created and Create control setting. I'll give it a description and hit Create. Once you've done that, scroll down to the bottom and Save changes. Once you've done that, select the General tab, and we can see the status under this last modified section. And it's telling us that it is still deploying. Now it does just take a few minutes to update the settings, so let's be patient. And as soon as it's finished deploying, then we can test that everything is working. After a few minutes, it's finished deploying, and we are ready to test. So select your distribution name and copy that. We'll open a new browser tab, paste in the name, then type forward slash and the name of our file. And it's ryan_faye.jpg and hit Enter. And if it's all worked, you should be able to access your file, and we've restricted access to the bucket to the CloudFront service only. So that is origin access control. And in the next lesson, we're going to use this CloudFront distribution again. So don't close down your AWS sandbox, and don't delete anything because we will be using this in the next lesson. But before we go on to that, let's take a look at my exam tips. And for the exam, just remember that CloudFront provides you with a content delivery network, which can speed up delivery of your static content to viewers all around the world. Origin access control allows you to restrict access to the contents of your bucket so that all users must use the CloudFront URL instead of a direct S3 URL. Origin access control allows us to restrict access to the contents of our bucket so that all users must use the CloudFront URL instead of a direct S3 URL. And this allows us to remove the public access from our bucket configuration. So that is it for this lesson. And if you're ready to continue and take a look at CloudFront logging, then I'll see you in the next one. Thank you.

Demo: Interpreting CloudFront Logs

Hello Cloud Gurus and welcome to this lesson where we'll be taking a look at CloudFront logs. And we'll use the CloudFront distribution that we created in the previous lesson. We'll create a new S3 bucket, and we're going to use that to store our CloudFront logs. Next, we'll enable logging on our CloudFront distribution and then generate some traffic to it. And then after a few minutes, we should see some logs appearing in S3. But just do be aware that when you first enable logging, it can sometimes take a little while for the logs to appear, and that's just due to the distributed nature of CloudFront. So don't be surprised if it does take a few minutes to see some logs. Well, if you're ready to get started with CloudFront logs, I'll see you in the console. And from the console, I'm

going search for CloudFront. And you will need to locate the distribution that we created in the previous lesson. And if you don't have it anymore, then don't worry. You can just pause the video here and take a few minutes to go back to the previous video and recreate it. So here is my distribution that we created in the previous lesson. The next thing I'll do is create an S3 bucket, and that's going to be used to store our logs in. So search for S3, open it in a new tab, and Create bucket. I'm going to call it my-cf-logs and add some random numbers on the end. Region is us-east-1. And now scroll down and Create bucket. Next, return to CloudFront. Select your distribution. Select Edit. And then scroll down until you find its Standard logging. There it is. And just select On. Select your S3 bucket that we just created. It's this one. And it's giving us this message that we need to enable ACLs for the selected bucket to allow CloudFront to write the logs. So I'm going to select Enable ACLs. It's now giving me this message, Bucket is ready for logging. So now scroll down. Save changes. And then in our settings, we can see that standard logging is enabled. So now that logging is enabled, let's generate some traffic to our CloudFront distribution. And to do that, I'm going to copy my domain name, open a new browser tab, paste in the domain name. And remember, we need to add the name of our file. So add forward slash and the name of our file. It's ryan_faye.jpg and hit Enter. And I'm just going to copy that domain name and open a few more browser tabs and paste in my domain name just to generate some more traffic to CloudFront. After you've done that, head back to S3. Open your CloudFront logs bucket, and let's see what we've got. Now it does usually take a few minutes for our log files to appear in the bucket, so just be patient. Have a quick break and step away from the screen for a few minutes and they should appear very soon. And there we are. After just a few minutes, a log file has appeared. And if it's not showing, then you can just use this button to refresh. So let's open up our log file. So select our log file name and select Open. And it's actually going to download to my local machine, and here is my log file. And it should appear as a zipped file in your Downloads directory. Now there's an awful lot of detailed information in here about our CloudFront requests. So here, we've got the request date. Here's the request time. This is my edge location. Here's my client IP address. The request is a GET request. Here's the object that was requested. Here, we've got any HTTP response codes. Over here is some information about my client machine and the browser that I use to access. And here is our edge location response. So it was a cache hit. Pretty cool, right? You get an awful lot of information in these CloudFront logs. So that is it for CloudFront logs. And for my exam tips, just remember that you can configure CloudFront to create logs that contain very detailed information about every user request that CloudFront receives. The files are stored in S3, and they are known as standard logs or access logs. What gets logged? Well, it's very detailed request information, so things like the request time, the edge location, the client's IP address, the method, for example a GET request, the object that was requested, the HTTP status code, and edge location response, for instance a hit, a miss, a redirect, or an error. CloudFront usually delivers the log files to S3 within 1 hour of the events happening. And you can actually receive multiple access logs per hour. And for that reason, you can end up with an awful lot of data to review. Therefore, they recommend that you actually consolidate your data and analyze it using Athena. So you can actually combine the logs that you want to analyze into one single file, and then you can go ahead and analyze the data using Athena. So that means that you can run standard SQL queries on the data. Well that's all for this lesson. If you have any questions, please let me know. Otherwise, please join me in the next one. Thank you.

Identifying CloudFront Caching Errors

Hello, Cloud Gurus. And welcome to this lecture, which is going to cover some of the common errors that you might see with CloudFront. And we'll start off by looking at a CloudFront scenario. We'll then go on to talk about 400 client-side errors. We'll briefly cover 404 errors, which you might have seen before. We'll then cover that the 500 errors. And these are server-side errors. I've got a little aide memoire to help you remember client-side and server-side errors. And we'll go on to my exam tips. So let's take a look at this scenario. We've got a web application, and it's being served from an EC2 instance, and we've also got some files in S3. And then the website is sitting behind a CloudFront distribution, and then we've got our users on the left-hand side, and they are accessing the application using CloudFront. So then imagine, there's some kind of problem with the website, so something's gone wrong. And when that happens, CloudFront is going to return an HTTP status code, and generally it's a 400 or a 500 status code, which means that something has gone badly wrong. Now, if you've been using AWS for a while, you will probably already be familiar with some of the 400 and 500 status codes that you can get and it's not just AWS. Lots of web-based services will also give you these kind of status codes occasionally. And you might see one or two questions in the exam where they're referring to an HTTP status code. And then they'll go on to ask you what this status code or what this error code relates to. And I wouldn't suggest that you try to memorize every different status code that there is. It's much more beneficial to just remember the difference between a 400 and 500 status code. And then you can use that information within the context of the situation or the scenario to help you work out what could have gone wrong. So first of all, let's take a look at the 400 errors, and a 400 error is always a client-side error. So usually there's something wrong with whatever the client is requesting. And I've got a few examples here. So first of all, we have the 400 error, and this means a bad request. So there's a problem with your request. For example, it's a malformed request, and maybe the request has come in using an incorrect format. Moving on to 403, and this always means access denied. So if you are running a website, of course your files must be publicly accessible. So that means all of your HTML files, your index.HTML, any images that you are serving on the website, etc., all of those will need to be publicly accessible, otherwise, your users are going to see this 403 access denied error. And let's say for example, you are using S3 to host your website. Then you will need to enable public access for the S3 objects, and that includes permission to run the S3 GetObject API call. And if you don't have public access enabled on your S3 website, of course your users are going to see this 403 access denied error. And this is one that can come up a lot in the exam. So the 403 one is definitely one that is worth remembering. And then the last one that I wanted to remind you about is the 404 error. And you've probably seen this one loads of times before when using the internet, and it's a really, really common error message, and it simply means the object does not exist. And in the context of our scenario, when we talk about client-side errors, it always means that something has gone wrong on the left-hand side of the diagram. So something within the client request has gone wrong. And the way that I try to remember it, is I think of 404 and 404 is always your fault. And because you've tried to visit a webpage that doesn't exist. So I try to remember there's 404 that you often see when you try to visit a webpage that doesn't exist. You know maybe you've typed in the address incorrectly, or maybe you've just tried to visit a webpage that just doesn't exist anymore. So I just try to remember if it's 404, it's always my fault that I've typed the address incorrectly or I've tried to visit a page that does not exist. So moving on to server-side errors. And 500 errors are always server-side errors. And I've got a few examples here beginning with 502, bad gateway. Now, generally this means that CloudFront cannot connect to the origin, So it can't connect to either the

S3 bucket or EC2 instance, which is serving your website 503, which means service unavailable. This usually indicates performance issues on your server so it could be that the server is receiving more requests than it is able to handle. And then finally we have 504, which is gateway timeout. And this is what happens when you get the request expiring before a response is received from the server. And once again, that can also indicate a performance issue because the server is not able to respond to all the requests that are coming in. And the common theme here that you may have noticed, is that all of these errors are often caused by high traffic to the website, and the server either not responding or not being able to deal with all the requests that are coming in. So when you see a 500 error, these are usually caused by high traffic to the website and the server not responding. So just remember, that 500 errors are always server-side, and the problem is always related to something going on on the right-hand side of the diagram, and it's often a performance-related problem with the EC2 instance or the S3 bucket. And your EC2 instance cannot handle the number of requests that are hitting your website. And to help you remember, I always think that the number 5 looks a little bit like the letter S. So I try and use that to remember that a 500 error is always a server-side error. And if you are shortsighted like me, if you wear glasses like me, then whenever I take my glasses off, the number 5 pretty much always looks like the letter S. So moving on to our CloudFront exam tips. First of all, remember that 400 errors. They are always client-side errors. And try to remember that familiar 404 error that you've probably seen many, many times, and just remember that that is a problem with the client-side request. So your client is trying to request something which does not exist. And if you do see a 404 error, first of all, you'll want to check that the requested object actually exists. Check that you have access to it, and check that your request is correctly formed as well. And finally, 403. It does tend to come up a lot in the exam, and this is the one that indicates that access is denied. And you can see this in relation to a lot of things on the internet, and you can definitely see it in relation to just general websites in relation to CloudFront and in relation to S3 as well. So do try to remember that 403 is access denied. Now when it comes to 500 errors, remember that those are always server-side errors, and the number 5, it looks a little bit like the letter S so that will just help you remember that it is always a server-side error. And 500 issues, they are generally caused by the server not responding for some reason. So maybe your server's gone down or you've got a performance issue, and it is usually due to high traffic to your website, and your server's becoming overwhelmed. So when you see 500 errors, I want you to think about capacity, and performance issues, and servers not responding. Now in the exam, they might give you a scenario, and ask you about a very specific error message, and the best thing that you can do is try to think about it within the context of the question. So are they asking you about a client-side error or is it a server-side error? And if you're really struggling to find the answer, then just try going through a process of elimination, and try and work out which answer seems to be the best fit. You know which fits the scenario best, and which seems the most likely. And once again, the one that comes up time and time again is the 403 access denied message. So definitely do try to remember that one if you can. So that is the end of this lecture. That's everything I wanted to tell you about the CloudFront error messages. If you have any questions, please let me know. If not, I will see you in the next lecture. Thank you.

CloudFront Cache Hit Ratio

Hello, Cloud Gurus, and welcome to this lecture, which is going to cover cache hit ratios. We'll take a look at what is a cache hit ratio, how to improve your cache hit ratio, and we'll explore why it is a balancing act. And then finally, we'll take a look at my exam tips. So what is cache hit ratio? Well, it

represents the proportion of requests that are served directly from the CloudFront cache. So requests which can be successfully served by CloudFront are known as a cache hit. However, if you request an object, which is not already cached in CloudFront, then that is known as a cache miss. And when you experience a cache miss, that means that CloudFront will need to go to your origin and request the object or the HTML page that you are asking for. And ideally you want the majority of your requests to be served by CloudFront. For example, you might want to target a cache hit ratio of 80% or greater, and that's going to give you great performance for the majority of the requests coming into your website. So how can you improve your cache hit ratio? Well, by default, each object held in the cache is going to automatically expire after 24 hours. And that is known as the time-to-live. However, this is configurable; so you can configure CloudFront to keep the files in the cache for a longer period of time, up to a maximum of 1 year. So that's quite a long time, but by extending the time-to-live, this can give you an improved cache hit ratio because if the files are stored in the cache for longer then they are more likely to be served directly from the cache, instead of needing to go all the way to the origin to fetch the objects. Now, it's a bit of a balancing act to get it right. Because if we increase the time-to-live for our objects, let's say we set it to 1 month, for example, this is going to improve our cache hit ratio and it will increase the performance of our website because a longer duration means that your users get better performance because when the files remain in the cache for longer, it is more likely that the files can be served directly from the edge location cache. However, if your content is frequently updated-- let's say your webpage is getting updated on a daily basis with a time-to-live of one month-- your users will not always see the latest version of your webpages. Whereas with a low TTL, this reduces the cache hit ratio and it will decrease the performance relative to having a higher TTL. So let's say we keep to the default time-to-live of 1 day. A shorter cache duration means that CloudFront will send requests to the origin more frequently. And this is great, if you want to determine whether or not an object has changed and to obtain the latest version. And this is great if you have content which changes frequently. However, do be aware that this can decrease the performance relative to having a higher time-to-live. So onto my exam tips, just remember that the cache hit ratio represents the proportion of requests that are served directly from the CloudFront cache. And you can influence the cache hit ratio by changing your time-to-live. So if you increase the time-to-live for objects in the cache, then you can increase your cache hit ratio and the maximum time to live is 1 year. However, do remember it is a bit of a balancing act. So experiment to find the right balance for your own application. And this will all depend on how frequently your content is changing; how important it is that your users get the latest version, and how important the performance of your website is. So that's it for this lecture. If you have any questions, please let me know. Otherwise, I will see you in the next lecture. Thank you.

Advanced CloudFront

Hello Cloud Gurus, and welcome to this lecture, which is going to cover some of the advanced features of CloudFront. We'll start off by looking at how you can forward cookies in CloudFront, Forwarding HTTP request headers, and using your own domain name, and then finally, we will cover my exam tips. Now you can use an origin request policy to configure CloudFront to forward cookies and to serve different versions of your objects based on the cookie values. So imagine we've got some objects in our S3 bucket, and we are using CloudFront as well. Let's say we've got 3 different versions of our files, or of our objects, and we want to serve a different version of our object based on the value of a cookie. Well, we can create an origin request policy in our CloudFront distribution, which will allow us to serve these different versions of our objects based

on the value of the cookie included in each HTTP request. And you can also forward HTTP request headers and configure CloudFront to serve separate versions of your objects based on the request header values as well. So here's our S3 bucket and our CloudFront distribution. We've got 3 versions of our objects, and once again, we use the origin request policy to serve a different version of the object based on the contents of the HTTP request header. And this is how it looks in the console. So this is our origin request policy. And then down here, this is where we can configure the headers and the cookies that we are going to include. And then if you would like to use your own domain name with CloudFront, there's a couple of options, and the first one, and I think the easiest one, is to use Route 53 to create an alias record which resolves to your CloudFront distribution. So here's our CloudFront distribution, and we can create a Route 53 alias that will resolve all requests for our own domain name. Let's say we own a domain name called myawesomewebsite.com, we can use the Route 53 alias to resolve our domain name to our CloudFront distribution domain name. And the other way that you can do this is in the CloudFront distribution settings. So we can specify an alternate domain name in CloudFront, but when you do that, you must provide a custom SSL certificate to prove that you are authorized to use that domain, and it's down here at the bottom. So you should provide the custom SSL certificate if you want your users to access the content using your alternate domain name. And you can select a certificate, either stored in AWS Certificate Manager in the us-east-1 region, or you can use a certificate that is stored in IAM. But personally, I think it's much easier to just set this up using Route 53, but you do have these 2 options. So on to my exam tips. Just remember, you can use an origin request policy to configure CloudFront to forward cookies and serve different versions of your content based on the cookie values. And you can also use an origin request policy to forward HTTP request headers and to configure CloudFront to serve separate versions based on the request header values. And then finally, when it comes to using custom domain names, you've got 2 choices: you can create a Route 53 alias record which resolves your CloudFront distribution, so we can set up an alias record for our own domain name and configure it to resolve to the domain name of your CloudFront distribution, or of course, you can configure it in the CloudFront console as well, and when you do that, you will need to provide a custom SSL certificate to prove that you are authorized to use that domain name. So that's it for this lecture. If you have any questions, please let me know. Otherwise, I will see you in the next lecture. Thank you.

Improving CloudFront Cache Hit Ratio When Forwarding Request Headers and Cookies

Hello, Cloud Gurus, and welcome to this lecture, which is going to discuss how you can improve your CloudFront cache hit ratio when you are forwarding headers and cookies, and we'll begin with a high-level example of forwarding HTTP request headers or cookie values, and we'll then cover 2 different ways to improve your cache hit ratio. The first is making sure you are forwarding specific cookies only, and the second is all about forwarding specific HTTP request headers only and then we will wrap up with my exam tips. Now, just imagine that you have an architecture like this where we are serving content from an S3 bucket and all user requests are coming through CloudFront. We have multiple versions of our objects and CloudFront is forwarding the request header or the value of a cookie to determine which version of an object is going to be served. So far, so good. However, when you are forwarding cookies and HTTP request headers, you are often sending the request to your origin instead of serving the content directly from your CloudFront cache, and this is going to negatively impact your cache hit ratio. So if you are using CloudFront to respond based on HTTP

request headers or cookie values, then you can improve your cache hit ratio by restricting what you forward and not just forwarding every HTTP header and cookie indiscriminately. So how does it work? Well, imagine that every incoming request contains 2 cookies. Each cookie has 1 of 3 possible values. For each request, CloudFront forwards the cookies to your origin to identify the correct version of the content to serve, but what if only one of these cookies determines which content to serve? What if only one of these cookies is actually relevant? In this situation, CloudFront will end up forwarding more requests to your origin than is really necessary, and this unnecessarily fills up the cache with multiple versions of the same object and that is bad news for our cache hit ratio. So if you want CloudFront to return a version of your object based on a specific cookie, then only forward that cookie and this will avoid filling up the cache unnecessarily and it will help to optimize our cache hit ratio. Now, unsurprisingly, the same principle relates to caching based on HTTP request headers. So let's take a look at an example of an HTTP request and the request headers contain an awful lot of information about the request itself and about the browser that was used to make the request. Now, in order to optimize your cache hit ratio, you should configure CloudFront to forward and cache only the specified headers that are relevant. For example, you might want to forward the accept-language header, which states the languages that the client understands, then you could serve different versions of your content based on your client's preferred language, and you should avoid forwarding all headers indiscriminately because for the headers that you specify, CloudFront is going to forward every combination of header name and value that it receives, and it will cache every response from your origin, even if those responses are identical. So this is another way to end up with multiple versions of the same thing in your cache. It can fill up your cache, and you've guessed it, that is bad news for our cache hit ratio. So if you want CloudFront to return a version of your object based on a specific HTTP request header, then only forward that specific header, and this is going to avoid filling up the cache unnecessarily and it will help us to optimize our cache hit ratio. So for the exam, the one thing that I want you to take away is that when you are forwarding HTTP request headers, less is more, so only forward what is necessary, and this will avoid filling up your cache unnecessarily and will help you to optimize your CloudFront cache hit ratio, and if we forward all cookies and headers indiscriminately, then CloudFront can end up caching multiple versions of identical content to cater for all the different header or cookie combinations, whether they are relevant or not. So that is it for this lecture. If you have any questions, please let me know. Otherwise, I'll see you in the next lecture. Thank you.

Section Review: Networking and Content Delivery Summary - Part 1

Hello Cloud Gurus, and welcome to this lecture, which is the summary of the Networking and Content Delivery section, including my exam tips. And we'll begin with VPCs. And I want you to think of a VPC as a logical data center in AWS, consisting of internet gateways, sometimes virtual private gateways, route tables, network access control lists, subnets, and security groups as well. And remember, 1 subnet is always in 1 Availability Zone, and a subnet cannot span across multiple Availability Zones. And if you remember in our demo, we created our own custom VPC. We configured a public subnet with its own security group and route table, and a route out to the internet using an internet gateway. We also created a private subnet with its own security group and its own route table. And we used a NAT gateway to allow for outbound access to the internet. We configured 1 of our EC2 instances in our public subnet as a bastion host, and from there, we were able to SSH onto our private instance. Moving on to network access control lists. And when we create a custom VPC, it automatically comes with a default network access control list which allows

inbound and outbound traffic. And you can create your own custom network ACLs. And by default, each custom network ACL will deny all inbound and outbound traffic until you go in and add some rules. And you can use a network access control list to block specific IP addresses, and if you would like to block an IP address, then you should do that using a network ACL, and not a security group. When it comes to subnet associations, a subnet can be associated with only 1 network ACL at a time. And a network ACL consists of a numbered list of rules, and these rules are evaluated in order, beginning with the lowest numbered rule first. You will need to create separate inbound and outbound rules, and each rule can either allow or deny traffic. And finally, network access control lists are stateless. So that means that you must configure both the inbound and outbound rules, unlike security groups, which are, of course, stateful, and what I mean by that is that you do not need to configure both the inbound and outbound rule when you are configuring a security group rule. On to VPC endpoints, and a VPC endpoint allows you to establish a private connection between your VPC and supported AWS services, and it uses a technology called AWS PrivateLink. And this means that the traffic between your VPC and the AWS service does not leave the Amazon network and is never exposed to the internet. And just remember that after you have created the endpoint, just go in and make sure that a route to the service is present in your routing table, and you will need to route any requests to the given service via the VPC endpoint that you have created. On to Systems Manager Session Manager, and of course, Session Manager enables secure remote login to your EC2 instances. It is browser based, and you get an interactive session using Powershell or Bash. You can access it using the AWS console, and it also supports the AWS CLI and SDK. And the great thing about Session Manager is that it is a single solution. So you can manage your Windows and Linux instances as well as on-premises physical or virtual hosts as well. So it's just one single solution for remote login. You don't need RDP or SSH, and there are no SSH keys or bastion hosts to manage either. And finally, it is super secure. So it is secured using TLS encryption, and you can send your session logs into CloudWatch. Moving on to VPC flow logs, and VPC flow logs capture information about the IP traffic going to and from the network interfaces in your VPC, and you can configure your flow log to record accepted traffic, rejected traffic, or all traffic. The log destination can either be CloudWatch or S3, and if you remember, in the demo, we configured our EC2 instance to send flow logs into CloudWatch. And VPC flow logs are great for troubleshooting network connectivity issues, as well as understanding the flow of network traffic to and from your VPC. And it's important to remember what a flow log record looks like. So here's our source IP address. Here's our destination IP address. This is our destination port and 3389 is RDP. And in this case, our traffic was rejected. And in this case, our traffic was rejected. On to VPC Peering, and VPC peering allows you to route traffic between 2 VPCs using private IP addresses. So it allows EC2 instances that are located in one VPC to communicate with instances in another VPC, and the instances will behave as though they were on the same private network. And for that reason, they must not be using the same IP address space. So your 2 VPCs must have a different CIRD range. And the great thing about VPC peering is that the VPCs do not even need to be in the same AWS account or in the same AWS Region. On to site-to-site VPNs, and VPN provides a secure, encrypted connection between your on-premises networks, remote offices, and client devices, and your AWS resources. You'll need a virtual private gateway configured in your VPC, and a customer gateway, which you need to configure in your on-premises network to establish the site to site VPN, and you get a secure, encrypted tunnel between your network and AWS, and it actually uses the industry standard IPsec protocol. And then finally, we have Direct Connect, and it's really important that you understand Direct Connect at a high level. So Direct Connect provides you with a dedicated private connection, and it directly connects your data center to AWS without using the

internet. And it's great for high throughput workloads with lots of network traffic, and you can select the network speed that is suitable for your organization. For example, 1 GB per second, 10 GB, or 100 GB per second. For example, 1 GB per second, 10, or 100 GB per second. And Direct Connect is secure and stable. So it provides a stable, reliable, consistent, and secure connection to your VPC. So if you are using a site-to-site VPN at the moment, and you're finding that your network connection is not as consistent as you would like, then consider Direct Connect as an alternative. So that's it for Part 1 of this section summary. If you have any questions, please let me know. And if you're ready to move on to Part 2, I will see you in the next lecture. Thank you.

Section Review: Networking and Content Delivery Summary - Part 2

Hello Cloud Gurus and welcome to this lesson, which is part 2 of our networking and content delivery summary, beginning with DNS. And just remember that DNS, or the domain name service, is what we use to convert human friendly domain names like acloud.guru into IP addresses like 82.124.53.1. The top-level domain is the last word in the domain name, for example .com, .edu, .gov, .guru, or .org, etc. And domain names are unique, and they are registered using a domain registrar like GoDaddy, Enom, or even AWS. The Route 53 Resolver resolves DNS queries to resources in your VPC, for example EC2 instances, elastic load balancers, and DNS records in private hosted zones in your VPC. For resources that are outside your VPC, so for DNS queries to other domain names outside your VPC, Route 53 Resolver will perform a recursive lookup against public name servers, for example if you need to resolve a domain name outside your VPC, like acloud.guru. And you can integrate Route 53 Resolver with any other DNS resolver on any other network that is reachable from your VPC, for example another peered VPC or an on-premises network that you can reach. Now when you register a domain name and Route 53 creates your hosted zone for you, it's going to automatically create the start of authority record. And remember, that's going provide important basic information about the domain like the email address of the administrator of the domain. It also includes an NS record. And of course, these are the name server records. So these are the DNS servers that are going to be authoritative for your domain, and they're going to resolve the domain names for your domain. And then another important DNS record type that you'll need to be aware of is A records. And of course, the A in A record stands for address, and these are used to route traffic to a resource like a web server using its IPv4 public address. Next, we've got aliases and Z names, and it's important to understand the differences. Now aliases allow you to map one DNS name to another. Let's say you wanted to map example.com to the domain name of your load balancer. Well you can do that using an alias. And you can create an alias record with the same name as your hosted zone, and it's also known as the zone apex or as a naked domain name, for example acloud.guru. Whereas, with a CNAME, this is used to resolve one domain name to another. For example, if you wanted m.acloud.guru and mobile.acloud.guru to resolve to the same address, then you can do that with a CNAME. And you cannot create a CNAME record with the same name as your hosted zone, also known as the zone apex or the naked domain name like acloud.guru. And for that, you will need to use an alias record. And generally, given the choice, and especially in the exam given the choice, go for alias over CNAME. Onto the different routing policies that are available with Route 53. And first of all, we've got the simple routing policy. Great if you want to route traffic to a single resource, for instance a web server for your website. Then we've got Route 53 routing policy, which is great for active/passive failover. Geolocation is the one to use if you want to route traffic based on the location of your users. Geoproximity allows you to route based on the physical distance between your users and resources,

and it also allows you to configure a bias to route more or less traffic to a given resource. Latency-based routing is great if you want to route traffic based on the best latency and provide the best performance for your users. There's multivalue answer if you want Route 53 to respond with up to eight healthy records selected at random. And then finally, we've got the weighted routing policy. Great if you want to route traffic to multiple resources based on a numerical weight that you specify, for example routing the majority of your traffic to the region where you've got the most resources. Moving on to CloudFront. And just remember the CloudFront terminology. So we've got edge location, and this is the location where content will be cached, and it's separate to an AWS region or availability zone. CloudFront origin. The origin is the origin of all the files that the CloudFront distribution will serve, and this can be an S3 bucket, an EC2 instance, an elastic load balancer, or even a Route 53 domain name. And then the distribution is the origin and configuration settings for the content that you're going to distribute using CloudFront. As you know, CloudFront can help speed up the delivery of your static content to viewers all around the world. Origin access control restricts access to the contents of your bucket so that all users must use the CloudFront URL instead of the direct S3 URL. And this allows you to remove public access from your bucket configuration. CloudFront logs are stored in S3, and they're known as standard logs or access logs. What gets logged? Well, it's the detailed request information, so things like request time, edge location, client IP, and method, etc. The logs usually appear within 1 hour, and standard logs are delivered at several times an hour. And because of that, you can end up with an awful lot of data and a lot of different files to review. And if you do have lots of different files to review, then the recommendation is that you consolidate them all into one. So combine the logs that you want to analyze into one file, and then you can easily analyze the data using Athena. Remember that edge locations are not just read-only, and you can write to them as well. For example, you can put an object onto them. Objects are cache for the time to live, and you can go in and clear cached objects if you want to by creating an invalidation. But do be aware you will be charged for doing that. And then finally, just remember that CloudFront is used for improving performance, optimizing performance for users, accessing your website from different regions all around the world. Moving on to CloudFront errors, and it's important to understand the difference between 400 and 500 errors. So just remember that 400 errors are always client-side errors. And I just try to remember the 404 error, and that's the one that you see when you're trying to access a web page that doesn't exist. And you see the message error, page not found. So if you do see a 400 error, just check that the requested object exists, that you've got access to it, and that your request is correctly formed. And for some reason, the 403 error does come up an awful lot in the exam. So just remember that that one is access denied. Whereas with 500 errors, I try to remember that if it's a 500 error, this is always a server-side error. And the way I remember it is that a 5 looks a little bit like an S if I take my glasses off. And 500 errors are generally caused by the server not responding. Usually, it's something to do with high traffic coming to your website, and your servers getting overwhelmed or unavailable. And it's usually indicative of a capacity or performance issue. Onto CloudFront cache hit ratio. And remember, the cache hit ratio relates to the proportion of requests that are served directly from the CloudFront cache, and you can influence the cache hit ratio by changing your time to live. So by increasing the time to live for objects in the cache, you can increase your cache hit ratio. And the maximum time to live is 1 year. But do remember that it is a balancing act to get it right, and you'll need to experiment to find the right balance for your own application. And just think about how frequently your content is changing and what is an appropriate time. to live. For advanced CloudFront, remember that we used the origin request policy to configure CloudFront to forward cookies, and we can serve different versions of our objects based on the cookie values. And we can

also use the origin requests policy to forward HTTP request header as well. And once again, use CloudFront to serve different versions of our objects based on the request header values. And if you want to use your own custom domain name with CloudFront, you can create a Route 53 alias record, which resolves to your CloudFront distribution. And that is a really easy way to do it. Alternatively, you could configure the custom domain name within the CloudFront console itself. And when you do that, you will need to provide an SSL certificate demonstrating that you are authorized to use that domain name. And if you are forwarding HTTP headers or cookies and you need to improve your cache hit ratio, just remember that less is more, and you should only forward what is absolutely necessary. And this is going to avoid filling up the cache unnecessarily and help optimize your cache hit ratio because if we forward all of the cookies and headers indiscriminately, then CloudFront can end up cache multiple versions of identical content just to cater for all of the different header and cookie combinations, filling up your cache unnecessarily. So that is it for this lesson. Any questions, please let me know. Otherwise, we will see you in the next section of the course. Thank you.

Cost and Performance Optimization

Section Introduction

Hello Cloud Gurus, and welcome back to this final section introduction on cost and performance optimization. In this section, we'll cover several topics which will help you better understand how to optimize your applications and architectures to be cost and performance-effective. We'll do this through understanding several topics. We'll talk through AWS Cost Explorer and cost allocation tags. We'll also discuss AWS tags and resource groups. We'll discuss how to improve your cost using AWS Budgets as well as get a hands-on demo with creating billing alarms. We'll talk through reducing your costs also through managed services, understanding AWS Compute Optimizer, as well as using metrics to improve performance and cost. We'll wrap up this section with a conclusion and some exam tips. So if you're ready to get started, join me in the next lecture.

AWS Cost Explorer and Cost Allocation Tags

Hello Cloud Gurus and welcome to this lesson on AWS Cost Explorer and cost allocation tags. In this lesson, we'll introduce AWS Cost Explorer and discuss some key information regarding tagging and cost allocation tags. We'll answer the question, what are tags, and also discuss how tags can be used to track cost using cost allocation tags. We'll review the AWS Cost Explorer, the reporting it provides. We'll then walk through the Cost Explorer service and conclude with some exam tips. So what are tags? Well, tags are key value pairs attached to AWS resources. So, for example, for each of my EC2 instances, I can create key value pairs to help describe the instance so that I can more easily track the resources that are being created in my account. For example, I can give my instance a createdBy tag that lets me know who created the resource. And this first part here is the key, and the second part here is the value. I could then create a tag to let me know which cost center this resource belongs to. I could also create a tag to let me know which cost center this resource belongs to or which stack this resource belongs to. And maybe you've got a test stack and a production stack, and it's really useful to tag your instances and your AWS resources so that you can understand who they belong to or what project or department they belong to. So that is our basic tags. Cost allocation tags, on the other hand, are simply tags that are used to help you track your AWS costs on a detailed level, and we've got two different types of tags. First of all, there's the user-defined cost

allocation tags, and these are cost allocation tags that are created, defined, and applied to your resources by you. On the other hand, we have AWS-generated cost allocation tags, and these are cost allocation tags that are created, defined, and applied to your resources by AWS for supported resources only. Moving on to Cost Explorer. And Cost Explorer allows you to visualize, view, and analyze your AWS usage costs, and this is what it looks like. And the connection between cost allocation tags and Cost Explorer is that using Cost Explorer, you can visualize, view, and analyze your AWS usage cost by simply filtering on the tags that you've created. And AWS Cost Explorer provides some pretty incredible reporting. And first of all, it gives us the ability to report on our cost and usage. And for our cost and usage, these are reports on aggregated costs across all AWS services. Now you can also create reports for any AWS savings plans that you've created, and these are simply reports on the utilization and coverage of your savings plan. And then the same is also true for reservations. So for your reservations, you can create a report reporting on the utilization and coverage of your reservations. So now, let's head over to the AWS console, and I'll show you Cost Explorer first hand. Now if you would like to explore your own costs, then you will need to do that in your own AWS account because in our cloud sandbox or cloud playground environment, we do not give you permission to use Cost Explorer. However, all of the concepts are pretty simple. And if you like, you can just watch as I talk you through everything. So from the management console, and this is my own AWS account, first of all, I need to activate my cost allocation tags. Up in the right-hand corner under my account name, I'm going to select the down arrow and then select the Billing Dashboard. Then in the left-hand menu, I'm going to select Cost allocation tags. Here's the user-defined cost allocation tags, and these are tags that have been defined for the different resources that I've created in this account. So these are all my own tags that I've created. And then if we select AWS-generated cost allocation tags, these are the cost allocation tags that have been generated by AWS services. Now keep in mind that these cost allocation tags must be activated. And to do that, I'll select them all here and select Activate. And these are all now live and showing a status of Active. And I'm going to do the same for my user-defined cost allocation tags. So I'll select them all here and select Activate. So now, let's head over to Cost Explorer. In the search box, I'm going to search for Cost Explorer. It's going to take me to a home page, and I'm going to select Cost Explorer from the left-hand menu. Next, select Reports. And now if I click on Create new report here are the different types of reports that we discussed earlier. So there's a cost and usage report, which is the aggregated costs across all of our AWS services. Down here is our savings plan reports. So these cover utilization and coverage of our savings plans. And here's our reservation reports. Now I'm not going to create a report right now. Instead, I'm going to head back over to Cost Explorer. Select it on the left. And then, using the report parameters on the right-hand side, I can set any of these report parameters, and it will reflect on the report generated on the left. So here, I can set the date range that I would like to see. And at the moment, it's displaying the last 6 months, and the granularity is monthly. But I could set it to hourly or daily. Under Dimensions, I can group the data by any of these dimensions, like region, instance type, resource, by tag, availability zone. But I'm going to keep it to service, and these are all the different services that are being used in my account at the moment. Down here, we can apply filters, and there are loads of different filters to choose from. But the one I want to show you is right down here because we can filter based on our tags. And if I open the drop-down, I can see all the tags that are available to me. And keep in mind, you must tag your resources in order for the tags to properly populate here in Cost Explorer. And let's say I want to find the cost of all the resources created by a particular user in the last 6 months. I can select the tag of aws:createdBy, so that's my tag. Next, I'll select the value, which is going to be my user. I'll select a user and Apply. And there we are. We can see the cost of my resources over the

past few months. And we can see that in December and January, I was using RDS quite a lot in the this account. And then if we scroll down, we can see all the different services that I've been using over the past 6 months. But then, if I clear up my tag filter, it will go back to displaying the total cost for the entire account. So we've got the complete breakdown of month over month, as well as which services created the cost. And then right at the top, we can also download this as a CSV file. So that is Cost Explorer. And for the exam, keep in mind that Cost Explorer allows you to visualize, view, and analyze your AWS usage cost. And one of the ways to filter within Cost Explorer is through the use of tags. And tags are, of course, key value pairs that are used to mark and track AWS resources. And using tags to describe your resources is a best practice within AWS. There's also cost allocation tags, which must be enabled through the AWS console in order to track your AWS resource costs. And lastly, Cost Explorer reporting allows you to create reports based on cost, usage, savings plans, and reservations. So that is all for this lesson. Any questions, please let me know. But if you're ready to move on, please join us for the next lesson. Thank you.

Improving Costs with AWS Budgets

Hello Cloud Gurus and welcome back to this lecture on improving cost with AWS Budgets. In this lecture, we'll discuss how you can improve your overall AWS cost using AWS Budgets. We'll discuss this by talking about the services and features of AWS Budgets, as well as the type of budgets that are available for you to configure in the service. We'll also discuss cost budgets and the filters you can use to get really granular with your budgets, as well as usage types and usage groups that'll be key to understanding budgets. We'll talk through savings plans as well as utilization and coverage reservations. We'll also discuss how you can enable alerting within AWS Budgets as well. Finally, we'll wrap up with some exam tips. So let's go ahead and get started. AWS Budgets allow us to create budgets for our AWS resources. And by that, within the service, we're able to first create and manage budgets. We're also able to refine budgets and get really granular by understanding where our cost coming from, as well as adding notifications to your budget which may be alerting through simple notification service, SNS, and more. Within AWS Budgets, there's several type of budgets you can configure to help you improve your AWS cost. You can first get started by creating a cost budget. And a cost budget allows you to track your AWS cost against the specified dollar amount. For example, if you may have budgeted $10, 000 for your AWS accounts for a specific month, this will be the perfect place to track your cost. You can also use a usage budget. And a usage budget allows you to monitor your usage of one or more specified usage types or usage type groups, and we'll discuss this in greater detail in just a moment. You can also create a savings plan budget. And the savings plan is where you prepay for compute. And with AWS Budget, you're able to track the utilization as well as the coverage associated with your prenegotiated savings plans. And the same works similarly with the reservation budget. And with the reservation, you can monitor the utilization or the coverage of your reservations associated with your reservations for specified services. So for example, if you use reserved EC2 instances, AWS Budgets is great in tracking your budget as it relates to those reserved instances. With creating cost budgets, you can add filters in order to make the cost budget as granular as you'd like. Let's briefly discuss the type of filters that are available to you when creating a cost budget. You can first create a cost budget to filter by service. So let's say for example, you'd like to budget based on your EC2 usage or your AWS Lambda usage. That's available through using the service filter. You can also track budgeting by the accounts linked to your AWS account. Keep in mind that AWS Budgets and budgeting must be configured in the master account, so all linked accounts that you select can be budgeted for. You

can also budget by resources in a specific region to help you understand your regional AWS cost. You can also budget by instance type for your EC2 instances. Perhaps you have instances that are specific to certain applications you may be running. Instance type will be very useful for that. You can also budget by usage type. So let's say how many gigabytes a certain service is using or how many seconds your Lambda is running. Through using tagging, you can create a budget based on a cost category, as well as different tags that you may have included within your services. You can also budget based on API operation as well as availability zone, the different purchasing options that are in your account, and the billing entity. For creating budgets for usage within AWS Budgets, you're able to create a budget by usage types or usage type groups. So what are usage types? Usage types are the units that a service uses to measure the usage of a particular type of resource. So for example, for EC2 instances, the usage type would be hours. For Lambda's, the usage type would be seconds. And for CloudWatch, a usage type that can be used is request. Usage type groups on the other hand, collect a specific category of usage type filters into 1 filter. So for example, with EC2 instances, there's several metrics that goes into creating the EC2 running hours usage type groups. And the same with DynamoDB, for measuring the provision throughput capacity which that can be both read and write being measured under that usage type group. For S3, there's a standard storage usage type group that's available. And for RDS, there's also a running hours usage type group and there's several more. With creating a savings plan budget, the saving plans budget allows you to track your savings plan usage as well as allotments. And a savings plan is pretty much a pricing model allowing discounts on committed usage for EC2 and Fargate. And through the AWS Budgets saving plan budget, you can track the utilization of your savings plans and this measures unused and underutilized savings plans. You can also measure the coverage of your savings plans. And this is useful in measuring how much instance usage is covered by your current savings plans. With reservations budgets, this operates similar to the savings plans budget. But for reservations, a reservation allows you to reserve compute capacity for EC2 instances in an availability zone or a specified duration. And for reservation budgets, you can also measure the utilization of your reservation, which measures unused and underutilized reservations. And you can also allow your budget to measure the coverage of your reservation, which measures how much instance usage is covered by your reservation. Through AWS Budgets, you can also configure alerting to let you know when you may be nearing a budget threshold. Alerts are configured to specific thresholds. You can either select an absolute value or a percentage of your budgeted amount. Alerts can also be triggered based on an actual budget amount or forecasted using machine learning. And alerts are sent either via email or using an SNS topic. So what are our exam tips for improving costs with AWS Budgets? The first thing you'll need to remember is AWS Budget allows you to create and manage budgets to improve cost for account usage. It's also useful to know that AWS Budget types. AWS Budgets allow you to create budgets based on cost, usage, reservations, and savings plans. And lastly, you're able to configure budget alerts. And budget alerts are sent via email and through SNS topics. That's all for this lecture. If you're ready to move on, join me in the next lesson.

Demo: Creating an AWS Budget and Billing Alarm

Hello Cloud Gurus and welcome to this demo on creating AWS budgets and billing alarms. And in the this demo, we'll walk you through the steps to help you configure your AWS budget and billing alarms within your personal account. So to complete these steps, you will need to be in your own personal AWS account, and you will not be able to do this in our cloud sandbox or playground. Now first of all, we'll enable billing alerting through the My Billing dashboard. Next, we'll

configure an AWS budget. We'll create a billing alarm in CloudWatch. We'll confirm that we've successfully created our billing alarm by receiving the AWS confirmation email. And as usual, we'll finish off with some exam tips. So if you are ready, please join me in the AWS console in your own personal account. So from the console, the first thing I'm going to do is enable billing alerts, and that can be done through the billing dashboard. So in the upper right hand of the management console, select Billing Dashboard and then scroll down to Billing preferences on the left. And we're going to check this box here that says Receive Billing Alerts. And this will allow you to receive billing alerts to the email address that is associated with this AWS account. And now go ahead and select Save preferences. Next, search for AWS Budgets and Create a budget. I'll use a template to create my budget. And down here, you can go ahead and select a template that best matches your use case. There's a zero spend budget, which is going to notify you if your spending exceeds the AWS free tier limits. There's a monthly cost budget and budgets that notify you about your daily savings plan coverage and daily reservation utilizations as well. However, today we're going to create a monthly cost budget. So select the Monthly cost budget template. Here's our budget name. And by default, it's given us a budgeted amount of $100, and I'm going to change that to 10. And then up here, it's just trying to fetch some data relating to my last bill, but there's just nothing in here because this is a brand new account. And then finally, I need to provide the email address that I would like the notifications to be sent to, so I'll enter my email address in the there. And then at the bottom, select Create budget. And now, if I select my budget, I can see information relating to my budget health. Here's my current spend versus budgeted. And as I use more AWS services in this account, I can expect to see more information appearing in here. Next, let's go ahead and create some billing alarms in CloudWatch. So search for CloudWatch. On the left-hand side under Alarms, select Billing. Select Create alarm on the right-hand side. I need to select my metric. And then down here, select Billing. And then I have a choice to select billing by service or by total estimated charge. So if I select By Service, because this is a new account, I've only got one service available in here. But as I use more services, then more are going to become available in here. So instead, I'm going to select Total Estimated Charge. I'll select US dollar as my currency and Select metric. Now that I've selected my metric, I can now add my conditions down here. So I can select whether I want my metric to use a static value as a threshold or use anomaly detection. So this is going to use a band or a range as a threshold. Our alarm condition is that I want the alarm to be triggered whenever the estimated charges is greater than my threshold. And let's say I wanted to set it for 120. So I'm going to be notified anytime I go over the $100 that I set for my budget when we configured our AWS budget. So for now, hit Next. And this is where we can configure our notification. So a notification is going to be generated using an SNS topic, and we'll go ahead and create a new topic. I'm going to accept the default name and add my email address in here. And this is the email address that is going to receive the notification. So then go ahead and Create topic. So now that my topic has been created, I can also add actions using an auto scaling action, EC2 action, or a Systems Manager action. I'm going to bypass those for now and click Next. And now I need to give my alarm a name, and I'll call it ACG-Test-CloudWatch-Billing-Alarm. Then hit Next. Scroll down and Create alarm. And now that my alarm has been created, I should also receive a confirmation email in my email account to begin receiving the SNS messages for this billing alarm. So I'll head over to my email, and it's actually ended up in my Spam folder. So if it's not in your main email folder, then do check your Spam folder because mine has been marked as spam. So here is my confirmation of my subscription to the SNS topic, and I need to select Confirm subscription. And that has now let me know that my subscription has successfully confirmed. So now back in the AWS console, within my CloudWatch billing alarm, I can see that under Actions I have a warning. And if I click on here,

I can see that it's because of my SNS topic endpoint is pending confirmation. So if I select Refresh, that warning message should go away because it's automatically confirmed. So now we know that our billing alarm is active. And for the exam, keep in mind that billing alarms are created in CloudWatch, and we're able to create alarms that are based on a few key metrics, for instance account costs, service costs, and total estimated charges. Using AWS Budgets, we are able to create budgets based on cost, usage, and reservations or savings plans. So that is all for this demo. And if you're ready to move on, please join us for the next lesson. Thank you.

Reducing Cost through Managed Services

Hello, Cloud Gurus, and welcome back to this lecture on Reducing Cost through Managed Services. In this lecture, we'll discuss how you can identify opportunities to use managed services to reduce your costs within your AWS account. We'll do this through discussing and defining managed services, as well as talking through several examples of managed services, including AWS Fargate, AWS' Elastic File Service, or EFS, as well as AWS's Relational Database Service, and that's RDS. We'll also conclude with some exam tips. Let's go ahead and get started. Let's start by talking through managed services. What are managed services? They're services which AWS is responsible for the managing of the infrastructure and software updates. And when we talk about the managing of the infrastructure, we mean that the services natively handle capacity through elasticity and scalability. And these services are AWS Fargate, which Fargate is a serverless compute engine for containers that works both with Amazon Elastic Container Service, as well as Amazon Elastic Kubernetes Service. And Fargate removes the need to provision and manage services. More importantly, allowing us to pay for resources per application. It also improves the security of our application through isolation. Another service that is also serverless and 'set and forget' is Amazon's Elastic File Storage, or EFS. And EFS is a file system which allows us to be able to grow and shrink automatically as we add and remove files. And this ability allows us to be able to easily provision our capacity to accommodate growth, which helps us to only pay for what we use. Another service would be Amazon's RDS service and with the RDS, or Relational Database Service, we can easily set up and operate a relational database, which provides us cost-efficient resizing capacity, which automatically allows us to provision database setup and patching, as well as backups. And the overall goal of these services is that using managed services within AWS increases our performance while reducing our support and costs. Now let's talk about how AWS Fargate reduces your costs. AWS Fargate saves you the need to have EC2 instances running for use by ECS, as Fargate offers you a managed service for running your containers. With AWS Fargate, you're able to first build your container image, which is great for streamlining your deployment pipeline, and then you're able to define your compute resources. After your compute and memory resources are defined, you're then able to run your applications. After running your applications, you only pay for the compute that you've used. Now, if you were using EC2 instances, you would have to pay for the instance size you use. And so this is the difference between Fargate and using a traditional EC2 instance, is that using AWS Fargate, your build in a serverless more container-friendly pay-as-you-go type way. Now, let's talk about Elastic File Service. So with Elastic File Service, it offers high availability and durability, and data written to EFS is written in 3 Availability Zones, and is accessible through all AZs within the Region, offering 99.99 % availability. It is also elastic and scalable, which EFS only charges for the capacity used and also scales to meet storage and throughput capacity. It's also great for container and serverless file storage and support, which EFS integrates with multiple container services, as well as Lambda and EC2. EFS also integrates with

AWS Backup for automatic backups. And last but not least, EFS offers storage classes and lifecycle management. EFS offers 4 storage classes: EFS Standard, Infrequent Access, or IA, One Zone, or One Zone-IA, or Infrequently Accessed. Lifecycle management moves files based on usage patterns through lifecycle policies. So, the scalability and storage availability within EFS offers us many cost reduction savings. Last but not least, Relational Database Service offers us several solutions which would help us reduce costs. First, starting with Aurora Serverless, which is also a serverless database compute. It is an auto-scaling version of Aurora which automatically scales capacity up and down, which meets the needs of our application. We can also use RDS Storage Auto Scaling, which is helpful in reducing our costs, allowing your RDS storage to grow along with our data. And lastly, we have our Reserved Database Instances, and through Reserved Database Instance plans, we can purchase compute up front for a reduced cost. So, for our exam tips for reducing costs through managed services, keep in mind what are managed services, and they're services which AWS is responsible for the managing of the infrastructure and software updates. And we talked about elasticity and scalability, as well as maintenance and software updates. And these services, just to name a few, are AWS Fargate, and we talked about the elasticity and scalability through offering containerized versions of your compute. We also discussed Amazon Elastic File Storage and how it also allows us to grow and shrink based on our file additions and subtractions. And lastly, we talked about Amazon's Relational Database Service, or RDS, which gives us several services and features including Aurora Serverless, which we can use to reduce the cost of our database storage costs. And keep in mind, using managed services within AWS increases performance while still reducing our operational support and costs. So that's all for this lesson. If you're ready to move on, join me in the next.

AWS Compute Optimizer

Hello, Cloud Gurus, and welcome back to this lecture on AWS Compute Optimizer. In this lesson, we'll discuss the role that AWS Compute Optimizer plays in reducing your overall AWS cost, as well as improving your performance. We'll do this through discussing what is AWS Cost Optimizer and how it works. We'll talk about the compute resources that it monitors and the recommendations it provides, as well as how you can export those recommendations. And as usual, we'll wrap up with some exam tips. So let's go ahead and get started. What is AWS's Compute Optimizer? So AWS Compute Optimizer is a machine learning-based service which helps you optimize the performance of your application, as well as your overall cost within your AWS account by looking at and monitoring your compute resources. Now, let's talk about how AWS's Compute Optimizer works. Well, the first thing you have to do is be opted in to this particular service, and you can do this either in your individual account, or the management of your AWS account must configure this or opt in. After you're opted in to AWS Compute Optimizer, the machine learning component of the service does a resource analysis, and what it's looking for is first, your resource configuration, how are your resources configured, as well as is the utilization data that's coming from CloudWatch about how your resources are currently utilized. And from that, AWS's Compute Optimizer is able to provide you smart recommendations. Compute Optimizer also has cross-service integrations with S3, for example, which allows you to be able to push any of your recommendations over to an S3 bucket of your choosing, as well as integrations with Cost Explorer and Systems Manager. And after you've received your recommendations, you're then able to reconfigure your resources in hopes that you can optimize your performance and cost. Now, let's talk about the compute resources that it analyzes, as well as the recommendations that it provides. So, starting with Elastic Compute

Cloud or EC2, AWS's Compute Optimizer lets us know if our EC2 instances are optimized, or if our instances are under-provisioned or over-provisioned. And it looks at these key compute metrics on your EC2 instances, starting with your CPU, your memory, your EBS throughput and IOPS, as well as your network traffic, which is your network bandwidth, or your packets per second, and also your disk IOPS, as well as your disk throughput. For your Auto Scaling groups, Compute Optimizer lets you know if your Auto Scaling groups are optimized or not optimized through looking at your average CPU utilization, and your average network in and out. For Elastic Block Store volumes, it also lets you know if your compute is optimized or not optimized through looking at your read and write operations, as well as your read and write bandwidth. And lastly, checking out your burst balance. And finally, it also looks at your Lambda functions, letting you know if they're optimized or not through looking at the duration of your Lambda, which is in milliseconds, the count of how many errors and invocations. It also lets you know if the data within your Lambda is unavailable based on the data it has received. AWS Compute Optimizer also allows you to export recommendations. Your recommendations can be exported to S3, and the recommendations can be exported in either CSV or JSON format. So what are our exam tips for AWS Compute Optimizer? Keep in mind, that AWS Compute Optimizer is a machine learning-based recommendation service which helps you optimize the performance, as well as the cost of your AWS compute resources. And that's done in several services. It's done through EC2, your Auto Scaling groups, as well as Elastic Block Store and Lambda. Keep those in mind. And also keep in mind, that your recommendations can be exported to S3 in either CSV or JSON format. So that's all for this lecture. If you're ready to move on, join me in the next lesson.

Using Metrics to Improve Performance and Costs

Hello Cloud Gurus and welcome to this lesson on using metrics to improve performance and costs. In this lesson, we'll discuss the use of CloudWatch metrics to help you improve your application performance and lower your overall AWS usage costs. We'll review CloudWatch metrics, metrics explorer, and metrics streams. Next, we'll do a walk through to help you understand how these are used. And we'll finish off with my exam tips. Now, if you recall, CloudWatch metrics allows us to get information about our application's performance and also keep an eye on the cost of our system. Metrics appear as you create and deploy new resources in your account, and metrics can also be published from within your application. CloudWatch metrics also give you the ability to be able to graph and visualize your CloudWatch data, and you can graph your metrics from AWS services or publish them from your applications, including custom metrics that are generated by your application. CloudWatch generates automatic dashboards and allows you to create dashboard widgets from your graphs for easy data retrieval. Metrics explorer allows us to filter, aggregate, and graph metrics within the console. We first choose a metric that we'd like to explore, and this could be metrics from EC2 instances, S3, SQS, or Lambda, and all of this is applied through tagging. So we select the resource by tag, selecting cost allocation tags or any other tags that we've configured. And then from there, we're able to aggregate and graph our metric data within CloudWatch. And then, after we've created a widget from this particular graph, we can then add all of that information to a dashboard. So that is metrics explorer. Metrics streams, on the other hand, allow us to configure our metrics to stream to different AWS services. So we can stream our metric data to S3 or configure a stream to Amazon Kinesis Firehose, and this can all be done within CloudWatch where CloudWatch automatically pushes our stream to an S3 bucket of our choice or to Kinesis Firehose. So now, let's head over to the AWS console to get a first-hand look at how CloudWatch metrics can

be configured to help you improve your performance or cost. And from the console, first of all, search for CloudWatch. Then, in the left-hand pane under Metrics, select All metrics. And then here, under Metrics, the important thing to note is that I've got billing metrics here, as well as metrics for all the other AWS services that I've used in this account. And if you configure any custom metrics, they will be available down here as well. But for this lesson, let's just create a simple widget, and I'm going to create a widget to track my EC2 costs and attach it to a dashboard. So select Billing. We can either look at the total estimated charge or charge by service, and I'm going to select charge by service. And we can see that there are metrics for all the different AWS services that I've used up to this point. But right now, I'm interested in my EC2 usage. So in the search box, type EC2 and hit Enter. Here's our EC2 metric. So select that, and it should appear in the graph above. Now it's not actually showing any data, and that is because it's currently set to only show the data for the past 3 hours. So select Custom, and I want to view the data for the past 4 weeks. So select 4 weeks, and there we go. It has populated our graph. And I can see my estimated charges over the last 4 weeks. So now, let's add this graph to our dashboard. Select Actions, Add to dashboard, Create new. Call it EC2-Charges, and don't use any spaces in the name, and then select Create. You can select a widget type. I'm going to stick with a line graph. Customize the title. I'm just going to add EC2 on the end. And then Add to dashboard. So just like that, it's very simple to create a dashboard to track your estimated billing for the services that you're using. And you can use this method to also track any of the metrics that appear in CloudWatch. So now, let's wrap up with our exam tips. Keep in mind that CloudWatch metrics are data about the performance and cost of your system. Metrics explorer allows you to filter, aggregate, and graph metrics in the AWS console. And CloudWatch metrics streams allow you to configure metrics to stream to either S3 or Kinesis Firehose. Well, that's all for this lecture. Any questions, please let me know. And if you're ready to move on, we'll see you in the next one. Thank you.

Increasing S3 Transfer Speeds with Transfer Acceleration

Hello, Cloud Gurus and welcome back to this lecture on Increasing S3 Transfer Speeds with Transfer Acceleration. In this lecture, we'll talk through increasing your S3 upload speeds through the use of Transfer Acceleration by defining what is S3 Transfer Acceleration as well as the key components of Transfer Acceleration. We'll also discuss when you should use it and then we'll conclude with some exam tips. Let's go ahead and get started. What is S3 Transfer Acceleration? So simply put, S3 Transfer Acceleration allows you to quickly and securely transfer your files to and from S3 over long distances using CloudFront's Edge networking. And this enables fast, easy, and secure transfer of files over long distances between your client and your S3 bucket. And this also is known to improve speed by up to 50 to 500%. Now let's discuss the key components of S3 Transfer Acceleration. First, S3 Transfer Acceleration uses CloudFront Edge locations to accelerate transfers to and from S3. It also uses Edge locations. Instead of uploading directly to your S3 bucket, you can use a special URL to upload to an Edge location nearby which will then transfer that file to your S3 bucket. You'll get a distinct URL to upload to which looks very similar to this one provided here which is acloudguru.s3accelerate .amazonaws .com. When should you use S3 Transfer Acceleration? So S3 Transfer Acceleration is ideal when there's an S3 bucket that's far away from your consumers or clients. Through the use of Edge locations, clients and customers can directly interact with Edge locations which can upload data and download data directly from the desired S3 bucket. And also S3 transfers are routed through CloudFont Edge locations and backbone networks instead of the public internet. For our exam tips on this lesson on increasing S3 transfer speeds with

Transfer Acceleration, keep in mind these key takeaways about S3 Transfer Acceleration: First, it's ideal for long distance transfers which allows you to quickly and securely transfer files to and from S3 over long distance. It also provides fast transfer speeds which is useful when you have content which needs to be updated over long distances. This will be great in improving your speeds from 50 to 500%. And keep in mind the use case for this particular feature is for customers with widespread users or applications that are hosted far away from your S3 bucket. So that's all for this lecture. If you're ready to move on, please join me in the next lesson.

Enhancing S3 Performance with Multipart Upload

Hello, Cloud Gurus. And welcome back to this lecture on Enhancing S3 Performance with Multipart Upload. In this lesson, we'll introduce how you can increase your S3 performance through using S3 Multipart Upload. We'll talk about that is a lot of data. We'll also talk about a better way to handle moving data which is multipart upload. And we'll conclude with some exam tips. Let's go ahead and get started. Let's start with this practical example of moving a building. Now, if we were told to move a building, we'd have a lot of challenges as it relates to this cumbersome task. Moving a building is first impractical. It's also time consuming, and it could also get potentially expensive as we ensure we don't harm anyone or the building itself. This is the exact challenge that we come across when moving a lot of data. So, uploading a big file to S3 presents similar challenges. First, a single S3 put can be up to 5 GiB of data. However, S3 objects can be up to 5 TiB of data. So how do we get 5 TiB of data into S3? We do it through the better way, which is our S3 multipart upload. So there's several steps that we follow here for S3 multipart upload. We first begin to prepare the data by breaking data into reasonably sized pieces. We then move the pieces to perform the multi-upload steps, to move all the data into your S3 bucket. Then, S3 is able to put the data back together. We then let S3 know that the upload is complete, and S3 puts the data back together in the bucket. Some other facts that we should keep in mind as it relates to S3 multipart upload, is that it's recommended for files over 100 MB, and it's required for files over 5 GB. Also, this is a great way to paralyze our uploads to increase our efficiency. We basically start with a big file, which is then split into multiple parts, and the multiple parts who then multi-upload it into our S3 bucket in parallel uploads. So for our exam tips for enhancing S3 performance with multipart upload, keep in mind that this solution is ideal for moving large objects, which can be cumbersome and expensive. Also, the use of S3 multipart uploads is to increase our performance when uploading files to S3. Finally, multipart uploads should be used for any files over 100 megabytes, and must be used for any file over 5 gigabytes. That's all for this lesson on Enhancing S3 Performance with Multipart Upload. If you're ready to move on, join me in the next lesson.

Using Placement Groups to Increase TCP/IP Traffic Flow

Hello, Cloud Gurus. And welcome back to this lecture on Enhancing S3 Performance with Multipart Upload. In this lesson, we'll introduce how you can increase your S3 performance through using S3 Multipart Upload. We'll talk about that is a lot of data. We'll also talk about a better way to handle moving data which is multipart upload. And we'll conclude with some exam tips. Let's go ahead and get started. Let's start with this practical example of moving a building. Now, if we were told to move a building, we'd have a lot of challenges as it relates to this cumbersome task. Moving a building is first impractical. It's also time consuming, and it could also get potentially expensive as we ensure we don't harm anyone or the building itself. This is the exact challenge that we come across when moving a lot of data. So, uploading a big file to S3 presents similar challenges. First, a

single S3 put can be up to 5 GiB of data. However, S3 objects can be up to 5 TiB of data. So how do we get 5 TiB of data into S3? We do it through the better way, which is our S3 multipart upload. So there's several steps that we follow here for S3 multipart upload. We first begin to prepare the data by breaking data into reasonably sized pieces. We then move the pieces to perform the multi-upload steps, to move all the data into your S3 bucket. Then, S3 is able to put the data back together. We then let S3 know that the upload is complete, and S3 puts the data back together in the bucket. Some other facts that we should keep in mind as it relates to S3 multipart upload, is that it's recommended for files over 100 MB, and it's required for files over 5 GB. Also, this is a great way to paralyze our uploads to increase our efficiency. We basically start with a big file, which is then split into multiple parts, and the multiple parts who then multi-upload it into our S3 bucket in parallel uploads. So for our exam tips for enhancing S3 performance with multipart upload, keep in mind that this solution is ideal for moving large objects, which can be cumbersome and expensive. Also, the use of S3 multipart uploads is to increase our performance when uploading files to S3. Finally, multipart uploads should be used for any files over 100 megabytes, and must be used for any file over 5 gigabytes. That's all for this lesson on Enhancing S3 Performance with Multipart Upload. If you're ready to move on, join me in the next lesson.

Identifying Underutilized Resources Using Trusted Advisor

Hello, Cloud Gurus, and welcome back to this lesson on identifying underutilized resources using Trusted Advisor. In this brief lesson, we'll help you get a better understanding of how you can identify underutilized resources within Trusted Advisor, and we'll cover the resources that it identifies within the service. As usual, we'll wrap up with some exam tips. Let's go ahead and jump in. Within Trusted Advisor, there's several resources outlined as underutilized, which will help you make sure you're not paying for more resources than you're using. Let's quickly cover these in this lesson. Starting with EC2 or Amazon Elastic Compute Cloud, Trusted Advisor lets you know if you have a low utilization of EC2 instances as well as idle load balancers or unassociated elastic IP addresses. Trusted Advisor also notifies you with Elastic Block Store, or EBS, if you have any low utilization of EBS volumes. And with RDS or Amazon's Relational Database Service, Trusted Advisor makes you aware of any idle database instances as well as any RDS Reserved Instances which can be optimized. And finally, Trusted Advisor alerts you of any Amazon Redshift resources which has low utilization of Redshift clusters. For exam tips on identifying underutilized resources using Trusted Advisor, keep in mind, AWS Trusted Advisor provides these types of optimization recommendations across multiple services: security, performance, service limits, fault tolerance. However, in this lesson, we covered cost optimization. And with cost optimization, here are some of the alerts that you can find within Trusted Advisor. Low utilization of EC2 instances, idle load balancers, idle Amazon RDS instances, savings plans, Amazon RDS Reserved Instance optimization, as well as AWS Lambdas with high error rates and timeouts. So that's all for this lecture. If you're ready to move on, please join me in the next lesson.

Demo: Performance Analysis Using RDS Performance Insights

Hello Cloud Gurus and welcome to this lesson, which is going to cover performance analysis using RDS Performance Insights. Now RDS Performance Insights provides database monitoring that helps you to monitor your database load on your RDS database. And it allows you to monitor your performance data by database waits, SQL queries, host connections, and users that are executing SQL queries. For our demo objectives, first of all, there is a prerequisite to download and install the

pgAdmin tool on your local machine. And you can download pgAdmin and install it from this website, pgadmin.org/download, which is linked in the resources for this lesson. And it's very straightforward. Just download the version that you need for your own operating system and install it on your local machine. Next, we'll launch our RDS instance using the RDS dashboard. And then we'll be able to analyze our performance data using RDS Performance Insights. So if you're ready to get started, download and install the pgAdmin tool. And then after that, please join me in the AWS console. So from the AWS console, search for RDS. Create database. Select Standard create and PostgreSQL. Under Templates, select Free tier. Scroll down to the Password section and provide a master password and confirm the password. Under Instance configuration, make sure that Burstable classes is selected and select t3.micro. The default storage configuration is fine for this demo. Then, under Connectivity, scroll down to Public access and select Yes. And we want to enable public access so that we can connect to the database later on from our local machine. Under Availability zone, we can select us-east-1a. So now, scroll down to the Monitoring section, and you'll see that Performance Insights is already enabled. So that's all good. And now, all we need to do is scroll down to the end and Create database. Now it will take a few minutes to finish creating the database. So have a cup of tea. Relax for a few minutes while it finishes initializing. And after a few minutes, we'll be good to continue. And a few minutes later, it's almost finished creating our database. Mine's in the status of backing up, so I think we're good to continue now. So the next thing we're going to do is configure the database security group, and that's going to allow us to connect from our local machine using pgAdmin. So select your database identifier. Then, under Connectivity & security, find the security group that is associated with your RDS instance. So I'm going to select the security group, edit inbound rules, add a rule. We're going to search for PostgreSQL. There it is. And source is going be Anywhere and Save rules. So now, we're going to head back to RDS. Select Databases. The status of my database is now available. So, we are ready to test our connectivity. And to do that, we are going to use pgAdmin. So hopefully, you've downloaded pgAdmin to your local machine. And when you first start it up, it might ask you to reset your master password. So select Reset Master Password, answer Yes to this question, and set a new master password. And now we are ready to connect to our database. So come to Servers on the left, right-click, select Register, and we're going to register a server. We'll give it a name. Then, select Connection. And for the host address, we need to provide the database endpoint. So from the AWS console, select your database identifier. And then under Connectivity & security, this is where you'll find the endpoint, and here it is. So copy that and paste it. It's already populated with the correct port, 5432. The username is going be Postgres, and you need to enter the database password that you created earlier. Then hit Save. And if it's all worked, you should be connected to your database. And there it is. So in the left-hand pane, select the drop-down. Select Postgres, right-click, and select Query Tool. And we're going to run an SQL query that I've provided in the resources for this lesson. And here it is. And this query is just going to create a table and insert some data so that our RDS database has something to work with. So copy all of this text, paste it into your query window, and then you can run the query by using this Play button to execute your query. And if it's been successful, then you'll see a message just like this one appear in the Messages section. And we can test that our data has been properly added by running another query. So underneath this, type SELECT * FROM cloud_users, and that's the name of our table. Then, use your mouse to highlight the new command. And then use the Play button to execute that query. And then down here, in the Data Output section, we can see the data that's been added to our database that's been selected using this query. So now that we've had some activity on our database, we can take a look at Performance Insights and see what activity is detected. So back in your RDS management console, on the left-hand pane, select Performance

Insights. Select your database from the drop-down. And straightaway, I can see some performance information relating to my database, and it's already started to populate in the graph down here. At the top of the screen, you can set the time range. So I'm going to leave it set to the last 1 hour. And in this first graph, I can see the number of commits that have occurred and when they happened. Then down here, I can view, information about the database load. But right now, it's not giving us much data. But what if I go back to pgAdmin and insert another 100,000 records? So back in pgAdmin, I'm going to use my mouse to highlight this section, so everything from INSERT to COMMIT. Just make sure that's highlighted, and then you can hit the Play button again. In fact, I'm going to hit it a few times just to insert loads of records into my database. And then we can come back to Performance Insights. We can refresh using the button at the top here. And you might even want to change the range to the last 5 minutes and hit Apply. And then when we scroll down, we can see information about our database waits. And then down here, we can get a summary of the top waits, so what's been waiting in my database, the top SQL queries that have been committed in the database. So here's our SQL statements. Under Top hosts, these are the top hosts that are connecting to our database, and this is actually the IP address of my local machine that I'm connecting on. Here's my top users, and it's actually mainly the Postgres user. And that's the admin user that I'm connecting to the database on. Top session types. It's our client back end, so our database client. And top application is pgAdmin. And then we've also got top databases as well. And the top database is Postgres, and that's just the name that was given to our database over here. So that is RDS Performance Insights, and it's a great tool for understanding what is going on with your RDS database. So now, let's review our exam tips. So for the exam, remember that RDS Performance Insights is a database monitoring tool, and it helps you to monitor your load on your RDS database. You can analyze the data by viewing the waits that are happening on your database, the SQL queries that are being executed, host connectivity, as well as the users who are executing queries. And you can also view the data for different databases within your RDS instance, the applications that are connecting, and session types as well. So that's all for this lesson. Any questions, let me know. Otherwise, feel free to move on to the next lesson. Thank you.

Increasing Scalability with RDS Proxy

Hello Cloud Gurus, and welcome back to this lecture on increasing scalability with RDS Proxy. In this lecture, we'll help you understand how you can increase the scalability of your applications through the use of RDS Proxy. We'll talk about how RDS Proxy works, RDS Proxy fault tolerance, as well as other use cases for RDS Proxy. And then we'll wrap up with some exam tips. So let's go ahead and jump in. Let's start by understanding how RDS Proxy works. In a traditional architecture, you have your client application where your application is pointed toward an RDS database and the RDS database receives information from the application through direct connection. Now, let's see how this would work with RDS Proxy, starting with your client application, which is now directly pointed toward the RDS Proxy and the RDS Proxy pools and shares database connections to assist with application scalability and database efficiency. The RDS database now receives information from the application directly through RDS Proxy. So that's generally how RDS Proxy works standing as a middleware between your client application and your RDS database to assist with scalability and efficiency. Now, let's discuss the benefits of RDS Proxy which is to provide fault tolerance and increase our overall application availability. RDS Proxy is serverless and scales automatically to your workload through pooling and sharing database connections. So let's say you have your applications which are directly routed to your RDS Proxy. RDS Proxy then pools and

shares those connections directly with the available database to make sure your information persists as you would expect. RDS also preserves the application connections during a failover. So let's say your primary database has a failure, RDS would then reroute that particular, those particular connections to your new and active primary database. RDS Proxy also detects failover and routes requests to your standby quickly. It also is deployable over multi-availability zones which protects us from infrastructure failure. So let's say there is an outage in your initial availability zone. RDS Proxy would be great for forwarding those connections across to another availability zone, which has availability. And finally, RDS Proxy has up to 66% faster failover times than a traditional connection, which is directly from your application to your RDS database. Now, let's discuss other use cases for RDS Proxy. So RDS Proxy is great for applications with unpredictable workloads as well as applications which open and close database connections infrequently. It's also good for applications which require availability through transient failures. And finally, it's great for applications with open but idle connections. Now let's discuss our exam tips for increasing scalability with RDS Proxy. Keep in mind the purpose of the RDS Proxy is to pool and share database connections to assist with application scalability and database efficiency. And just keep in mind that it stands between the client application and the RDS database. Also, keep in mind that it's goal is to increase application availability and these are the specific reasons and how it does that. It's serverless and scales automatically to your workload through pooling and sharing database connections. It preserves application connections during a failover. It detects over and routes requests to stand by quickly. It's also deployable over multi availability zones for protection from infrastructure failure and it's up to 66% faster failover times. So that's all for this lecture on increasing scalability with RDS Proxy. If you're ready to move on, please join me in the next lecture.

Enhancing EC2 Performance Using Instance Store

Hello, Cloud Gurus, and welcome back to this lecture on Enhancing EC2 Performance Using Instance Store. In this lesson, we'll help you get a better understanding of how you can enhance your EC2 and application performance through the use of EC2 instance store. We'll first get started by understanding the different types of storage which are available to you as you run applications on your EC2 instances. We'll also discuss what exactly is the EC2 instance store as well as the instance store performance. And as usual, we'll conclude with exam tips. Let's jump right in. Let's begin by discussing the types of storage that's available to you as you run your applications on your EC2 instances. There's 2 types of overarching storage types that we should become familiar with. There's first file storage and we'll discuss which services in AWS belongs to file storage and there's block-level storage. So when we begin to look at file storage, we first need to understand that Amazon S3 is a part of file storage and the benefit of Amazon S3 is it stores our data from anywhere on the web, and it's also great for storing snapshots, AMIs, data files, et cetera. There's also Amazon Elastic File Storage or EFS and this is a scalable file system for workload and multi-instance applications. It's also great for mounting multi-AZ targets for highly available data retrieval. Now, on the other hand, with block-level storage we have Amazon EBS or Elastic Block Store and this would be long-term block-level storage for your EC2 instances. And this particular storage exists independently from the life of the instance. And this is great for primary storage for file systems, databases, as well as unformatted block-level storage. However, the most important one that we're going to discuss within this lesson is the instance store or the EC2 instance store. And this is temporary block-level storage devices for your EC2 instances. And these do not exist

independently from the life of the EC2 instances. These are great for frequently changing data, caches, temporary content, buffers, et cetera. Now let's discuss EC2 instance store in greater detail. The important thing to note about EC2 instance store is that it is the ephemeral or short-term storage and this is how it works. Let's say you have an AMI that's sitting in or residing in an Amazon S3 bucket and then you have your host computer where your instance store can have several ephemerals located within it. And these ephemerals can be launched from any of the AMIs that are currently residing in your S3 bucket. They can also be mapped to block device mappings and can be stored as your primary instance or an additional drive on your host computer. Now, let's keep in mind these key details about EC2 instance store. EC2 instance store can have a maximum of 10 gigabytes on the instance store and the throughput is greater than the throughput that you could expect from Elastic Block Store or EBS. The volumes are created from a template in S3 and also they can be selected as a root volume or as additional volumes, and they must be configured at the launch of the EC2 instance. The overall thing we'd like to note here is because of the ephemeral storage, you have to be careful that you do not lose your data. And these are some potential underlying issues that may result in losing your data through the use of EC2 instance store. Your instance may potentially terminate, the instance may stop, or the instance may hibernate, and those 3 things can cause you to lose your data on your EC2 instance store. So what do you do? Here's our tip. Our tip is for your EC2 instance store, here are the suggestions. Distribute instance store data across multiple availability zones, as well as automate the backup of the instance store volume to persistent storage on a regular basis. Now, for additional optimal instance performance, AWS recommends that you first use a recent version of an Amazon AMI as well as properly map the SSD instance store on EC2 launch and enable swap space on your volume. Now, let's conclude with our exam tips for EC2 instance store. Keep in mind that EC2 instance store is a temporary block-level storage device for your EC2 instance and it does not exist independently from the life of the instance. And this is great for frequently changing data, caches, temporary content, buffers, et cetera. As it relates to your EC2 instance store performance, for optimal performance, AWS recommends that you first use a recent version of the Amazon AMI. Also properly map the SSD instance store on the launch of your EC2 instance and finally, enable swap space on the volume. So that's all for this lecture on enhancing EC2 performance using the instance store. If you're ready to move forward, please join me in the next lesson.

Section Review: Cost and Performance Optimization Summary

Hello Cloud Gurus. An awesome job at making it through the final section of this course. Let's review our cost and performance optimization summary, which are the highlights of what we've covered in this section. We started this section by discussing AWS Cost Explorer and cost allocation tags. And we understand that Cost Explorer allows us to visualize, view, and analyze our AWS usage cost. In this lesson, we also define tags, which tags are key value pairs used to mark and track AWS resources. And if you'd like to track your AWS resource cost, you would use cost allocation tags. And cost allocation tags must be enabled through the AWS console to track AWS resource cost. And the service we use to track our cost is AWS Cost Explorer, which provides us reporting. And Cost Explorer allows you to create reports based on cost, usage, savings plans, as well as reservations. We also discuss how you can improve your cost using AWS Budgets. And we discuss that AWS Budgets allows you to create and manage budgets to improve cost for your account usage. And there's several budget types that you can create within AWS Budgets. This would include a cost budget which was based on a specific dollar amount, a usage budget, reservation, and savings plans.

You can also configure budget alerts, and budget alerts are sent via email and through SNS topics. We also demonstrated how to create an AWS Budget and billing alarm. And we discussed that billing alarms are created through CloudWatch. And in CloudWatch, you can create alarms based on your account cost as well as your service cost and a total estimated charge. AWS Budgets on the other hand allows you to create budgets based on your cost, your usage, your reservation, and your savings plan. We also discuss how to reduce your cost through the use of managed services, and we discuss that managed services are services for which AWS is responsible for the managing of the infrastructure. Keep in mind, scalability and elasticity to keep up with your growing demands as well as software updates. And we discuss 3 key services for managed services, which were: AWS Fargate, which is key for managing elastic container services or elastic Kubernetes services; we discuss AWS's file storage or EFS, which is great for managing a file system and growing and shrinking to meet the throughput demand of your file system; and lastly, we discussed RDS or relational database storage, and the features like Aurora Serverless, which is a serverless relational database which allows you to have elasticity and scalability to meet your performance and cost. So you only pay for what you use. And the entire thought of why you should use managed services is because it increases your AWS performance, while reducing your AWS operational support and cost. We also discussed the use of AWS Compute Optimizer. And AWS Compute Optimizer is a machine learning-based recommendation service which helps you optimize the performance as well as the cost of your compute resources. AWS Compute Optimizer does this by performing an analysis of your EC2 instances, your auto scaling groups, your Elastic Block Stores, as well as your Lambda. And all recommendations provided by AWS Compute Optimizer are available in CSV or JSON and can be exported to an S3 bucket of your choosing. We also discussed that metrics can be used to improve your performance and cost. Keep in mind, CloudWatch metrics are data about the performance and cost of your system. Metrics Explorer allows you to filter, aggregate, and graph metrics within the AWS console. And CloudWatch metrics streams allows you to configure your application to stream to S3 or Kineses Firehose. We also discuss increasing S3 transfer speeds with Transfer Acceleration. And here are some exam tips to keep in mind for this topic. S3 transfer speeds with Transfer Acceleration is great for long distance transfers, which allows you to quickly and securely transfer files to and from S3 over long distances. It also provides faster transfer speeds for content transfer speed improvements of up to 50 to 500% for long distance transfers. And this would be a great use case for customers with widespread users or applications that are hosted far away from your S3 bucket. We also discuss how to enhance S3 performance with multipart upload. And here are some exam tips on multipart upload for S3. Keep in mind that moving large objects can be cumbersome and expensive. This is why we should use multipart uploads to increase the performance when uploading these large files to S3. Multipart uploads should be used for any file over 100 megabytes and must be used for any file over 5 gigabytes. For our exam tips for reserved versus spot instances, keep in mind how we understood in the scenario that we gave to differentiate the 2. We compared this to an employee who would work out of town and would need to find accommodations, so that employee who would stay a longer term period would make a commitment to a corporate apartment while an employee who only needs for a short, short period or random amount of time would use a hotel. And the price differences would be as such. The corporate apartment or the reserved instance would provide a 40 to 60% discount, while the hotel would provide a 50% to 90% discount because of its pricing during its availability. The reserved instance would require a 1-3 year commitment, and the spot instance would have no commitment. And the reserved instance would allow us to pay upfront, partially upfront, or monthly, while the spot instance we would pay the spot price which is currently in effect. We also discuss the use of

placement groups to increase our TCP/IP traffic flow, and here are some exam tips that we can take into our exam to remember around this topic. Placement groups allow us to control how our EC2 instances are placed in the strategy we use to define it. And keep in mind these 3 placement group strategies. We have the cluster placement group, and this is the placement group strategy where all of our instances are placed in the same availability zone on the same rack. And this is suitable for low-latency, high network throughput applications. We also have the partition placement group, and this is where instances are created in a partition with their own rack and independent power and networking. And this is suitable for HDFS, Hbase, and Cassandra applications. And finally, we'd talked about the spread placement group. And the spread placement group allow our instances to be placed on a distinct rack with their own network and power source. And this is great for apps running a small number of critical instances you need to segregate for resilience. We also discussed how increasing scalability with RDS proxy provides fault tolerance and increased application availability. We discussed that it's serverless and scales automatically. It preserves our application connections during a failover. It also detects failover and routes requests to stand by quickly. And it also is deployable over multi-availability zone for protection from instance infrastructure failure. And it provides up to 66% faster failover times. For our instance store exam tips, EC2 Instance Store, as we spoke previously within this section, is a temporary block level storage device for our EC2 instances. However, they do not exist independently from the life of the instance. This particular storage is great for frequently changing data, caches, temporary content, buffers, et cetera. And from an EC2 instance store performance perspective, for optimal instance store performance, AWS recommends that you use a recent version of an Amazon AMI as well as properly map the SSD instance store on EC2 instance launch. And enable swap space on the volume. We also discussed in this section when it's great to use EC2 enhanced networking. We started by discussing what's EC2 enhance networking and it's improved networking for your EC2 instances, which allows you to provide higher bandwidth, higher packets per second, as well as lower latency between instances. And we discussed the critical component in order to use EC2 Enhance networking, which is the Elastic Network Adapter. And the Elastic Network Adapter is a custom interface used to optimize network performance. Finally, we discussed performance analysis using RDS Performance Insights. And RDS Performance Insights is RDS database monitoring, which helps you monitor your database load. And RDS Insights allows you to analyze your database performance insights data by the waits in your RDS database as well as the SQL that's being executed within your RDS database, your host, as well as which users are executing queries within your database. So that's all for this section review on cost and performance optimization. If you're ready to move forward, please join us in the next lecture.

Additional Resources to Help You Prepare for the Exam

Additional Resources to Help You Pass the Exam

Hello Cloud Gurus, and welcome to this lecture, which is going to go over all the additional resources that are going to help you prepare for and pass your exam the very first time. So, first of all, I'm going to talk to you about the official AWS FAQs, which you can use to find answers to commonly raised questions about the AWS services that you will need to know for this exam. There's also the whitepapers, which have loads of information, including AWS-recommended architectures and some really good diagrams and general guidance and best practices. And then finally, we have the re:Invent videos, and these are great videos that you can find on YouTube, and

they are official AWS videos, which contain loads of information about best practices, deep dives on some of the services, as well as demos. So let's take a look at the FAQs, which are going to be most useful when preparing for the SysOps Administrator Associate exam. And I've tried to group these according to the exam domains. So first of all, for monitoring, logging and remediation, you'll want to take a look at the FAQs for CloudWatch, CloudTrail, Systems Manager, Config, and EventBridge. For reliability and business continuity, it's Auto Scaling, ElastiCache, and also RDS, and you want to focus on the reliability and business continuity features that are built into RDS. So things like multi-AZ, read replica and snapshots. When it comes to deployment, provisioning, and automation, you'll want to take a look at CloudFormation, EC2, Elastic Block Store, and Systems Manager. For security and compliance, take a look at Identity and Access Management, Single Sign-On, AWS Organizations, KMS, Inspector and Trusted Advisor, AWS WAF and Shield, Security Hub, and GuardDuty as well. And for networking and content delivery, focus on VPCs, VPC endpoints, Route 53, Direct Connect, CloudFront, and Elastic Load Balancer. And then finally, for storage and data management, take a look at S3, RDS, Athena, ElastiCache, Elastic File System, Storage Gateway, and RDS. And of course it's really, really important to get as much hands-on practical experience as possible with all of these services. So if you do feel like you've got any weak areas, just go back and redo our practical labs again, and definitely, if you do have time, review these FAQs. So moving on to the whitepapers that we recommend you read, and there are actually loads of them for the SysOps Administrator. Starting off with Development and Test on AWS. That's a really good one to read. There's also Disaster Recovery of Workloads on AWS, Amazon VPC Connectivity Options, Building a Scalable and Secure Multi-VPC AWS Network Infrastructure. There's also Introduction to AWS Security, and of course security is super important for all of the Associate-level certifications, so you're definitely going to want to read that one. There's also the AWS Well-Architected Framework, and Web Application Hosting in the AWS Cloud, and I've included links to all of these whitepapers in the Resources section of the course, just to group everything together and make it really easy for you to find. And I know you might be thinking, "please, no more whitepapers", but I'm going to give you my tips for getting the most out of these whitepapers. So first of all, it's not bedtime reading, so you don't need to read it from beginning to end like a novel. And the first thing that I do when I sit down to read one of these whitepapers is I just browse through the entire document, and I really try to get a feel for the material that's within the document. So I read the introduction, first of all, I take a look at the table of contents, and I try to get an understanding of the structure of the document and the material it contains, and then I go ahead and look at any diagrams, any headings, any tables, etc. And then only after that, I revisit the sections that I really think are worth reading, and I really try to focus on the things that I don't know very well. And I try not to spend too much time reviewing or reading over material that I already know really well. And I always try to keep in mind what I'm expecting to get out of the whitepaper. So if I'm reading a whitepaper about disaster recovery best practices, then that is exactly what I try to focus on. And I try not to get bogged down with any of the background information, because of course, there's also going to be information in there that is not relevant to the SysOps exam. And then finally, I always take notes. So I always try to keep it simple, and I just write down some key words that are going to trigger an idea when I read them back. I try to make my own diagrams and use mind mapping techniques, but I do also try to make sure that my notes are as brief as possible, so they only include the key points of the whitepaper. And if you really hate reading and you really can't stomach another whitepaper, then don't worry, because I've curated a list of really good AWS re:Invent videos that you can watch instead. So there's one on VPC Fundamentals and Connectivity Options, Networking Best Practices with the Well-Architected

Framework, Security Best Practices, Getting Started with AWS Identity, and Monitor All Your Things: Amazon CloudWatch in Action, and I've created a YouTube playlist so that you can find all of these really easily, and you'll find the link to my playlist in the Resources section of the course. So that's it for this lecture. If you have any questions, do let me know. Otherwise I will see you in the next lecture. Thank you.

Interactive Study Guide

Hello, Cloud Gurus and welcome to the SysOps Administrator Associate study guide. For this course, we are providing the study guide in a brand new interactive format. At the beginning of each section, you'll find a summary breakdown like this, allowing you to navigate the topics. So you can head straight to the topic that you would like to review. When reviewing diagrams like this, you can hover or click this icon to reveal an explanation of each component in the diagram. We've included lots of diagrams which allow you to test your understanding, and then hover to reveal the definition. Do you remember which services the AWS WAF integrates with? Well, this diagram will help you check if you are correct or not. We hope you enjoy this new interactive format. Keep being awesome, Cloud Gurus.