The AWS Dictionary of Pain

a straightforward guide to thorny cloud terms

A Cloud Guru

The AWS Dictionary of Pain

This idea of helping a team reach a “baseline cloud fluency” is a phrase we hear a lot at A Cloud Guru. You can have two engineers that excel at one kind of cloud operation, but if the rest of the team can’t speak their language, nothing gets done. Things slow down—or stop—when people can’t communicate.

At A Cloud Guru, our goal is to teach the world to cloud. So we turned to our data to help us understand how we can help bring organizations up to that baseline cloud fluency—or at least in the ballpark. Our goal with this guide is to make sure the next time someone mentions a major issue with the AWS SQS, you won’t be left scratching your head.

We analyzed 2.7 million responses to hundreds of questions across multiple areas of cloud expertise. We specifically looked at tough questions, where the correct response rate fell below what you’d probably call a C average (or 60%). From these questions we identified key products, technologies, and topics that were often represented in those questions—and are here to help you get a handle on them.

In this guide, we’ll cover some core concepts (like load balancing and availability zones) as well as some of the most common—though clearly challenging!—AWS tools you’ll find in a modern cloud stack.

You’ll still probably want to do a lot better than just a C average. But while you won’t need everyone to be an expert at everything, they still need to speak the same language. Or, at least, understand when someone’s telling a cloud-related joke that isn’t meant to be taken literally.

Here are the trip-up topics that stump cloud teams the world over: the AWS Dictionary of Pain.

Pain Forecast: Widespread Rain

Amazon DynamoDB

[amə͵zän, dīnə͵mō, dē'bē]

WHAT IS IT? DynamoDB is Amazon's approach to non-relational databases—or NoSQL for short. Developers can use DynamoDB to rapidly scale applications that have to work with a lot of data. DynamoDB can also automatically scale up and down depending on traffic. Amazon offers on-demand backups and point-in-time recovery through DynamoDB snapshots, protecting from accidental writes or deletes. Amazon replicates your data across multiple availability zones, ensuring that you don't lose your data.

NoSQL works for specific data models that offer flexible schemas. These databases are well-suited to modern applications that require high scalability, such as gaming. They excel at scalability and performance. Relational databases are optimized for storage, while NoSQL is optimized for compute power.

WHY IS IT HARD? DynamoDB (and NoSQL), like any tool, comes with tradeoffs. Traditional SQL databases support a wide range of queries, but they can struggle to perform cost-effectively at scale. NoSQL databases like DynamoDB can scale as big as you want, but it's up to you to store the data in a way that will make it easy to get out when you need it.

Beware DynamoDB's simple interface and low barrier to entry; if you approach it with SQL-esque assumptions, you may soon find yourself frustrated by the lack of traditional query operations like JOIN and GROUP BY. But if you take the time to learn how to construct performant NoSQL data models, DynamoDB is an unparalleled way to help ensure that your service isn't collapsing under the weight of its success.

Our learners missed tough questions 47% of the time related to DynamoDB. Here are a few sample questions:

An insurance provider will be migrating all of its applications to AWS. The production environment will reside in a single VPC for its web tier, app tier, data tier, and control plane resources. Many of the transactional applications make heavy use of Amazon DynamoDB. New data classification mandates have been issued to ensure that information with different access requirements is segregated. Which networking architecture will provide the most secure data classification solution for the DynamoDB information? (34%)

Your development team has created a gaming application that uses DynamoDB to store user statistics and provide quick game updates back to users. The team has begun testing the application but needs a consistent data set to perform tests. The testing process alters the dataset, so the baseline data must be retrieved upon each new test. Which AWS service can meet this need by exporting data from DynamoDB and importing data into DynamoDB? (39.1%)

Pain Forecast: Thunderstorm

Amazon EBS & Snapshot

[amə,zän, ē'bē'es, and, snap,SHät]

WHAT IS IT? We can't talk about EBS without talking about snapshots, so we'll cover both.

Amazon Elastic Block Store (EBS) is a virtual storage drive (like a hard disk or SSD) for your Amazon EC2 instances. You can think of it as the hard drive connected to your server.

1 : General-purpose SSDs balance price and performance for a wide variety of workloads. **2 :** Provisioned IOPS SSD is the highest-performance SSD volume, best for mission-critical applications. **3 :** Throughput optimized hard disk drive is a low-cost HDD volume designed for frequently accessed, throughput intensive operations. **4 :** A cold hard disk drive is the lowest cost for less-frequently accessed operations. **5 :** Magnetic drives are previous-generation drives, old and slow.

Anything your server writes to disk ends up stored in EBS as raw blobs of data called blocks. If you want to make a backup of that EBS volume, you take a snapshot of it and store it in S3. Each snapshot serves as an incremental backup for an EBS volume.

WHY IS IT HARD? Like EC2 instances, EBS volumes scale "vertically": if you outgrow your volume, you have to make it bigger, rather than splitting it into lots of little volumes. If you run out of space, it's up to you to provision more storage for the volume; if your disk can't keep up with the fast pace of writes to your busy database, you have to up the IOPS (input/output operations per second). That requires monitoring and general operational awareness, plus it can get expensive.

It's worth taking a moment as well to understand exactly what's going on when you initiate a snapshot of an EBS volume. EBS snapshots are incremental: one snapshot contains the changes from the snapshot before it. And each incremental snapshot represents the incremental change from one volume to the next. If you have three snapshots and roll back to the second one, you're deleting the difference in data between the second and third ones.

So if you were to say, "the third snapshot is my backup," that isn't necessarily true. Rolling back to that third snapshot will be reverting to the machine state at that time, but the third snapshot isn't a backup of all of your data.

But snapshots do tend to save money as they only incrementally increase the data that you have backed up, rather than back the entire environment up over and over. If you're regularly "backing up" your data, it makes more sense to take multiple snapshots rather than replicate the entire volume over and over again.

Our learners missed tough questions 48.6% of the time related to EBS volumes. Here are a few sample questions.

You are about to delete the second snapshot of an EBS volume that had 10 GiB of data when the first snapshot was taken. 6 GiB of that data had changed before the second snapshot was created. An additional 2 GiB of data have been added before the third and final snapshot was taken. Which of the following statements about that is correct? (36.4% correct).

You are a system administrator, and you need to take a consistent snapshot of your EC2 instance. Your application holds large amounts of data in the cache that is not written to disk automatically. What would be the best approach to taking an application-consistent snapshot? (44.9% correct).

Pain Forecast: ☀ Mostly Sunny

Amazon EC2

[amə̦zän, ē'sē'to͞o]

Azure *Azure Virtual Machines* | ***Google*** *Google Compute Engine*

WHAT IS IT? The granddaddy of cloud computing services for the better part of two decades, EC2 is a hosted version of the virtual machines that you might already be running in your data center - but backed by the power of Bezos' billions. Amazon EC2 reduces the time required to obtain and boot new server instances, allowing you to increase or decrease capacity as your needs change quickly. Hence the name "elastic."

But there's not just one kind of EC2 instance; you'll soon find there's an alphabet soup of instance types that are great for specific kinds of scenarios. You can use FPGA hardware for flexible machine learning problems, memory-optimized instances that are great for running databases, all the way down to tiny little A-series servers that are just right for taking your very first steps into the cloud.

WHY IS IT HARD? Unlike many cloud services, EC2 can feel deceptively *familiar* --it's not that hard to "lift and shift" workloads from your data center to EC2 without making a lot of changes along the way. And therein lies the problem: if you're not careful with spend management, EC2 can turn out to be less of a bargain than you hoped for, because you're renting the compute capacity you used to own.

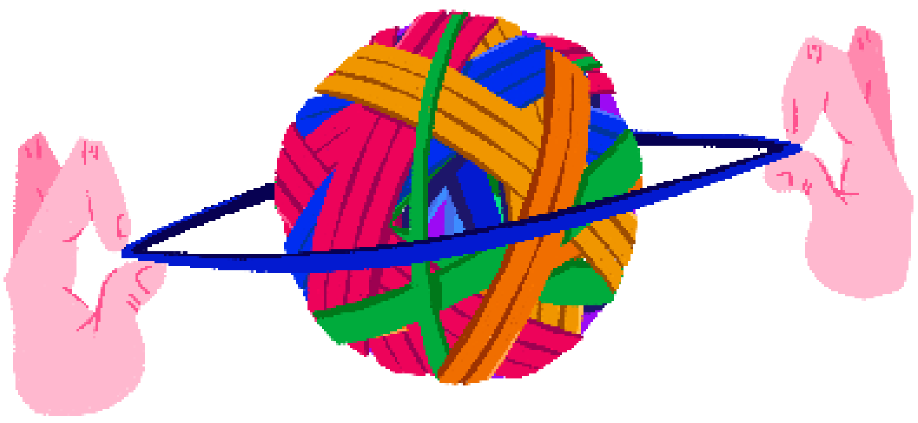
You'll have to consider pricing and how to maximize your efficiency while keeping your costs as low as possible. All this requires a clear understanding of what every team is working on; you'll need cross-functional collaboration as you consider the multifarious EC2 pricing options:

1 : On-demand: pay a fixed rate by the hour (or second). On-demand EC2 is useful for flexibility and avoiding long-term commitments.

2 : Reserved: you can reserve capacity on contracts for 1 or 3 years, which can save you a lot of money. So if you think you can accurately predict how quickly you're going to scale up or scale down, you'll be able to cut your costs.

3 : Spot-pricing: dirt-cheap server time based on AWS's excess capacity. But the price will move around just like the stock market—it'll go up and down. And spot instances can be yanked out from underneath your application with just a two-minute warning!

4 : Dedicated hosts: Physical EC2 hardware dedicated to you - no virtualized time-sharing with other AWS customers.Sometimes necessary if you have specific licensing or regulatory requirements.



Our learners missed tough questions 50.3% of the time related to EC2 instances. Here are a few sample questions.

You want to enable EC2 instances in your AWS environment to download software updates over HTTP port 80 from the internet. What security group settings will enable this? (31.3% correct)

A company is planning on hosting their website in AWS using eight EC2 instances. The host instances should not be accessible from the public internet, and all public traffic should be routed via an internet-facing application load balancer. What VPC design would implement these requirements? (32.1% correct)

Pain Forecast: Scattered Thunderstorm

Amazon Route 53 (DNS)

[amə͵zän, ro͞ot, fiftē-THrē]

Azure *Azure DNS* | ***Google*** *Cloud DNS*

WHAT IS IT? Route 53 is Amazon's DNS web service for AWS. It enables developers to help their users find the quickest route from the end user's computer to the server running an application.

Route 53 is particularly essential for Failover, or just switching to some redundant live systems to make sure your user can still use your app. You can use Route 53's health checks to route traffic to ensure that the user always arrives at a thing that is working.

Users may be accessing your app from all over the world, and Route 53 helps you figure out how to build the metaphorical freeway from their computer or phone (or fridge) across the internet to your service.

WHY IS IT HARD? When someone on your team brings up anything related to Route 53, your ears should perk up as it generally pertains to your services' overall health. You'll use Route 53 to route traffic with different routing types like latency or even how physically close they are to your server.

Sticking with the freeway analogy (which Amazon loves), Route 53 finds an alternate freeway if a sinkhole destroyed part of your primary freeway. The end-user shouldn't notice a difference because, well, it's the internet—and you'd probably get motion sickness anyway if you looked outside your window.

Our learners missed tough questions 49.7% of the time related to DNS and traffic management. Here are a few sample questions.

Your company is migrating a number of its applications to AWS. Services used will include Amazon EC2, Amazon S3, Amazon ELB application load balancers, NAT gateways, and Amazon RDS MySQL instances. You'll be using AWS's bring your own IP address (BYOIP) offering to keep all server IP addresses the same as they were on-premises. You'd also like to leverage Elastic IP addresses for failover scenarios. Which approach will provide the most reliable IP addressing for your new AWS environment? (33.8%)

A prescription drug company runs its in-house developed fulfillment application on EC2 with multiple instances behind an ELB network load balancer in a single AWS region. They'd like to failover to a different AWS region if issues arise with the application in the primary area. Security policy requires that both the primary and failover EC2 instances run in private subnets. Which architecture will provide the most reliable solution for failover between areas? (36.8%)

Pain Forecast: Thunderstorms

Amazon VPC

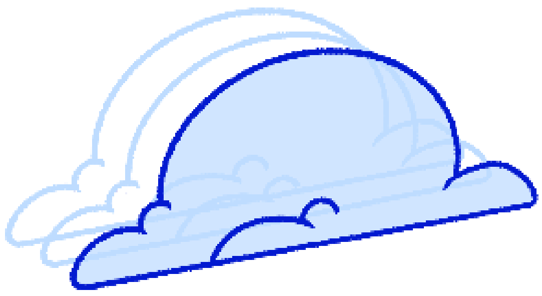
[amə͵zän, vē'pē'sē]

***Azure** Virtual Private Network* | ***Google** Also Virtual Private Network*

WHAT IS IT? Think of a VPC as a virtual data center in the cloud. Your data center may have public web servers that are directly accessible to the internet, and a more private set of servers that can only be accessed by direct connection or over a VPN. If you're dealing with especially sensitive customer information (such as something regulated), you'd want to isolate that data even more.

VPCs allow you to provision an isolated section of AWS where you can launch resources in a spot you define. So you can think of it as a way to do stuff with your private information without that confidential information touching the internet. VPCs are the place to put your database, your application servers, your back-end reporting processes - anything you don't want directly exposed to anyone with an internet connection.

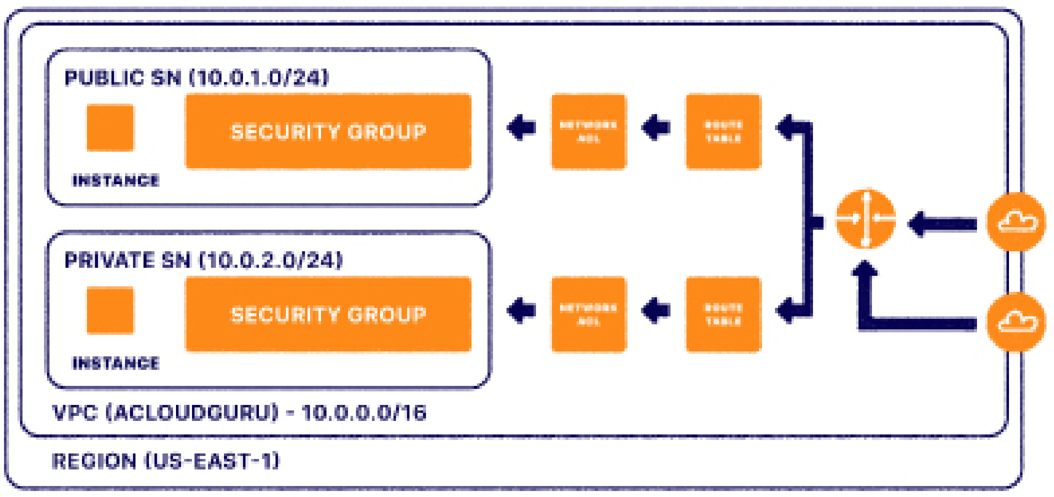
VPCs allow you to set your IP range, create subnets, and configure root tables and network gateways. You could create a public-facing subnet connected to the web while your backend systems are isolated and not connected to the internet at all. Multiple layers of security can help you control access to each subnet - what's known as "defense-in-depth".



Every AWS account gives you a default VPC out of the box, but you'll likely want to set up your own for improved security and customization.

WHY IS IT HARD? You're essentially trying to keep some information safe and off the internet while connecting internet-facing services to both that information the web simultaneously. That's complex! Just look at this flow chart:

VPC WITH PUBLIC & PRIVATE SUBNET(S)



All those elements are there to defend your information. But with all those arrows, you can see that there are many points where things can break down. The whole process can get particularly hairy when you're configuring a custom VPC but don't have a ton of expertise in networking.

Just one example: there is no transitive peering between VPCs. You can have one VPC talk to another VPC, but if that VPC talks to a third one, the first one can't speak to the third one. If VPC A can talk to VPC B, and B can talk to C, A still CAN NOT speak to C!

Increasingly, many people will also question whether they need a VPC in the first place. After all, the rise of "zero-trust" networking and serverless architectures has moved more workloads out of private networks and into reliance on good authorization strategies (like AWS IAM).

But the reality is that anyone trying to connect legacy services to the cloud will probably need to implement a VPC -- it's the only way to talk to the private network in your old data center, after all.

Our learners missed tough questions 50.3% of the time related to VPCs. Here are a few sample questions.

> Your company wants the on-premises network of its nearby branch office to securely connect with the instances launched into the amazon virtual private cloud VPC environment of its headquarters. Which of the following proposed solutions is the correct one? (27% correct)

> You are experiencing issues with HTTP traffic going through a network load balancer. You have recently set up access logging is enabled on the NLB and VPC. Flow logs have been configured. In this situation, which of the following monitoring tools is the most appropriate to help with troubleshooting HTTP client requests? (30.6% correct)

Pain Forecast: Mostly Sunny

Auto-Scaling

[amə̦zän, ē'sē'to͞o]

Azure *Azure Autoscale* | ***Google*** *also auto-scaling!*

WHAT IS IT? Auto-scaling helps you increase the number of servers in action to ensure your system can handle all the traffic coming its way There are three distinct autoscaling options on AWS at present:

> **1 :** Amazon EC2 Scaling focuses on EC2—and nothing else. **2 :** Application auto-scaling is used to control scaling for stuff that isn't EC2. You can use this when you're working with DynamoDB, elastic containers, Apache tools, and so on. **3 :** AWS Auto Scaling provides a scalable predictive feature and provides a central way to manage scalability.

When you're configuring an auto-scaling group, you'll want to think of four parameters: how many servers do you want to maintain uptime, do you want to adjust your server count manually, do you want to schedule when to scale up or down, or would you like it to be based on conditions with your product performance.

WHY IS IT HARD? Let's say you've just launched a new product, and suddenly you have a swarm of users frantically trying to pre-order it through your app or website. If you haven't configured your auto-scaling correctly—say, that you expected them to show at a different time instead of right at an announcement—you're about to collapse under the weight of your success.

Not only do you have to make the right call on which auto-scaling option you want to use, but you'll also have to ensure you pick the right route for auto-scaling. Having the wrong configuration or selections here can be quite costly, and you're probably going to be attempting a differential diagnosis of something right in the middle of some problem with your scaling.

Our learners missed tough questions 52.6% of the time related to auto-scaling. Here are a few sample questions.

An application server running in an autoscaling group is terminating and re-launching every few minutes. What is the most likely cause? (25.8% correct)

Your application runs in an auto-scaling group to scale based on user demand; the auto-scaling group runs behind an elastic load balancer. When you check the ELB logs, you notice that several instances are failing the health check during high demand periods. New models are launching, but they periodically fail health checks, and subsequent models are being launched, increasing costs. What would you do to troubleshoot this issue? (36.6% correct)

Pain Forecast: Scattered Thunderstorms

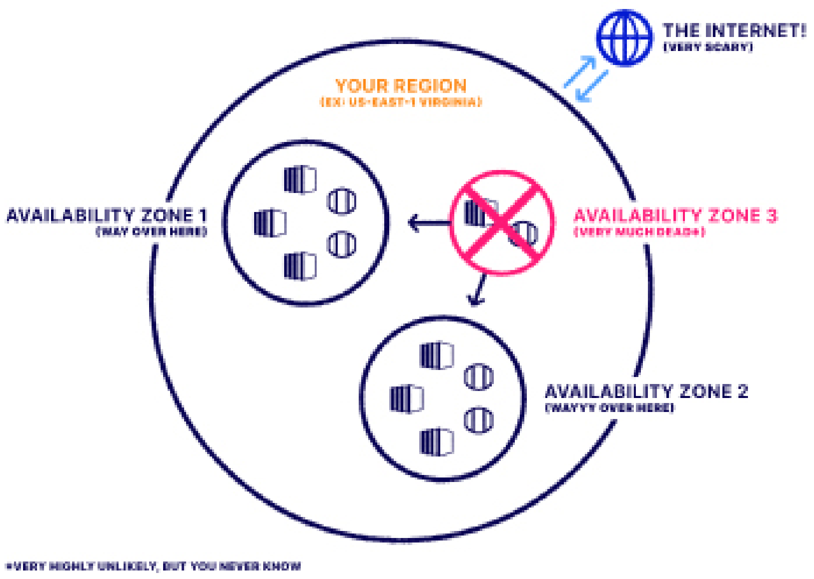
Availability Zones

[əˌvāləˈbilədē, zōns]

Azure *Azure Autoscale* | ***Google*** *also auto-scaling!*

WHAT IS IT? Each AWS region has multiple availability zones, which can contain several subnets. You can and should utilize multiple availability zones to create redundancy, allowing for high availability and fault tolerance. An availability zone consists of one or more AWS data centers. When you create a VPC, it spans all availability zones across a region, and you can add subnets to each availability zone.

Availability zones exist so AWS can ensure that its customers' applications are consistently available in the face of issues with the servers themselves. That could come in the form of component failure or a backhoe cutting the network link to a data center. Availability zones are situated geographically far enough apart that a localized problem shouldn't affect more than one at the same time.



WHY IS IT HARD? You can effectively use availability zones to create high availability and fault tolerance for your applications. High availability means that you're minimizing downtime, while fault tolerance means you can continue serving services even if things start to break down. By launching instances in multiple availability zones, you can protect your service from going down.

Say we have two availability zones, with instances in public subnets in each availability zone. You can configure your EC2 instance in one availability zone to automatically replicate over to another EC2 instance in another availability zone in the case of a failover scenario. But you still have to ensure you set up those backups in the first place.

Our learners missed tough questions 49.1% of the time related to availability zones. Here are a few sample questions.

> You work for a cosmetic company which has their production website on AWS. The site itself is in a two-tier configuration with web servers in the front end and database servers at the back end. The site uses elastic load balancing and auto-scaling. The databases maintain consistency by replicating changes to each other as and when they occur. The databases need to have extremely low latency. Your website needs to be highly redundant and must be designed so that if one availability zone goes offline and auto-scaling cannot launch new instances in the remaining availability zones, the site will not go offline. How can the current architecture be enhanced to ensure this? (45.6%)

> You are helping the application support team with their annual disaster recovery testing at an investment bank. The company hosts the primary production PostgreSQL in RDS multi-AZ deployment, with multiple applications running on a combination of EC2 and Lambda. They have asked you to help the team demonstrate the impact that a failed availability zone will have on the database. Which of the following do you suggest? (45.6%)

Pain Forecast: Severe Thunderstorms

AWS Lambda

[ā' dəbəlˌyo͞o' es, lamdə]

Azure *Azure Serverless* | ***Google*** *Cloud Functions*

WHAT IS IT? AWS Lambda lets you run code on a tiny piece of compute called an "execution environment", even more abstracted than an EC2 instance. (That's why Lambda-based systems are often called "serverless": there are still servers somewhere down there, you just don't have to know or care about them.)

You can trigger functions with events like a click of a button on your website or upload of a photo (or perhaps an A Cloud Guru lecture).

For example, when you talk to Alexa, your question might not be querying a specific relational database to check for the response. Instead it may be passing a request through AWS API Gateway to a Lambda function. And those Lambda functions can trigger another Lambda function. Once again, in very classic AWS fashion, it's Lambdas all the way down.

To be clear, this kind of scaling is different from auto-scaling if you have users spread across multiple servers. A million users hitting your site may trigger a million Lambda events. The difference between 1 user and 1 million users calling that action is a function of cost, not speed. You only pay for whenever your application executes code.

WHY IS IT HARD? It's a whole different paradigm! Lambda functions have short runtime limits, rely heavily on event-driven programming, and don't play well with some legacy technologies.

And yet many people refer to serverless architecture as the future of applications. There are a lot of reasons why that's the case. But from a business perspective (especially if you're the one with the team corporate card), the primary reason is serverless architectures can be an extraordinary cost saving. Lambda calls are free up to 1 million requests, and only 20 cents per million requests after that. And that's not considering all the sysadmin time you're freeing up.

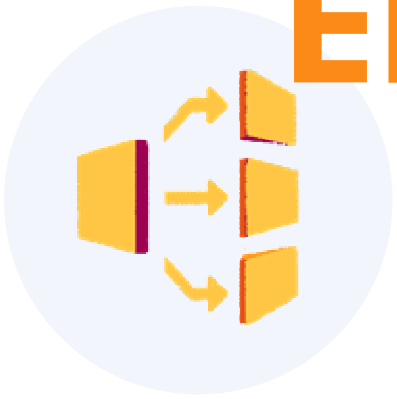
To put this into perspective: by using serverless and Lambda on the back end, our compute costs to serve millions of users here at A Cloud Guru became a rounding error. If you see an opportunity to offload some of your operations to Lambda for any reason, there's a good chance you probably should.

Our learners missed tough questions 48% of the time related to Lambda and serverless. Here are a few sample questions.

You are developing a serverless application written in Node.JS that will run on Lambda. During performance testing, you notice that the application is not running as quickly as you would like, and you suspect that your lambda function does not have enough CPU capacity. Which of the following options will improve the overall performance of your function? (34.3% correct)

You have several Cloudwatch log groups and Lambda functions that send logs to them. You need to use a tool to quickly search and analyze the log data in the log streams. The tool should automatically discover information in the Lambda logs, such as the timestamp, max memory used, and execution duration. It should also help you to perform the query using simple and pre-built query languages. Which tool is the best one for you to choose from? (38.2% correct)

Pain Forecast: Thunderstorm



Elastic Load Balancing

[əˈlastik, lōd, balənsING]

Azure *Azure Load Balancer* | ***Google*** *Cloud Load Balancing*

WHAT IS IT? It's exactly what it sounds like: a physical or virtual device designed to help you balance the network load across multiple servers. You have a variety of ways to plan out your auto-scaling and generally have three options:

1 : Application load balancers can see inside your application and the requests it's making. Application load balancers are great for a scaling engine that will make intelligent decisions for you. **2 :** Network load balancers work best when you need to focus on performance. These load balancers operate at the connection level. **3 :** Classic load balancers are just legacy elastic load balancers. This is a cheap option if you don't care how AWS routes your traffic. (Amazon has largely deprecated these.)

WHY IS IT HARD? Two reasons: you first have to select the best load balancer, and then you have a lot of specific features you can enable to make those load balancers more efficient. As always, there are going to be tradeoffs. The challenge is maximizing the value you get out of your cloud configurations.

Application load balancers are best-suited for HTTP and HTTPS traffic, operating at layer seven. Applications use Network load balancers to balance TCP traffic, working on the connection level (layer four). Legacy balancers can use layer seven-specific features for HTTP/HTTPS applications, but it isn't application-aware.

Once you've selected a load balancer, you're going to have to address how you'd like to route your traffic to your various EC2 instances. These instances may be across multiple availability zones and tied to different load balancers. But it is possible to intelligently route traffic to optimize your performance for each EC2 instance.

Here are a few top-level configurations you'll have to address:

STICKY SESSIONS : You may want to ensure that one user is only ever tied to one EC2 instance—such as if they're saving information to disk. But you may also end up with unused EC2 instances by using sticky sessions.

CROSS-ZONE LOAD BALACING : You may want to automatically balance your incoming traffic across all EC2 instances regardless of the availability zone you're currently using. You may sacrifice some performance by sending your traffic into different geographic regions if you use cross-zone load balancing.

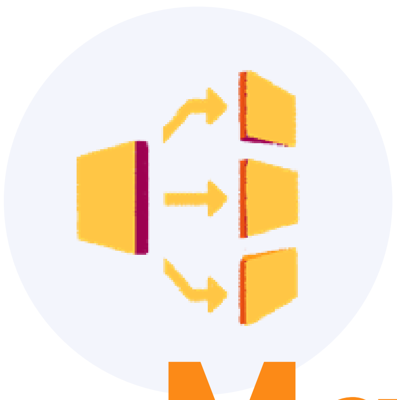
PATH PATTERNS : You can route your traffic intelligently based on URL paths, such as myurl.com/images. But this may also not account for surges in traffic to specific pages—like, say, some celebrity posts an image on your images page.

Our learners missed tough questions 53% of the time related to load balancers. Here are a few sample questions.

> An organization needs its first cloud-based application to be available on the same static IP address at all times due to the nature of its security posture. Which load balancer configuration will deliver this outcome? (23.3% correct)

> You are experiencing issues with an application load balancer you have recently set up and have not changed any of the default logging settings in this situation. Which monitoring tools are most appropriate to help with troubleshooting client requests? (34% correct)

Pain Forecast: Scattered Thunderstorm



Identity & Access Management

[īden(t)ədē, and, akˌses, manijmənt]

***Azure** Identity And Access Management* | ***Google** Cloud Identity And Access Management*

WHAT IS IT? IAM allows you to manage users and their level of access to an AWS console. It will enable you to set up users, groups, permissions, and roles. You can grant access to different parts of the AWS platform—the keyword here being parts. You can set up very granular permissions down to an individual user getting access to one service and not another.

IAM is core to any cloud usage, from giving developers access to resources to push updates to production or giving auditors access to inspect your work. And IAM is a global service -- IAM resources are managed at the AWS account level, not just for specific regions or availability zones.

Ultimately, IAM is how AWS resources speak to each other, how you audit them, and how you control access for your developers to update them. And that means that getting IAM right has implications not just for the security team...but for everybody.

WHY IS IT HARD? AWS IAM is a bespoke system, built from the ground up to handle the authentication and authorization challenges of massively distributed cloud applications. So even though it's foundational to the success of your AWS deployment, developers often have to learn it from scratch; it's not quite like anything you had back in the data center.

You should always set up multi-factor authentication on root AWS accounts and customize password rotations. But if you're a little too cavalier when using identity federation, one compromised account could end up leading to a breach across your entire AWS footprint. And "long-lived" access and secret keys are easy to provision for users, but if those keys leak, you could see an outside hacker assuming the rights of your own employees!

Ultimately, there's no substitute for a careful plan when it comes to laying out your AWS IAM strategy. Infrastructure as code, federated identities, and properly restrictive policies aren't simple defaults to implement -- but they'll go a long way toward ensuring that your systems have exactly the access they need.

Our learners missed tough questions 50.5% of the time related to IAM. Here are a few sample questions.

> Your organization is going through a security audit with an external party. As the security administrator for your AWS account, you are tasked with providing three auditors permissions to view and access resources in the AWS account. You have set up a trust policy between your account and the external AWS account and have created an IAM role that the auditors can assume. How would you specify the principal element on the IAM policy to grant auditors access? (29% correct)

> You are experiencing issues with an application load balancer you have recently set up and have not changed any of the default logging settings in this situation. Which monitoring tools are most appropriate to help with troubleshooting client requests? (34% correct)

Pain Forecast: ☼ Mostly Sunny

Relational Database Service (RDS)

[rə'lāSH(ə)n(ə)l, dadəˌbās, sərvəs]

Azure *Also RDS* | ***Google*** *CloudSQL*

WHAT IS IT? A managed database that, well, manages everything. RDS abstracts out the whole process of running and maintaining a database so you are only engaging it when you need to read or write data. CPU, memory, storage, and IOPS are split apart. You can scale them independently—bringing each one up or down. The aim is to lower the overall cost of ownership and strip away routine tasks. That can include provisioning, backup, recovery, and many other core requirements for any system. As a result, you can focus on higher-level tasks that pertain directly to your business.

WHY IS IT HARD? RDS gives you a lot of database engines to choose from! There are commercial databases like Oracle and Microsoft SQL Server. There are open-source engines like MariaDB, Postgres, and MySQL. And then there are cloud-native solutions like Amazon Aurora.

RDS also is not a magic bullet to getting everything else off your plate. You'll still have a lot of things to consider, such as: **1 :** What database engine are you using? **2 :** How are you handling backups and monitoring? **3 :** What size will your underlying storage and compute be?

And when there's a lot to configure, especially when you're planning for a set-it-and-forget-it tool, there's a lot that can go wrong—especially as you scale up. When things start to go south, you're going to have to dig back through and reconfigure some of your setups (and hope you aren't trying to repair something while it's already in action).

Our learners missed 49.7% of the tough questions related to RDS. Here are a few sample questions.

> You're auditing your RDS Estate and discover an RDS production database that is not encrypted at rest. This violates company policy, and you need to rectify this immediately. What should you do to encrypt the database as quickly and as efficiently as possible? (29.2% correct.)

> A company's disaster recovery policy requires that all RDS backups are retained in a secondary AWS region. What is the optimal solution to meet this requirement? (29.9% correct.)

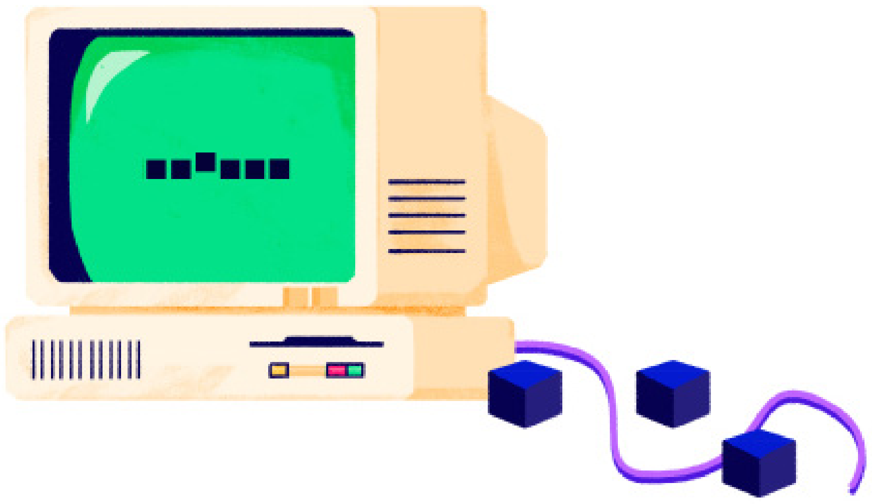
Pain Forecast: Severe Thunderstorms

Simple Queue Services (Amazon SQS)

[simpəl, kyōō, sərvəses]

WHAT IS IT? SQS is one of AWS's oldest services, first announced in 2004. It gives you access to a message queue to store messages while waiting for a computer to process them. Amazon SQS enables you to decouple components of an application and let them communicate asynchronously (on their own schedules). Each message is sent to a service like a Lambda function or EC2 instance.

Those decoupled components can store their messages in a fail-safe queue that will wait for computation. This can be really useful if you, for example, lose an availability zone for whatever reason. The message becomes available in the queue if an EC2 instance goes down, enabling other EC2 instances to pick up the slack.



WHY IS IT HARD? If you're building on SQS, you're by definition creating a distributed application: one that does work across a network via decoupled pieces. And that brings with it a whole new layer of design and operational considerations.

You can choose whether you want to create a standard queue or a FIFO (first-in-first-out) queue. Both have some advantages:

Standard queues ensure that your message is delivered at least once and make a best effort to arrange them in the same order as received. But that's a best effort, and you shouldn't assume order!

FIFO queues deliver messages in the exact order as they were received. The message is delivered and remains available until a consumer processes it and deletes it. Duplicates are not allowed into the queue in this case.

SQS also implements visibility timeout—essentially how long the queue waits for one consumer before making it available to the next one. For example, let's say a message is made available to one EC2 instance and isn't executed in some target time. That message will then be available for another EC2 instance to grab it and run it. But this also means SQS may deliver your message more than once.

Our learners missed tough questions 49% of the time related to SQS. Here are a few sample questions.

> Your application reads data from an SQS queue. SQS then forwards the reads to Lambda downstream for processing critical customer information, including purchasing and inventory data. It is critical that this data is not lost. How would you accommodate failed Lambda captures of the data? (38.2% correct)

> At your company, the customer service organization just told you that a client purchase from your website was processed twice. Your order process involves EC2 instances processing messages from an SQS queue. What changes might you make to ensure this does not happen again? (39.5% correct)

Pain Forecast: Rain

Subnets

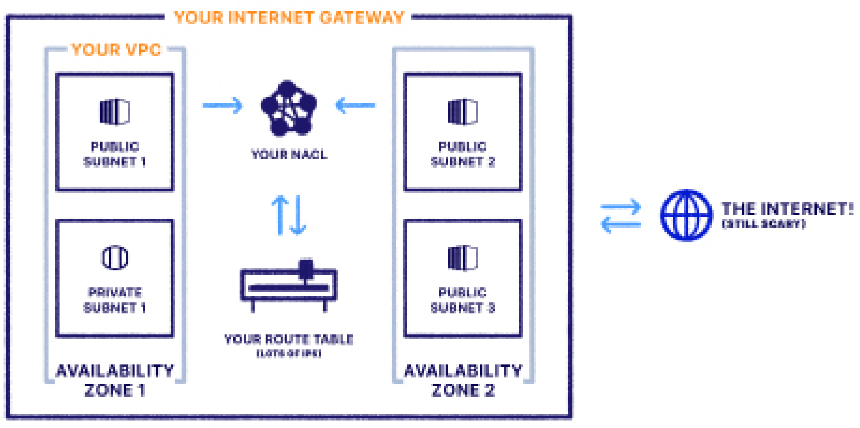
[səb'net]

Azure *Azure Virtual Network* | ***Google*** *VPC Virtual Network*

WHAT IS IT? Shorthand for subnetwork, a subnet is just a subsection of a network. When you create a VPC, you'll have a series of subnets associated with all the applications within a specific availability zone. We'll provision resources, like an EC2 instance or an RDS database, within particular subnets—which can be public or private.

Public subnets have a route to the internet that's associated with an internet gateway. Public subnets can also talk to other public subnets. Private subnets do not have a path to the internet, but they can connect to public subnets within your VPC.

WHY IS IT HARD? You can't have one subnet across multiple availability zones. You'll probably hear something along the lines of "one subnet equals one availability zone." Let's say you've decided to launch a VPC within a particular region, and within that region, AWS offers a set of availability zones. If you'd like to keep some information private—such as a set of customer information in an RDS database—you would launch a private subnet within one availability zone.



However, suppose you wanted to launch a subnet within a different availability zone (such as looking for some redundancy). In that case, that subnet will not be able to talk to your private subnet in a different availability zone. Your private subnet won't span multiple availability zones. So this makes it an important consideration when figuring out how to handle disaster recovery.

Our learners missed tough questions 49.1% of the time related to subnets. Here are a few sample questions.

You have a three-tier application that you want to deploy into AWS. This application is accessed by users around the world over the internet. The design calls for an application load balancer, EC2 instances for the application software, and another set of EC2 instances to run the custom relational database system for the application periodically. You want the instances to download updates from the internet via an already-deployed gateway in the public subnet. Which of the below deployments would you recommend keeping in mind that you want to keep costs and complexity to a minimum? (39.3%)

A CLOUD GURU

Cloud Smarter. Cloud Better. Learn with A Cloud Guru.

Since 2015, we've helped more than two million engineers learn to cloud with approachable courses taught by industry experts. Scale your teams, get them certified, generate value, and learn a ton of cool stuff along the way.

Accelerate Adoption
Learn the critical skills you need to supercharge your products with top cloud tools and technology.

Speak Cloud
Our scalable, sprint-based program lifts everyone to a common base of knowledge quickly and effectively.

Stay Informed
See the latest trends in the way learners build their cloud knowledge based on our research.

Learn By Doing
Give your team valuable hands-on experience—and let them break everything—without exposing your production environments.

Keep Growing
Provide ongoing, guided learning that takes your employees from novice to guru in specialized cloud career tracks.

Ready to take off? We'd love to help you reach the sky.

Request A Demo