E-LEARNING

AWS Cloud Practitioner Essentials

Module 3 – Global Infrastructure and Reliability

Contents

Learning objectives

- Summarize the benefits of the AWS Global Infrastructure.
- Describe the basic concept of Availability Zones.
- Describe the benefits of Amazon CloudFront and edge locations.
- Compare different methods for provisioning AWS services.

Selecting a Region

It's not good enough to have one giant data center where all of the resources are. If something were to happen to that data center, like a power outage or a natural disaster, everyone's applications would go down all at once. You need high availability and fault tolerance. Turns out, it's not even good enough to have two giant data centers. Instead, AWS operates in all sorts of different areas around the world called Regions. **Regions are geographically isolated areas.**

Throughout the globe, AWS builds Regions to be closest to where the business traffic demands. Locations like Paris, Tokyo, Sao Paulo, Dublin, and Ohio**. Inside each Region, we**

have multiple data centers that have all the compute, storage, and other services you need to run your applications.

Each Region can be connected to each other Region through a **high speed fibre network**, controlled by AWS, a truly global operation from corner to corner if you need it to be. Now before we get into the architecture of how each Region is built, it's important to know that you, the business decision maker, gets to choose which Region you want to run out of. And each Region **is isolated from every other Region** in the sense that absolutely **no data goes in or out of your environment in that Region without you explicitly granting permission** for that data to be moved. This is a critical **security** conversation to have.

For example, you might have government compliance requirements that your financial information in Frankfurt cannot leave Germany. Well this is absolutely the way AWS operates outta the box. Any data stored in the Frankfurt Region never leaves the Frankfurt Region, or data in the London region never leaves London, unless you explicitly, with the right credentials and permissions, request the data be exported.

Regional data sovereignty is part of the critical design of AWS Regions. With data being subject to the local laws and statutes of the country where the Region lives.

When determining the right Region for your services, data, and applications, consider the following **four** business factors.

Compliance with data governance and legal requirements

Depending on your company and location, you might need to run your data out of specific areas. For example, if your company requires all of its data to reside within the boundaries of the UK, you would choose the London Region.

Not all companies have location-specific data regulations, so you might need to focus more on the other three factors.

Proximity to your customers

Selecting a Region that is close to your customers will help you to get content to them faster. For example, your company is based in Washington, DC, and many of your customers live in Singapore. You might consider running your infrastructure in the Northern Virginia Region to be close to company headquarters, and run your applications from the Singapore Region. Locating close to your customer base, usually the right call.

Available services within a Region

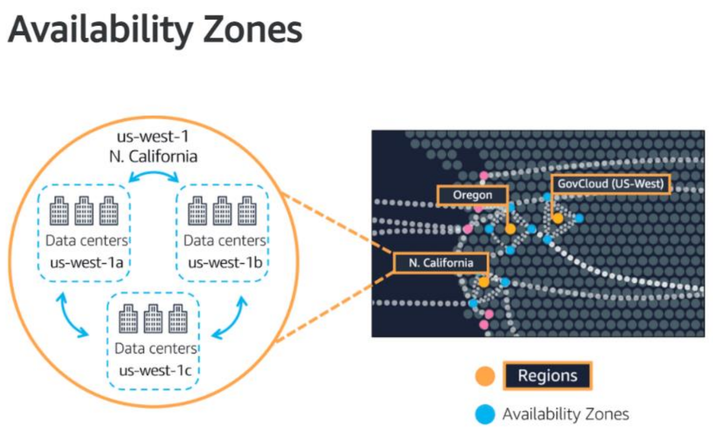
Sometimes, the closest Region might not have all the features that you want to offer to customers. AWS is frequently innovating by creating new services and expanding on features within existing services. However, making new services available around the world sometimes requires AWS to build out physical hardware one Region at a time.

Suppose that your developers want to build an application that uses Amazon **Braket** (AWS **quantum computing** platform). As of this course, Amazon Braket is not yet available in every AWS Region around the world, so your developers would have to run it in one of the Regions that already offers it.

Pricing

Suppose that you are considering running applications in both the United States and Brazil. The way Brazil's tax structure is set up, it might cost 50% more to run the same workload out of the São Paulo Region compared to the Oregon Region. You will learn in more detail that several factors determine pricing, but for now know that the cost of services can vary from Region to Region.

Availability Zones

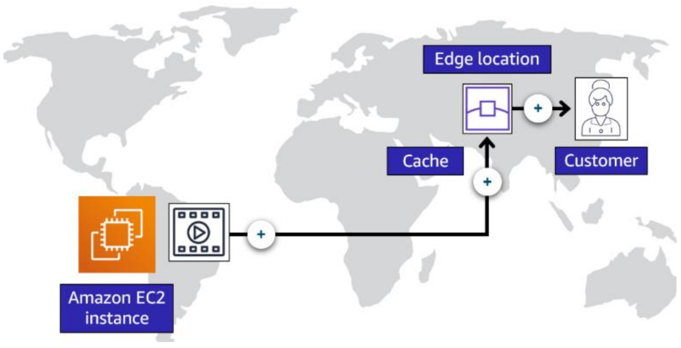


An **Availability Zone** is a **single data center or a group of data centers within a Region**. Availability Zones are located tens of miles apart from each other. This is close enough to have low latency (the time between when content requested and received) between Availability Zones. However, if a disaster occurs in one part of the Region, they are distant enough to reduce the chance that multiple Availability Zones are affected.

Each Availability Zone is one or more discrete data centers with redundant power, networking, and connectivity. When you launch an Amazon EC2 instance, it launches a virtual machine on a physical hardware that is installed in an Availability Zone. This means **each AWS Region consists of multiple isolated and physically separate Availability Zones within a geographic Region**.

Edge Locations

An **edge location** is a site that **Amazon CloudFront** uses to **store cached copies of your content** closer to your customers for faster delivery.



Suppose that your company's data is stored in Brazil, and you have customers who live in China. To provide content to these customers, you don't need to move all the content to one of the Chinese Regions. Instead of requiring your customers to get their data from Brazil, you can **cache a copy locally at an edge location** close to your customers in China.

When a customer in China requests one of your files, Amazon CloudFront retrieves the file from the cache in the edge location and delivers the file to the customer. The file is delivered to the customer faster because it came from the edge location near China instead of the original source in Brazil.

Caching copies of data closer to the customers all around the world uses the concept of **content delivery networks**, or **CDNs**. CDNs are commonly used, and on AWS, we call our CDN **Amazon CloudFront**.

Amazon CloudFront is a service that helps deliver data, video, applications, and APIs to customers around the world with low latency and high transfer speeds. Amazon CloudFront uses what are called **Edge locations**, all around the world, to help accelerate communication with users, no matter where they are.

Edge locations are separate from Regions, so you can push content from inside a Region to a collection of Edge locations around the world, in order to accelerate communication and content delivery. AWS Edge locations, also run more than just CloudFront. They run a **domain name service, or DNS, known as Amazon Route 53**, helping direct customers to the correct web locations with reliably low latency.

But what if your business wants to use, AWS services inside their own building? Well sure. AWS can do that for you. Introducing **AWS Outposts**, where AWS will basically install a fully **operational mini Region, right inside your own data center**. That's owned and operated by AWS, using 100% of AWS functionality, but isolated within your own building. It's not a solution most customers need, but if you have specific problems that can only be solved by staying in your own building, AWS Outposts can help.

Ways to interact with AWS services

We've been talking about a few different AWS resources as well as the AWS global infrastructure. You may be wondering, how do I actually interact with these services? And the answer is APIs. **In AWS, everything is an API call**. An API is an application programming interface. And what that means is, there are pre-determined ways for you to interact with AWS services. And you can invoke or call these APIs to provision, configure, and manage your AWS resources.

For example, you can launch an EC2 instance or you can create an AWS Lambda function. Each of those would be **different requests and different API calls to AWS**. You can use the AWS Management Console, the AWS Command Line Interface, the AWS Software Development Kits, or various other tools like AWS CloudFormation, to create requests to send to AWS APIs to create and manage AWS resources.

AWS Management Console

The **AWS Management Console** is a web-based interface for accessing and managing AWS services. You can quickly access recently used services and search for other services by name, keyword, or acronym. The console includes wizards and automated workflows that can simplify the process of completing tasks.

You can also use the AWS Console mobile application to perform tasks such as monitoring resources, viewing alarms, and accessing billing information. Multiple identities can stay logged into the AWS Console mobile app at the same time.

Through the console, you can manage your AWS resources visually and in a way that is easy to digest. This is great for getting started and building your knowledge of the services. It's also useful for building out test environments or viewing AWS bills, viewing monitoring and working with other non-technical resources. The AWS Management Console is most likely the first place you will go when you are learning about AWS.

However, once you are up and running in a production type environment, you don't want to rely on the point and click style that the console gives you to create and manage your AWS resources. For example, in order to create an Amazon EC2 Instance, you need to click through various screens, setting all the configurations you want, and then you launch your instance. By having humans do this sort of **manual provisioning**, you're opening yourself up to potential **errors**. It's pretty easy to forget to check a checkbox or misspell something when you are doing everything manually.

The answer to this problem is to use tools that allow you to script or program the API calls.

AWS Command Line Interface (CLI)

To save time when making API requests, you can use the **AWS Command Line Interface (AWS CLI)**. AWS CLI enables you to **control multiple AWS services directly from the**

command line within one tool. AWS CLI is available for users on Windows, macOS, and Linux.

By using AWS CLI, you can automate the actions that your services and applications perform through scripts. For example, you can use commands to launch an Amazon EC2 instance, connect an Amazon EC2 instance to a specific Auto Scaling group, and more.

The CLI allows you to make API calls using the terminal on your machine. This is different than the visual navigation style of the Management Console. Writing commands using the **CLI makes actions scriptable and repeatable**. So, you can write and run your commands to launch an EC2 Instance. And if you want to launch another, you can just run the pre-written command again. This makes it **less susceptible to human error**. And you can have these scripts run automatically, like on a schedule or triggered by another process.

Automation is very important to having a successful and predictable cloud deployment over time.

AWS Software Development Kits (SDK)

Another option for accessing and managing AWS services is the **software development kits (SDKs)**. SDKs make it easier for you to use AWS services through an **API designed for your programming language or platform.** SDKs enable you to use AWS services with your existing applications or create entirely new applications that will run on AWS.

To help you get started with using SDKs, AWS provides documentation and sample code for each supported programming language. Supported programming languages **include C++, Java, .NET,** and more.

AWS Elastic Beanstalk

There are also other ways you can **manage your AWS environment** using **managed tools** like AWS **Elastic Beanstalk**, and AWS **CloudFormation**.

With **AWS Elastic Beanstalk**, you provide code and configuration settings, and Elastic Beanstalk **deploys** the resources necessary to perform the following tasks:

- Adjust capacity
- Load balancing
- **Automatic scaling**
- Application health monitoring

Instead of clicking around the console or writing multiple commands to build out your network, EC2 instances, scaling and Elastic Load Balancers. You can instead **provide your application code and desired configurations to the AWS Elastic Beanstalk service**, which then takes that information and builds out your environment for you.

AWS Elastic Beanstalk also makes it easy to save environment configurations, so they can be deployed again easily. AWS Elastic Beanstalk gives you the convenience of not having to provision and manage all of these pieces separately, while still giving you the visibility and control of the underlying resources.

AWS CloudFormation

With **AWS CloudFormation**, you can treat your **infrastructure as code**. This means that you can build an environment by writing lines of code instead of using the AWS Management Console to individually provision resources.

AWS CloudFormation is an **infrastructure as code tool** that allows you to define a wide variety of AWS resources in a **declarative way** using **JSON or YAML** text-based documents called CloudFormation templates. A declarative format like this allows you to define what you want to build without specifying the details of exactly how to build it. CloudFormation lets you define what you want and the CloudFormation engine will worry about the details on calling APIs to get everything built out.

It also isn't just limited to EC2-based solutions. CloudFormation supports many different AWS resources from storage, databases, analytics, machine learning, and more. Once you **define your resources in a CloudFormation template**, CloudFormation will **parse the template and begin provisioning all the resources you defined in parallel**. CloudFormation manages all the calls to the backend AWS APIs for you. You can run the same CloudFormation template in multiple accounts or multiple regions, and it will create identical environments across them. There is less room for human error as it is a totally automated process.

AWS CloudFormation provisions your resources in a safe, repeatable manner, enabling you to frequently build your infrastructure and applications without having to perform manual actions or write custom scripts. It determines the right operations to perform when managing your stack and rolls back changes automatically if it detects errors.

Recap

To recap, the AWS Management Console is great for learning and providing a visual for the user. The AWS Management Console is a manual tool. So right off the bat, it isn't a great option for automation. You can instead use the CLI to script your interactions with AWS using the terminal. You can use the SDKs to write programs to interact with AWS for you or you can use manage tools like AWS Elastic Beanstalk or AWS CloudFormation.

Summary

We covered how logical clusters of data centers make up Availability Zones, Availability Zones in turn make up Regions, and those are spread globally. You then choose what

Regions and Availability Zones you want to operate out of and as a best practice, you should always deploy infrastructure across at least two Availability Zones. And some AWS services like Elastic Load Balancing, Amazon SQS, and Amazon SNS already do this for you.

We also talked about Edge locations and how you can deploy content there to speed up delivery to your customers. We even touched upon edge devices like AWS Outposts which allow you to run AWS infrastructure right in your own data center.

Another thing we discussed was how to provision AWS resources through various options, such as the AWS Management Console, the SDK, CLI, AWS Elastic Beanstalk, and AWS CloudFormation, where you learned how you can set up your infrastructure as code.

I hope you learned how globally available AWS is and how easy it is to get started with provisioning resources.

Quiz

Which statement is TRUE for the AWS global infrastructure?

- A Region consists of two or more Availability Zones. For example, the South America (São Paulo) Region is sa-east-1. It includes three Availability Zones: sa-east-1**a**, sa-east-1**b**, and sa-east-1**c**.

Which factors should be considered when selecting a Region? (Select TWO.)

- Compliance with data governance and legal requirements
- Proximity to your customers

Which statement best describes Amazon CloudFront?

- A global content delivery service. Amazon CloudFront is a content delivery service. It uses a network of edge locations to cache content and deliver content to customers all over the world. When content is cached, it is stored locally as a copy. This content might be video files, photos, webpages, and so on.

Which site does Amazon CloudFront use to cache copies of content for faster delivery to users at any location?

- Edge location

Which action can you perform with AWS Outposts?

- Extend AWS infrastructure and services to your on-premises data center.

Additional resources

Review these resources to learn more about the concepts that were explored in Module 3.

- Global Infrastructure
- Interactive map of the AWS Global Infrastructure
- Regions and Availability Zones
- AWS Networking and Content Delivery Blog
- Tools to Build on AWS
- AWS Customer Stories: Content Delivery

Updated by Lionel Tchami – December