Logo

AWS CSA Training

Gaurav Manchanda
Solution Architect
Date

Sample Categories with Bulleted Information

AWS CSA-

Associate

Topic Covered

- Introduction to Cloud Computing and AWS
- Amazon Elastic Compute Cloud and Amazon Elastic Block Store
- Amazon Simple Storage Service and Amazon Glacier Storage
- Amazon Virtual Private Cloud
- Databases
- AWS Identity and Access Management
- CloudTrail, CloudWatch, and AWS Config
- The Domain Name System and Network Routing: Amazon Route 53 and Amazon CloudFront

Exam Objectives

- The AWS Certified Solutions Architect – Associate exam is intended for people who have experience in designing distributed applications and systems on the AWS platform.

In general, you should have the following before taking the exam:

- A minimum of one year of hands-on experience designing systems on AWS
- Hands-on experience using the AWS services that provide compute, networking, storage, and databases
- Ability to define a solution using architectural design principles based on customer requirements.
- Ability to provide implementation guidance
- Ability to identify which AWS services meet a given technical requirement
- An understanding of the five pillars of the Well-Architected Framework
- An understanding of the AWS global infrastructure, including the network technologies used to connect them
- An understanding of AWS security services and how they integrate with traditional on-premises security infrastructure
- The exam covers five different domains, with each domain broken down into objectives.

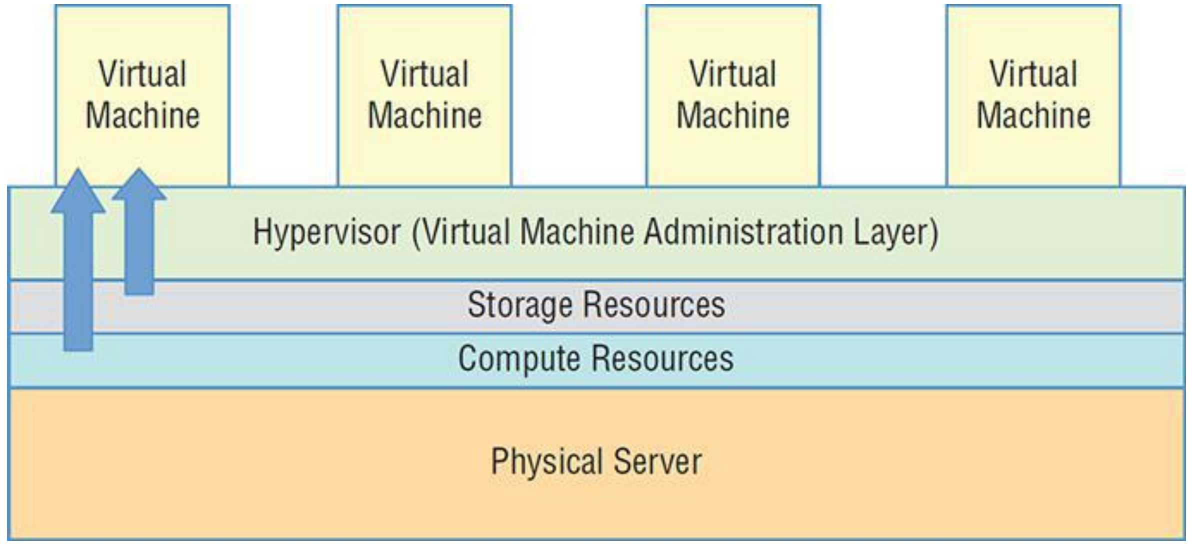
AWS

- Introduction to Cloud Computing and AWS
- Amazon Elastic Compute Cloud and Amazon Elastic Block Store
- Amazon Simple Storage Service and Amazon Glacier Storage
- Amazon Virtual Private Cloud
- Databases
- Authentication and Authorization—AWS Identity and Access Management
- CloudTrail, CloudWatch, and AWS Config
- The Domain Name System and Network Routing—Amazon Route 53 and Amazon CloudFront

Introduction to Cloud Computing and AWS

Cloud Computing and Virtualization

- Virtualization lies at the core of all the cloud operations
- Lets you divide the hardware resources of a single physical server into smaller units
- Physical server could therefore host multiple virtual machines running their own complete operating systems, each with its own memory, storage, and network access.
- Provision a virtual server in a matter of seconds, run it for exactly the time your project requires, and then shut it down.
- The resources released will become instantly available to other workloads.
- Make the greatest value from your hardware and makes it easy to generate experimental and sandboxed environments.



Introduction to Cloud Computing and AWS

Cloud Computing Architecture

– A cloud computing platform offers on-demand, self-service access to pooled compute resources where your usage is metered and billed according to the volume you consume.

– Cloud computing systems allow for precise billing models, sometimes involving fractions of a penny for an hour of consumption

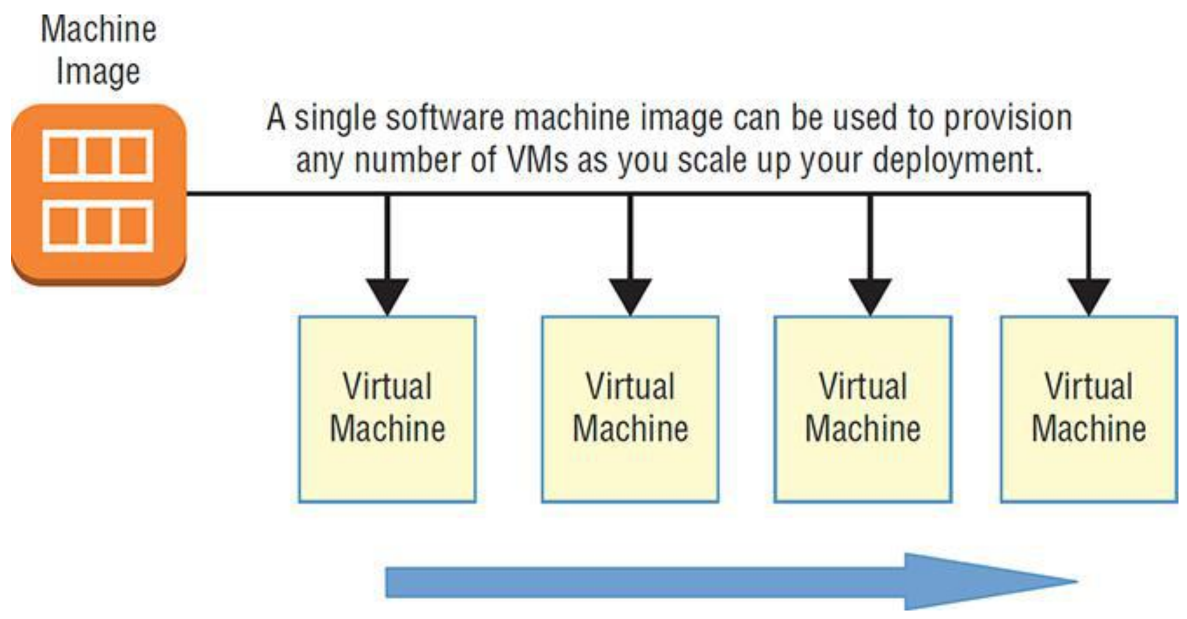
Introduction to Cloud Computing and AWS

Cloud Computing Optimization

– Effective deployment provisioning requires following three features.

Scalability

– A scalable infrastructure can efficiently meet unexpected increases in demand for your application by automatically adding resources


Machine Image
A single software machine image can be used to provision any number of VMs as you scale up your deployment.
Virtual Machine
Virtual Machine
Virtual Machine
Virtual Machine

Introduction to Cloud Computing and AWS

Elasticity

- The principle of elasticity covers some of the same ground as scalability—both address managing changing demand. However, an elastic infrastructure will automatically *reduce* capacity when demand drops.
- This makes it possible to control costs, since you'll run resources only when they're needed.
- *Think of a Rubber Band.*

Cost Management

- Cloud computing transitions your IT spending from a capital expenditure (capex) framework into something closer to operational expenditure (opex).
- You won't have to expose yourself to risky speculation about your long-term needs. If, sometime in the future, changing demand calls for new hardware, AWS will be able to deliver it within a minute or two.
- AWS provides a free Total Cost of Ownership (TCO) Calculator at https://aws.amazon.com/tco-calculator/.
- Provide comparisons between your current data center costs and what an identical operation would cost you on AWS.

Introduction to Cloud Computing and AWS

The AWS Cloud

As a solutions architect, your main focus should be on the core service categories.

Category	Function
• Compute	Services replicating the traditional role of local physical servers for the cloud, offering advanced configurations including autoscaling, load balancing, and even serverless architectures (a method for delivering server functionality with a very small footprint)
• Networking	Application connectivity, access control, and enhanced remote connections
• Storage	Various kinds of storage platforms designed to fit a range of both immediate accessibility and long-term backup needs
• Database	Managed data solutions for use cases requiring multiple data formats: relational, NoSQL, or caching
• Application management	Monitoring, auditing, and configuring AWS account services and running resources
• Security and identity	Services for managing authentication and authorization, data and connection encryption, and integration with third-party authentication management systems
• Application integration	Tools for designing loosely coupled, integrated, and API-friendly application development processes

Introduction to Cloud Computing and AWS

Compute

Service	Function
Elastic Compute Cloud (EC2)	EC2 server instances provide virtual versions of the servers you would run in your local data center. EC2 instances can be provisioned with the CPU, memory, storage, and network interface profile to meet any application need, from a simple web server to one part of a cluster of instances providing an integrated multitiered fleet architecture. Since EC2 instances are virtual, they're much more resource-efficient and deploy nearly instantly.
Lambda	Lambda provides serverless application architectures that allows you to, provide responsive public-facing services without the need for a server that's actually running 24/7. Instead, network events (like consumer requests) can trigger the execution of a predefined code-based operation. When the operation (which can currently run for as long as 15 minutes) is complete, the Lambda event ends, and all resources automatically shut down.
Auto Scaling	Copies of running EC2 instances can be defined as image templates and automatically launched (or *scaled up*) when client demand increases. As demand drops, unused instances can be terminated (or *scaled down*).
Elastic Load Balancing	Incoming network traffic can be directed between multiple web servers to ensure that a single web server isn't overwhelmed while other servers are underused or that traffic isn't directed to failed servers.
Elastic Beanstalk	Beanstalk is a managed service that abstracts the provisioning of AWS compute and networking infrastructure. You are required to do nothing more than push your application code, and Beanstalk automatically launches and manages all the necessary services in the background.

Introduction to Cloud Computing and AWS

Networking

Service	Function
Virtual Private Cloud (VPC)	VPCs are highly configurable networking environments designed to host your EC2 and RDS instances. You use VPC-based tools to closely control inbound and outbound network access to and between instances.
Direct Connect	By purchasing fast and secure network connections to AWS through a third-party provider, you can use Direct Connect to establish an enhanced direct tunnel between your local data center or office and your AWS-based VPCs.
Route 53	Route 53 is the AWS DNS service that lets you manage domain registration, record administration, routing protocols, and health checks, which are all fully integrated with the rest of your AWS resources
CloudFront	CloudFront is Amazon's distributed global content delivery network (CDN). It stores cached versions of your site's content at edge locations around the world so they can be delivered to customers on request with the greatest efficiency and lowest latency.

Introduction to Cloud Computing and AWS

Storage

Service	Function
Simple Storage Service (S3)	S3 offers highly versatile, reliable, and inexpensive object storage that's great for data storage and backups. It's also commonly used as part of larger AWS production processes, including through the storage of script, template, and log files.
Glacier	When you need large data archives stored cheaply over the long term. It requires long retrieval time. Glacier's lifecycle management is closely integrated with S3.
Elastic Block Store (EBS)	EBS provides the virtual data drives that host the operating systems and working data of an EC2 instance. Think of it as a storage drives and partitions attached to physical servers.
Storage Gateway	Storage Gateway is a hybrid storage system that exposes AWS cloud storage as a local, on-premises appliance. It can be a great tool for migration and data backup and as part of disaster recovery operations.

Introduction to Cloud Computing and AWS

Database and Application Management

Service	Function
Relational Database Service (RDS)	RDS is a managed service that builds you a stable, secure, and reliable database instance. You can run a variety of SQL database engines on RDS, including MySQL, Microsoft SQL Server, Oracle, and Amazon's own Aurora.
DynamoDB	DynamoDB can be used for fast, flexible, highly scalable, and managed nonrelational (NoSQL) database workloads.
	Application Management
CloudWatch	No deployment is complete without some kind of ongoing monitoring in place. And generating endless log files doesn't make much sense if there's no one keeping an eye on them. CloudWatch can be set to monitor process performance and utilization through events and, when preset thresholds are met, either send you a message or trigger an automated response.
CloudFormation	It enables you to use template files to define full and complex AWS deployments. The ability to script your use of any AWS resources makes it easier and more attractive to automate, standardizing and speeding up the application launch process.
CloudTrail	CloudTrail collects records of all your account's API events. This history is useful for account auditing and troubleshooting purposes.
Config	The Config service is designed to help you with change management and compliance for your AWS account. You first define a desired configuration state, and Config will get to work evaluating any future states against that ideal. When a configuration change pushes too far from the ideal baseline, you'll be notified.

Introduction to Cloud Computing and AWS

Security Management & Application Integration

	Service	Function
Security and identity	Identity and Access Management (IAM)	You use IAM to administrate user and programmatic access and authentication to your AWS account. Through the use of users, groups, roles, and policies, you can control exactly who and what can access and/or work with any of your AWS resources.
	Key Management Service (KMS)	KMS is a managed service that allows you to administrate the creation and use of encryption keys to secure data used by and for any of your AWS resources.
	Directory Service	For AWS environments that need to manage identities and relationships, Directory Service can integrate AWS resources with identity providers like Amazon Cognito and Microsoft AD domains.
Application integration	Simple Notification Service (SNS)	SNS is a notification tool that can automate the publishing of alert *topics* to other services (to an SQS Queue or to trigger a Lambda function, for instance), to mobile devices, or to recipients using email or SMS.
	Simple WorkFlow (SWF)	SWF lets you coordinate a series of tasks that must be performed using a range of AWS services or even nondigital (meaning, human) events. SWF can be the "glue" and "lubrication" that both speed a complex process and keep all the moving parts from falling apart. (think of shopping workflow of AWS)
	Simple Queue Service (SQS)	SQS allows for event-driven messaging within distributed systems that can decouple while coordinating the discrete steps of a larger process. The data contained in your SQS messages will be reliably delivered, adding to the fault-tolerant qualities of an application. (message que)
	API Gateway	This service enables you to create and manage secure and reliable APIs for your AWS-based applications.

Introduction to Cloud Computing and AWS

AWS Platform Architecture

- AWS maintains data centers for its physical servers around the world.
- It reduce your own services’ network transfer latency by hosting your workloads geographically close to your users.
- It help you manage compliance with regulations requiring you to keep data within a particular legal jurisdiction.
- Data centers exist within AWS regions, of which there are currently 17—not including private U.S. government AWS GovCloud regions—*** this number is constantly growing.
- Certain AWS services are offered from designated edge network locations. Eg Amazon CloudFront, Amazon Route 53, AWS Firewall Manager, AWS Shield, and AWS WAF.
- You organize your resources from a particular region within one or more virtual private clouds (VPCs). A VPC is effectively a network address space within which you can create network subnets and associate them with particular availability zones. When configured properly, this architecture can provide effective resource isolation and durable replication

Introduction to Cloud Computing and AWS

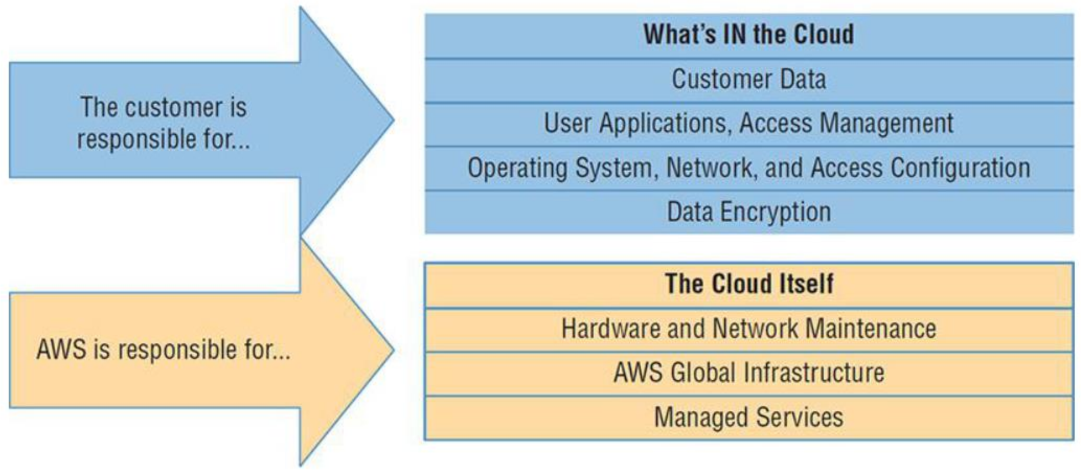
- **The AWS Shared Responsibility Model**

  AWS guarantees the secure and uninterrupted operation **of its "cloud."**

  - physical servers,
  - storage devices,
  - networking infrastructure,
  - managed services.

  AWS customers are responsible for whatever happens ***within* that cloud**.

  - the security and operation of installed operating systems,
  - client-side data,
  - the movement of data across networks,
  - end-user authentication and access,
  - customer data.


The customer is responsible for...
What's IN the Cloud
Customer Data
User Applications, Access Management
Operating System, Network, and Access Configuration
Data Encryption
AWS is responsible for...
The Cloud Itself
Hardware and Network Maintenance
AWS Global Infrastructure
Managed Services

Introduction to Cloud Computing and AWS

- **Working with AWS**

Console

The AWS CLI

- The AWS Command Line Interface (CLI) lets you run complex AWS operations from your local command line.
- **Ie** simple script that you can run from your local terminal shell or PowerShell interface using the AWS CLI.
- For Installing and configuring the AWS CLI on Linux, Windows, or macOS machines see https://docs.aws.amazon.com/cli/latest/userguide/installing.html.

AWS SDKs

– If you want to *incorporate access to your AWS resources into your application code*, you'll need to use an AWS software development kit (SDK) for the language you're working with.

– AWS currently offers SDKs for nine languages including Java, .NET, and Python, and a number of mobile SDKs that include Android and iOS.

– There are also toolkits available for Eclipse, Visual Studio, and VSTS.

Exam Essentials

Understand the global infrastructure.

Understand AWS platform architecture.

- AWS divides its servers and storage devices into globally distributed regions and, within regions, into availability zones.
- This division permit replication to enhance availability
- Permits process and resource isolation for security and compliance purposes.
- You should design your own deployments in ways that take advantage of these features

Understand how to use AWS administration tools.

- While you will use your browser to access the AWS administration console at least from time to time, most of your serious work will probably happen through the AWS CLI and, from within your application code, through an AWS SDK.

Understand how to choose a support plan.

- Understanding which support plan level is right for a particular customer is an important element in building a successful deployment. Become familiar with the various options

Amazon Elastic Compute Cloud and Amazon Elastic Block Store

EC2 Instances

An EC2 instance may only be a virtualized and abstracted subset of a physical server, but it behaves just like the real thing. It'll have access to storage, memory, and a network interface, and its primary drive will come with a fresh and clean operating system running.

You can decide hardware resources for your instance ,operating system and software stack, and, payment plan

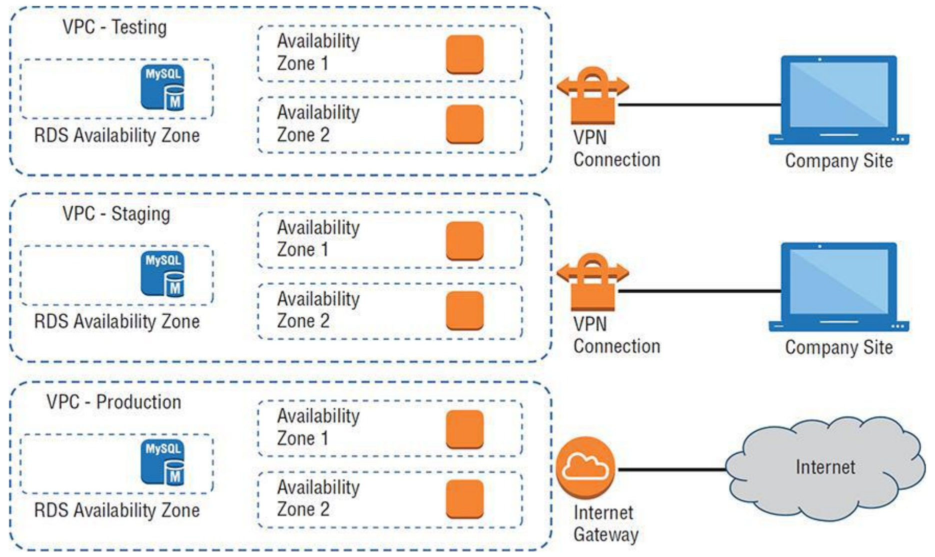
Provisioning Your Instance

- EC2 Amazon Machine Images
- Amazon Quick Start AMIs
- AWS Marketplace AMIs
- Community AMIs
- Private AMIs

Instance Types

Instance Type Family	Types	Usage
General purpose	T3, T2, M5, M4	• Provide a balance of compute, memory, and network resources • T2s are burstable, you can accumulate CPU credits when your instance is underutilized that can be applied during high-demand periods in the form of higher CPU performance. • M5 and M4 instances are recommended for many small and midsize data-centric operations.
Compute optimized	C5, C4	• For more demanding web servers and high-end machine learning workloads
Memory optimized	X1e, X1, R5, R4, z1d	• work well for intensive database, data analysis, and caching operations
Accelerated computing	P3, P2, G3, F1	• for demanding workloads such as 3D visualizations and rendering, financial analysis, and computational fluid dynamics.
Storage optimized	H1, I3, D2	• work well with distributed file systems and heavyweight data processing applications

Amazon Elastic Compute Cloud and Amazon Elastic Block Store



Configuring an Environment for Your Instance

Three primary details to get right

Geographic region

- launch an EC2 instance in the region that's physically closest to the majority of your customers
- within a jurisdiction that meets your compliance needs.

Virtual private cloud (VPC)

- easy-to-use AWS network organizers and great tools for organizing your infrastructure
- easy to isolate the instances in one VPC from whatever else you have running,
- you can create a new VPC for each one of your projects or project stages.

Tenancy

- Shared tenancy (default)
  - » instance will run as a virtual machine on a physical server that's concurrently hosting other instances.
- Dedicated Instance
  - » ensures that your instance will run on its own dedicated physical server.
  - » allows to identify and control the physical server you've been assigned to meet more restrictive licensing or regulatory requirements

Amazon Elastic Compute Cloud and Amazon Elastic Block Store

Configuring Instance Behavior

Bootstrapping

- Execute commands on your instance as it boots by pointing to user data in your instance configuration
- You can have script files bring your instance to any desired state
- User data can consist of a few simple commands to install a web server and populate its web root,etc..

Amazon Elastic Compute Cloud and Amazon Elastic Block Store

Instance Pricing

On-demand model

- For always-on deployments that you expect to run for less
- closely control how much you pay by stopping and starting your instances according to your need
- most expensive.

Reserve instance

- Compute need of 24/7 for more than a year (steady state)
- significantly discounted

Amazon's Spot market

- For workloads that can withstand unexpected disruption (like computation-intensive genome research applications
- The idea is that you enter a maximum dollar-value bid for an instance type running in a particular region. The next time an instance in that region becomes available at a per-hour rate that's equal to or below your bid, it'll be launched using the AMI and launch template you specified. Once up, the instance will keep running either until you stop it—when your workload completes, for example—or until the instance's per-hour rate rises above your maximum bid

Note : Always combine multiple models within a single application infrastructure

Amazon Elastic Compute Cloud and Amazon Elastic Block Store

Instance Lifecycle

- Terminating the instance will shut it down and cause its resources to be reallocated to the general AWS pool.
- This will, in most cases, destroy all data kept on the primary storage. The exception to this would be an Elastic Block Store (EBS) volume that has been set to persist after its instance is terminated
- If your instance won't be needed for some time but you don't want to terminate it, you can save money by simply stopping it and then restarting it when it's needed again. The data on an EBS volume will in this case not be lost, although that would not be true for an instance volume.
- a stopped instance that had been using a nonpersistent public IP address will most likely be assigned a different address when it's restarted. If you need a predictable IP address that can survive restarts, allocate an Elastic IP address and associate it with your instance.
- You can change its instance type to increase or decrease its compute, memory, and storage capacity. You will need to stop the instance, change the type, and then restart it.

Resource Tags

- AWS resource tags can be used to label everything you'll ever touch across your AWS account
- Tags have a key and, optionally, an associated value.

Key	Value
`production-server`	`server1`
`production-server`	`server2`
`production-server`	`security-group1`
`staging-server`	`server1`
`staging-server`	`server1`
`staging-server`	`staging-security-group1`
`test-server`	`server1`
`test-server`	`test-security-group1`

Amazon Elastic Compute Cloud and Amazon Elastic Block Store

EC2 Storage Volumes

Elastic Block Store Volumes

- attach as many Elastic Block Store (EBS) volumes to instance
- Use it as a hard drives, flash drives, or USB drives with your physical server
- SLA of at least 99.999 percent availability

Four EBS volume types

EBS-Provisioned IOPS SSD

- If your applications require intense rates of I/O operations
- provides a maximum IOPS/volume of 32,000
- maximum throughput/volume of 500 MB/s.

EBS General-Purpose SSD

- For most regular server workloads that, ideally, deliver low-latency performance
- Provide maximum of 10,000 IOPS/volume

Amazon Elastic Compute Cloud and Amazon Elastic Block Store

- **EC2 Storage Volumes**
- **Throughput-Optimized HDD**
  - provide reduced costs with acceptable performance
  - Suited for throughput-intensive workloads including log processing and big data operations
  - deliver only 500 IOPS/volume but with a 500 MB/s maximum throughput/volume
- **Cold HDD**
  - Suitable for working with larger volumes of data that require only infrequent access
  - Deliver 250 IOPS/volume type

	EBS-Provisioned IOPS SSD	**EBS General-Purpose SSD**	**Throughput-Optimized HDD**	**Cold HDD**
Volume size	4 GB-16 TB	1 GB-16 TB	500 GB-16 TB	500 GB-16 TB
Max IOPS/volume	32,000	10,000	500	250
Max throughput/volume (MB/s)	500	160	500	250
Price (/month)	$0.125/GB + $0.065/prov IOPS	$0.10/GB	$0.045/GB	$0.025/GB

Amazon Elastic Compute Cloud and Amazon Elastic Block Store

Elastic Block Store Volumes Features

- can be copied by creating a snapshot
- snapshots can be used to generate other volumes that can be shared and/or attached to other instances or converted to images from which AMIs can be made.
- generate an AMI image directly from a running instance-attached EBS volume—(to sure no data is lost, it's best to shut down the instance first).
- can be encrypted to protect their data while at rest or as it's sent back and forth to the EC2 host instance
- EBS can manage the encryption keys automatically behind the scenes or use keys that you provide through the AWS Key Management Service (KMS).

Amazon Elastic Compute Cloud and Amazon Elastic Block Store

Instance Store Volumes

- instance store volumes are ephemeral (when the instances they're attached to are shut down, their data is permanently lost)
- Instance store volumes are SSDs that are physically attached to the server hosting your instance and are connected via a fast NVMe interface
- The use of instance store volumes is included in the price of the instance itself.
- work especially well for deployment models where instances are launched to fill short-term roles (as part of autoscaling groups, for instance), import data from external sources, and are, effectively, disposable.

Amazon Elastic Compute Cloud and Amazon Elastic Block Store

Accessing Your EC2 Instance

- EC2 instances are identified by unique IP addresses. All instances are assigned at least one private IPv4 address
- Out of the box, you'll only be able to connect to your instance from within its subnet, and the instance will have no direct contact to the Internet.
- an instance can also be assigned a public IP through which full Internet access is possible.
- the default public IP assigned to your instance is ephemeral and probably won't survive a reboot. Therefore, you'll usually want to allocate a permanent elastic IP for long-term deployments.
- As long as it's attached to a running instance, there's no charge for elastic IPs.
- Running the curl command from the command line while logged into the instance will return a list of the kinds of data that are available: $ curl http://169.254.169.254/latest/meta-data/ ami-id etc

Amazon Elastic Compute Cloud and Amazon Elastic Block Store

Securing Your EC2 Instance

AWS provides four tools to help you securing your EC2.

- **Security Groups**
  - plays the role of a firewall
  - By default deny all incoming traffic
  - User define group behavior by setting policy rules that will either block or allow specified traffic types
  - Traffic is assessed by examining its source and destination, the network port it's targeting, and the protocol it's set to use
  - Eg, a packet that's using TCP on the SSH port 22 could, for instance, only be allowed access to a particular instance if its source IP address matches the local public IP used by computers in your office. This lets you open up SSH access on your instance without having to worry about anyone from outside your company getting in.

Amazon Elastic Compute Cloud and Amazon Elastic Block Store

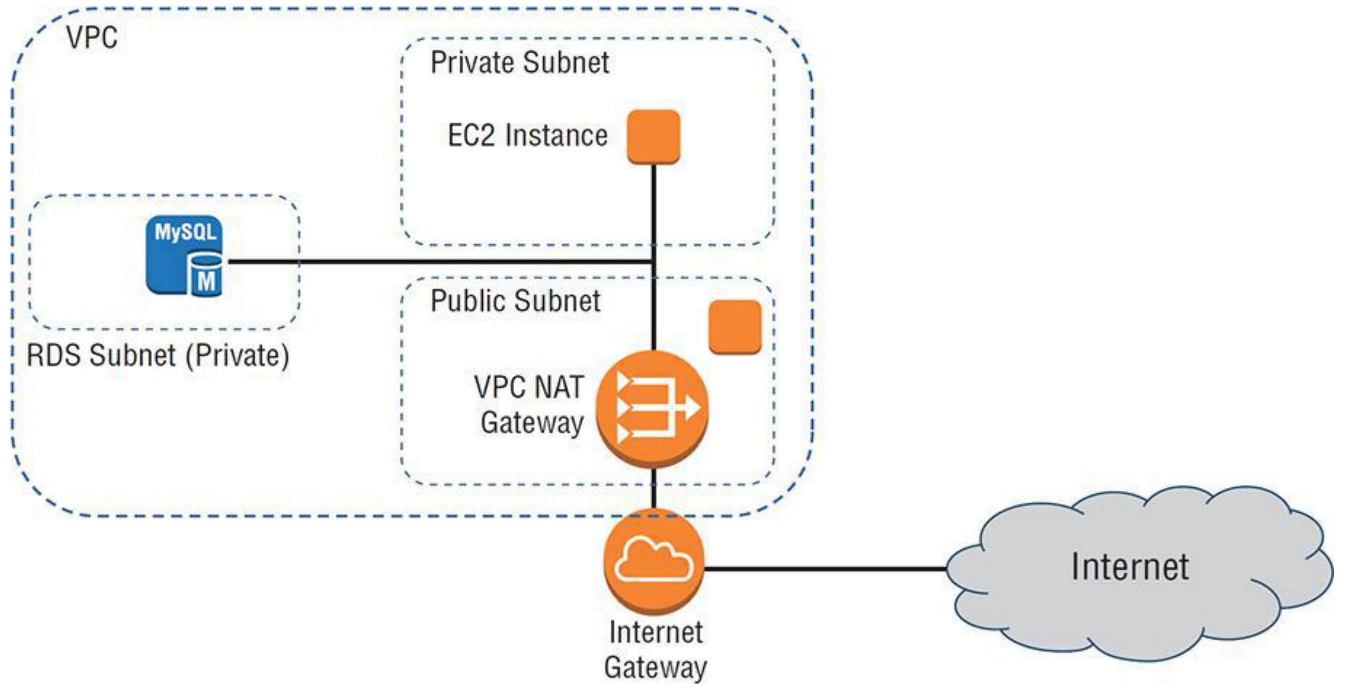
Securing Your EC2 Instance

- **IAM Roles**
  - define an IAM role by giving it permissions to perform actions on specified services or resources within your AWS account.
  - When a particular role is assigned to a user or resource, they'll gain access to whichever resources were included in the role policies
  - you can give a limited number of entities (other resources or users) exclusive access to resources like your EC2 instances
  - you can also assign an IAM role *to* an EC2 instance so that processes running within it can access the external tools—like an RDS database instance—it needs to do its work.
- **NAT Devices**
- **Key Pairs**

Amazon Elastic Compute Cloud and Amazon Elastic Block Store

Securing Your EC2 Instance

- **NAT Devices**
  - To give instance in the private subnet, a limited Internet access so it can receive security patches and software updates.
  - AWS gives you two ways to do that: a NAT instance and a NAT gateway(manged service)
- **Key Pairs**



Amazon Elastic Compute Cloud and Amazon Elastic Block Store

Securing Your EC2 Instance

- **Key Pairs** (remote login to EC2)
  - To ensure properly secured sessions (remote login to EC2), you'll need to generate a key pair, save the public key to your EC2 server, and save its private half to your local machine.
  - Windows AMI,
    - you'll use the private key file to retrieve the password you'll need to authenticate into your instance.
  - Linux AMI,
    - the private key will allow you to open an SSH session.
  - Each key pair that AWS generates for you will remain installed within its original region and available for use with newly launched instances until you delete it.
  - You *should* delete the AWS copy in the event your public key is lost or exposed.
  - Just be careful before you mess with your keys: your access to an instance might depend on it.

Amazon Elastic Compute Cloud and Amazon Elastic Block Store

Other EC2-Related Services

AWS Systems Manager

is a collection of tools for monitoring and managing the resources you have running in the AWS cloud *and* in your own on-premises infrastructure

Through the Systems Manager portal, you can organize your AWS resources into resource groups, mine various visualization tools for insights into the health and behavior of your operations, directly execute commands or launch tasks remotely without having to log on, automate patching and other lifecycle events, and manage service parameters and access secrets.

Other EC2-Related Services

Placement Groups

Placement groups are useful for multiple EC2 instances that require especially low-latency network interconnectivity.

There are two placement group strategies

- *Cluster* groups launch each associated instance within a single availability zone within close physical proximity to each other
- *Spread* groups separate instances physically across hardware to reduce the risk of failure-related data or service loss

Amazon Elastic Compute Cloud and Amazon Elastic Block Store

Other EC2-Related Services

AWS Elastic Beanstalk

Lets you upload your application code and define a few parameters, and AWS will configure, launch, and maintain all the infrastructure necessary to keep it running.

This include

- EC2 load-balanced and auto-scaled instances,
- RDS database instances,
- All the network plumbing you would otherwise have had to build yourself
- Compatible languages and platforms include .NET, Java, Node.js, Python, and Docker.
- Elastic Beanstalk adds no charges beyond the cost of the running infrastructure itself

Amazon Elastic Compute Cloud and Amazon Elastic Block Store

Other EC2-Related Services

Amazon Elastic Container Service and AWS Fargate

ECS

AWS lets you launch a prebuilt Docker host instance and define the way you want your Docker containers to behave (called a *task*), and ECS will make it all happen.

The containers exist within an infrastructure that's automated and fully integrated with your AWS resources.

AWS Fargate

Fargate tool further abstracts the ECS configuration process, removing the need for you to run and configure instances for your containers.

With Fargate, all you do is package your application and set your environment requirements.

Amazon Elastic Compute Cloud and Amazon Elastic Block Store

Other EC2-Related Services

AWS Lambda

"Serverless" applications are powered by programming code that's run on servers—just not servers under the control of the application owners. Instead, code can be configured to run when AWS's Lambda servers are triggered by preset events.

Lambda allows you to instantly perform almost any operation on demand at almost any time but without having to provision and pay for always-on servers.

VM Import/Export

VM Import/Export allows you to easily move virtual machine images back and forth between your on-premises

VMware environment and your AWS account (via an S3 bucket).

This can make it much simpler to manage hybrid environments and to efficiently migrate workloads up to the AWS cloud.

Amazon Elastic Compute Cloud and Amazon Elastic Block Store

Other EC2-Related Services

Elastic Load Balancing and Auto Scaling

A load balancer directs external user requests between multiple EC2 instances to more efficiently distribute server resources and improve user experience. Autoscaling will react to preset performance thresholds by automatically increasing or decreasing the number of EC2 instances you have running to match demand.

Both Elastic Load Balancing (ELB) and Auto Scaling can be closely integrated with your EC2 infrastructure, making for simple and seamless operation.

Amazon Elastic Compute Cloud and Amazon Elastic Block Store

Summary

The base software stack that runs on an EC2 instance is defined by your choice of Amazon Machine Image and any scripts or user data you add at launch time, and the hardware profile is the product of an instance type. A tenancy setting determines whether your instance will share a physical host with other instances.

As with all your AWS resources, it's important to give your EC2 instances easily identifiable tags that conform to a system-wide naming convention. There are limits to the number of resources you'll be allowed to launch within a single region and account-wide. Should you hit your limit, you can always request access to additional resources.

If you plan to run an instance for a year or longer, you can save a significant amount of money over paying for on-demand by purchasing a reserve instance. If your workload can withstand unexpected shutdowns, then a spot instance could also make sense.

Amazon Elastic Compute Cloud and Amazon Elastic Block Store

Summary

There are four kinds of Elastic Block Store volumes: two high IOPS and low-latency SSD types and two traditional hard drives. Your workload and budget will inform your choice. In addition, some EC2 instance types come with ephemeral instance store volumes that offer fast data access but whose data is lost when the instance is shut down.

All EC2 instances are given at least one private IP address, and should they require Internet access, they can also be given a nonpermanent public IP. Should you require a permanent public IP, you can assign an Elastic IP to the instance

You secure access to your EC2 instances using software firewalls known as security groups and can open up secure and limited access through IAM roles, NAT instances or NAT gateways, and key pairs

Amazon Elastic Compute Cloud and Amazon Elastic Block Store

Exam Essentials

Understand how to provision and launch an EC2 instance. You'll need to select the right AMI and instance type, configure a security group, add any extra storage volumes that might be needed, point to any necessary user data and scripts, and, ideally, tag all the elements using descriptive key values.

Understand how to choose the right hardware/software profile for your workload. Consider the benefits of building your own image against the ease and simplicity of using a marketplace, community, or official AMI. Calculate the user demand you expect your application to generate so you can select an appropriate instance type. Remember that you can always change your instance type later if necessary.

Understand EC2 pricing models and how to choose one to fit your needs. Know how to calculate whether you'll be best off on the spot market, with on-demand, or with reserve—or some combination of the three.

Amazon Elastic Compute Cloud and Amazon Elastic Block Store

Exam Essentials

Understand how to configure a security group to balance access with security to match your deployment profile. Security groups act as firewalls, applying policy rules to determine which network traffic is allowed through. You can control traffic based on a packet's protocol and network port and its source and intended destination.

Know how to access a running instance. Instance data, including private and public IP addresses, can be retrieved from the AWS Console, through the AWS CLI, and from metadata queries on the instance itself. You'll need this information so you can log in to administrate the instance or access its web-facing applications.

.

Amazon Elastic Compute Cloud and Amazon Elastic Block Store

Exam Essentials

Understand the features and behavior of storage volume types. SSD volumes can achieve higher IOPS and, therefore, lower latency but come at a cost that's higher than traditional hard drives.

Know how to create a snapshot from a storage volume and how to attach the snapshot to a different instance. Any EBS drive can be copied and either attached to a different instance or used to generate an image that, in turn, can be made into an AMI and shared or used to launch any number of new instances.

Storage : S3 and Glacier

- S3 is where individuals, applications, and a long list of AWS services keep their data. It's an excellent platform for the following:
  - Maintaining backup archives, log files, and disaster recovery images
  - Running analytics on big data at rest
  - Hosting static websites
  - Can be integrated with operations running within or external to AWS
- S3 provides a space for effectively unlimited *object storage*.

Storage : S3 and Glacier

Block Storage	Object Storage
Data on a raw physical storage device is divided into individual blocks whose use is managed by a file system	provides what you can think of as a flat surface on which to store your data.
NTFS is a common file system used by Windows, while Linux might use Btrfs or ext4	When you write files to S3, they're stored along with up to 2 KB of metadata. The metadata is made up of keys that establish system details like data permissions and the appearance of a file system location within nested buckets.
The file system, on behalf of the installed OS, is responsible for allocating space for the files and data that are saved to the underlying device and for providing access whenever the OS needs to read some data.	This simple design avoids some of the OS-related complications of block storage and allows anyone easy access to any amount of professionally designed and maintained storage capacity.

Simple Storage Service (S3)

S3 Service Architecture

- organize your S3 files into buckets
- 100 buckets for each of your AWS accounts (soft limit)
- S3 bucket and its contents exist within only a single AWS region
- the name of bucket must be globally unique within the entire S3 system
- URL used to access a file called *file name* that's in a bucket called *bucket name* : https://s3.amazonaws.com/bucketname/filename
- S3 stores objects within a bucket on a flat surface without subfolder hierarchies.
- Use prefixes and delimiters to give your buckets the *appearance* of a more structured organization.
- A prefix is a common text string that indicates an organization level. For example the *certification* when followed by the delimeter / would tell S3 to teat a file with a name like certification/aws.pdf as an object that should be grouped together with a second file named certification/azure.pdf

Simple Storage Service (S3)

S3 Service Architecture

- Unlimited amount of data can be stored within a bucked
- a single object may be no larger than 5 TB
- Use Multipart Upload for any object larger than 100 MB.
- Multipart Upload breaks a large object into multiple smaller parts and transmits them individually to their S3 target. If one transmission should fail, it can be repeated without impacting the others.
- Multipart Upload will be used automatically while using AWS CLI or a high-level API, but for low-level API it is manually initiated

Simple Storage Service (S3)

Encryption

- use encryption keys to protect your data while it's at rest within S3
- Use Amazon's encrypted API endpoints for data transfers—protect data during its journeys between S3 and other locations.

Server-Side Encryption

- AWS encrypts data objects as they're saved to disk and decrypt them when you send properly authenticated requests for retrieval.
- Server Side Encryption Options
  - **Amazon S3-Managed Keys (SSE-S3) :** AWS uses its own enterprise-standard keys to manage every step of the encryption and decryption process
  - **AWS KMS-Managed Keys (SSE-KMS) :** beyond the SSE-S3 features, the use of an *envelope key* is added along with a full audit trail for tracking key usage. You can optionally import your own keys through the AWS KMS service.
  - **Customer-Provided Keys (SSE-C) :** lets you provide your own keys for S3 to apply to its encryption

Simple Storage Service (S3)

Encryption

- **Client-Side Encryption**
- Encrypt data before it's transferred to S3 using AWS KMS–Managed Customer Master Key (CMK) which produces a unique key for each object before it's uploaded.
- You can also use a Client-Side Master Key, which you provide through the Amazon S3 encryption client.

Simple Storage Service (S3)

Logging

- **Client-Side Encryption**
- While enabling logging, specify both a source bucket (the bucket whose activity you're tracking) and a target bucket (the bucket to which you'd like the logs saved).
- Optionally, you can also specify delimiters and prefixes to make it easier to identify and organize logs from multiple source buckets that are saved to a single target bucket
- S3 log contains following details
  - The account and IP address of the requestor
  - The source bucket name
  - The action that was requested (get, put, etc)
  - The time the request was issued
  - The response status (including error code)

S3 buckets are also used by other AWS services—including CloudWatch and CloudTrail—to store their logs or other objects (like EBS Snapshots).

Durability and Availability

Durability : data survival in the AWS cloud

- S3 measures durability as a percentage. The 99.999999999 percent durability guarantee for most S3 classes and Amazon Glacier corresponds to an average annual expected loss of 0.000000001% of objects
- For example, if you store 10,000,000 objects with Amazon S3, you can on average expect to incur a loss of a single object once every 10,000 years

Guaranteed reliability standards for S3 storage class

	S3 Standard	**S3 Standard-IA**	**S3 One Zone-IA**	**Reduced Redundancy**
Durability guarantee	100.00%	100.00%	100.00%	99.99%
Concurrent facility fault tolerance	2	2	1	1

Durability and Availability

Availability : how quickly you can retrieve data

- Object availability is also measured as a percentage, it's the percentage you can expect a given object to be instantly available on request through the course of a full year
- 99.99% means there will be less than nine hours each year of down time.

Guaranteed availability standards for S3 storage

	S3 Standard	**S3 Standard-IA**	**S3 One Zone-IA**	**Reduced Redundancy**
Availability guarantee	99.99%	99.90%	99.50%	99.99%

Simple Storage Service (S3)

Eventually Consistent Data

– Uploading a new version of a file or, alternatively, deleting an old file altogether can result in one site reflecting the new state with another still unaware of any changes.

– To ensure that there's never a conflict between versions of a single object——you should treat your data according to an Eventually Consistent standard.

– expect a delay (usually just two seconds or less) and design your operations accordingly.

– S3 provides read-after-write consistency for the creation (put) of *new* objects

Simple Storage Service (S3)

S3 Object Lifecycle

- S3 lets you automate all this with its versioning and lifecycle features

Versioning

- If you enable versioning at the bucket level, then older overwritten copies of an object will be saved and remain accessible indefinitely.
- This solves the problem of accidentally losing old data, but it replaces it with the potential for archive bloat.

Lifecycle Management

- Configure lifecycle rules for a bucket that will automatically transition an object's storage class after a set number of days.
- You can have new objects remain in the S3 Standard class for their first 30 days after which they're moved to the cheaper One Zone IA for another 30 days.
- If regulatory compliance requires that you maintain older versions, your files could then be moved to the low-cost, long-term storage service Glacier for 365 more days before being permanently deleted.

Simple Storage Service (S3)

Accessing S3 Objects

Access Control

- new S3 buckets and objects will be fully accessible to your account but to no other AWS accounts or external visitors.
- You can strategically open up access at the bucket and object levels using
  - Access control list (ACL) rules,
  - **Finer-grained S3 bucket policies** : are formatted as JSON text and attached to your S3 bucket
    - » Use case: where you want to control access to a single S3 bucket for multiple external accounts and users.
  - Identity and Access Management (IAM) policies.
    - » IAM policies exist at the account level within IAM
    - » Use case: when you're trying to control the way individual users and roles access multiple resources, including S3.

– Amazon recommends applying S3 bucket policies or IAM policies instead of ACLs.

– **Presigned URLs**

Simple Storage Service (S3)

The following code is an example of an S3 bucket policy that allows both the root user and the user Steve from the specified AWS account to access the S3 MyBucket bucket and its contents. Both users are considered principals within this rule.

```
{ "Version": "2012-10-17", "Statement": [ {
"Effect": "Allow", "Principal": { "AWS":
["arn:aws:iam::xxxxxxxxxxxx:root",
"arn:aws:iam::xxxxxxxxxxxx:user/Steve"] },
"Action": "s3:*", "Resource":
["arn:aws:s3:::MyBucket",
"arn:aws:s3:::MyBucket/*"] } ] }
```

Simple Storage Service (S3)

Accessing S3 Objects

- **Presigned URLs**
  - To provide temporary access to an object that's otherwise private
  - you can generate a presigned URL
  - The URL will be usable for a specified period of time, after which it will become invalid
  - You can build presigned URL generation into your code to provide object access programmatically.

The following AWS CLI command will return a URL that includes the required authentication string.

```
aws s3 presign s3://MyBucketName/PrivateObject --expires-in 600
```

The authentication will become invalid after 10 minutes (600 seconds).

The default expiration value is 3,600 seconds (one hour).

Simple Storage Service (S3)

Static Website Hosting

Steps to configure an Amazon S3 bucket for static website hosting:

- Create a bucket with the same name as the desired website hostname.
- Upload the static files to the bucket.
- Make all the files public (world readable).
- Enable static website hosting for the bucket. This includes specifying an Index document and an Error document.
- The website will now be available at the S3 website URL:
- <bucket-name>.s3-website-<AWS-region>.amazonaws.com.
- Create a friendly DNS name in your own domain for the website using a DNS CNAME, or an Amazon Route 53 alias that resolves to the Amazon S3 website URL.
- The website will now be available at your website domain name.

S3 and Glacier

S3 and Glacier Select

- Apply SQL-like queries to stored objects so that only relevant data from within objects is retrieved, permitting significantly more efficient and cost-effective operations.
- Use case :
  - Large CSV files containing sales and inventory data from multiple retail sites.
  - Your company's marketing team might need to periodically analyze only sales data and only from certain stores.
  - Using S3 Select, they'll be able to retrieve exactly the data they need—just a fraction of the full data set—while bypassing the bandwidth and cost overhead associated with downloading the whole thing.

S3 and Glacier

Amazon Glacier

- Glacier guarantees 99.999999999 percent durability
- Can be incorporated into S3 lifecycle configurations.
- the term *archive* is used to describe an object like a document, video, or a TAR or ZIP file.
- Archives are stored in *vaults*—the Glacier equivalent of S3's buckets.
- Glacier vault names do not have to be globally unique.

Glacier	S3
supports archives as large as 40 TB	5 TB limit in S3
archives are encrypted by default	encryption on S3 is an option you need to select
Glacier archives are given machine-generated IDs	S3's "human-readable" key
Getting the objects in an existing Glacier archive can take a number of hours. Expedited will incur a premium charge.	nearly instant access from S3
provide inexpensive long-term storage for data that will be needed only in unusual and infrequent circumstances.	

S3 and Glacier

Other Storage-Related Services

Amazon Elastic File System

- EFS provides automatically scalable and shareable file storage. EFS-based files are designed to be accessed from within a virtual private cloud (VPC) via Network File System (NFS) mounts on EC2 instances or from your on-premises servers through AWS Direct Connect connections.

AWS Storage Gateway

- AWS Storage Gateway provides software gateway appliances (based on VMware ESXi, Microsoft Hyper-V, or EC2 images) with multiple virtual connectivity interfaces. Local devices can connect to the appliance as though it's a physical backup device like a tape drive, while the data itself is saved to AWS platforms like S3 and EBS.

AWS Snowball

- To move terabyte or even petabyte-scaled data for backup or active use within AWS, ordering a Snowball device might be the best option
- When requested, AWS will ship you a physical, 256-bit, encrypted Snowball storage device onto which you'll copy your data.
- You then ship the device back to Amazon where its data will be uploaded to your S3 bucket(s).

S3 and Glacier

Summary

- Amazon S3 provides reliable and highly available object-level storage for low-maintenance, high-volume archive and data storage. Objects are stored in buckets on a "flat" surface but, through the use of prefixes, can be made to appear as though they're part of a normal file system
- Always encrypt S3 data using either AWS-provided or self-serve encryption keys. Encryption can take place when data is at rest using either server-side or client-side encryption
- There are multiple storage classes within S3 relying on varying degrees of data replication that allows to balance durability, availability, and cost. Lifecycle management automate the transition of your data between classes until it's no longer needed and can be deleted.
- Control who and what get access to your S3 buckets—and when—through legacy ACLs or through more powerful S3 bucket policies and IAM policies. Presigned URLs are also a safe way to allow temporary and limited access to your data
- Reduce the size and cost of your requests against S3 and Glacier-based data by leveraging the SQL-like Select feature.
- Created inexpensive and simple static websites through S3 buckets.
- Amazon Glacier stores data archives in vaults that might require hours to retrieve but that cost considerably less than the S3 storage classes.

S3 and Glacier

Exam Essentials

Understand the way S3 resources are organized

- S3 objects are stored in buckets whose names must be globally unique. Buckets are associated with AWS regions. Objects are stored within buckets on a "flat" surface, but prefixes and delimiters can give data the appearance of a folder hierarchy.

Understand how to optimize your data transfers

- While the individual objects you store within an S3 bucket can be as large as 5 TB, anything larger than 100 MB *should* be uploaded with Multipart Upload, and objects larger than 5 GB *must* use Multipart Upload

Understand how to secure your S3 data

- You can use server-side encryption to protect data within S3 buckets using either AWS-generated or your own privately generated keys. Data can be encrypted even before being transferred to S3 using client-side encryption.

Understand how S3 object durability and availability are measured

- The various S3 classes (and Glacier) promise varying levels of infrastructure reliability, along with data availability.

S3 and Glacier

Exam Essentials

Understand S3 object versioning and lifecycle management

- Older versions of S3 objects can be saved even after they've been overwritten. To manage older objects, you can automate transition of objects between more accessible storage classes to less expensive but less accessible classes. You can also schedule object deletion.

Understand how to secure your S3 objects

- You can control access through the legacy, bucket, and object-based ACL rules or by creating more flexible S3 bucket policies or, at the account level, IAM policies. You can also provide temporary access to an object using a presigned URL.

Understand how to create a static website

- S3-based HTML and media files can be exposed as a website that, with the help of Route 53 and CloudFront, can even live behind a DNS domain name and use encrypted HTTPS pages.

Understand the differences between S3 and Glacier

- Glacier is meant for inexpensive long-term storage for data archives that you're unlikely to need often.

S3 and Glacier

Exam Essentials

- S3 can be used to share files, but it doesn't offer low-latency access—and its eventual consistency won't work well with filesystems. Storage Gateway is designed to simplify backing up archives to the AWS cloud; it's not for sharing files. EBS volumes can be used for only a single instance at a time.
- Object metadata contains information used by S3 to manage an object's security profile, behavior, and the way it's exposed to client requests. Storing this information requires very little space—2 KB will normally be more than enough
- In theory, at least, there's no limit to the data you can upload to a single bucket or to all the buckets in your account or to the number of times you upload (using the PUT command) By default, however, you are allowed only 100 S3 buckets per account.

VPC

Amazon Virtual Private Cloud (Amazon VPC)

Introduction

– It provides the networking layer of EC2

– It is virtual network that can contain EC2 instances as well as network resources for other AWS services.

– By default, every VPC is isolated from all other networks.

– You can, however, connect your VPC to other networks, including the Internet and other VPCs.

– You can have multiple VPCs in your account and create multiple VPCs in a single region.

– When you create a VPC in a region, it won't show up in any other regions.

Amazon Virtual Private Cloud (Amazon VPC)

VPC CIDR Blocks

- a VPC consists of at least one range of contiguous IP addresses. This address range is represented as a *Classless interdomain routing* (CIDR) block.
- The CIDR block determines which IP addresses may be assigned to instances and other resources within the VPC. You must assign a primary CIDR block when creating a VPC.
- CIDR notation, (*slash notation)*. For example, the CIDR 172.16.0.0/16 includes all addresses from 172.16.0.0 to 172.16.255.255—a total of 65,536 addresses
- prefix length of a VPC CIDR can range from /16 to /28 The smaller the prefix length, the greater the number of IP addresses in the CIDR. A /28 prefix length gives you only 16 addresses.
- it's best to use one in the RFC 1918 range to avoid conflicts with public Internet addresses.
- 10.0.0.0–10.255.255.255 (10.0.0.0/8)
- 172.16.0.0–172.31.255.255 (172.16.0.0/12)
- 192.168.0.0–192.168.255.255 (192.168.0.0/16)
- Make sure the VPC CIDR you choose doesn't overlap with addresses already in use on the other network
- Once created the primary CIDR block can’t be changed, so think carefully about your address requirements before creating a VPC.

Amazon Virtual Private Cloud (Amazon VPC)

Secondary CIDR Blocks

– You may optionally specify secondary CIDR blocks for a VPC after you've created it. These blocks must come from either the same address range as the primary or a publicly routable range, but they must not overlap with the primary or other secondary blocks

– For example, if the VPC's primary CIDR is 172.16.0.0/16, you may specify a secondary CIDR of 172.17.0.0/16. But you may not specify 192.168.0.0/16.

– If you think you might ever need a secondary CIDR, be careful about your choice of primary CIDR. If you choose 192.168.0.0/16 as your primary CIDR, you won't be able to create a secondary CIDR using any of the RFC 1918 ranges.

IPv6 CIDR Blocks

– You may let AWS assign an IPv6 CIDR to your VPC. Unlike the primary CIDR, which is an IP prefix of your choice, you can't choose your own IPv6 CIDR.

– Instead, AWS assigns one to your VPC at your request. The IPv6 CIDR will be a publicly routable prefix from the global unicast IPv6 address space. For example, AWS may assign you the CIDR 2600:1f18:2551:8900/56. Note that the prefix length of an IPv6 VPC CIDR is always /56.

Amazon Virtual Private Cloud (Amazon VPC)

Subnets

- A subnet is a logical container within a VPC that holds your EC2 instances.
- In concept, subnets are similar to virtual LANs (VLANs) in a traditional network.
- A subnet lets you isolate instances from each other, control how traffic flows to and from your instances, and lets you organize them by function.
- Create one subnet for public web servers that need to be accessible from the Internet and create another subnet for database servers that only the web instances can access.

Subnet CIDR Blocks

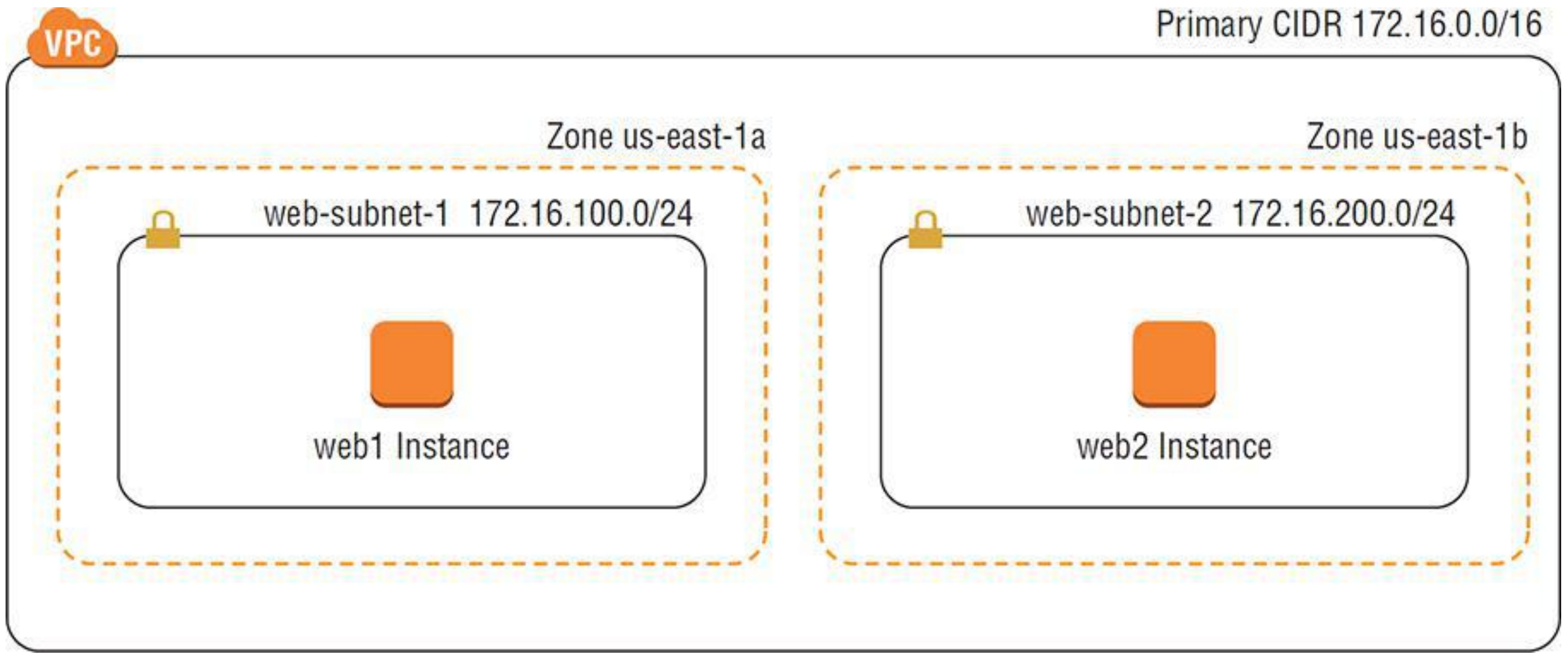
- Each subnet has its own CIDR block that must be a subset of the VPC CIDR that it resides in. For example, if your VPC has a CIDR of 172.16.0.0/16, one of your subnets may have a CIDR of 172.16.100.0/24. This range covers 172.16.100.0–172.16.100.255, which yields a total of 256 addresses
- AWS reserves the first four and last IP addresses in every subnet. You can't assign these addresses to any instances.
- The restrictions on prefix lengths for a subnet CIDR are the same as VPC CIDRs.
- Subnet CIDR blocks in a single VPC can't overlap with each other.
- Once you assign a CIDR to a subnet, you can't change it

Amazon Virtual Private Cloud (Amazon VPC)

Availability Zones

– A subnet can exist within only one *availability zone* (AZ, or *zone* for short), which is roughly analogous to a relatively small geographic location such as a data center. Although availability zones in an AWS region are connected, they are designed so that a failure in one zone doesn't cause a failure in another.

– You can achieve resiliency for your applications by creating two subnets in different availability zones and then spreading your instances across those zones..


VPC
Primary CIDR 172.16.0.0/16
Zone us-east-1a
Zone us-east-1b
web-subnet-1 172.16.100.0/24
web-subnet-2 172.16.200.0/24
web1 Instance
web2 Instance

Amazon Virtual Private Cloud (Amazon VPC)

Elastic Network Interfaces

- An elastic network interface (ENI) allows an instance to communicate with other network resources including AWS services, other instances, on-premises servers, and the Internet.
- It also makes it possible for you to connect to the operating system running on your instance to manage it.
- As the name suggests, an ENI performs the same basic function as a network interface on a physical server, although ENIs have more restrictions on how you can configure them
- Every instance must have a *primary network interface* (also known as the *primary ENI*), which is connected to only one subnet. This is the reason you have to specify a subnet when launching an instance.
- You can't remove the primary ENI from an instance.

Amazon Virtual Private Cloud (Amazon VPC)

Primary and Secondary Private IP Addresses

- Each instance must have a *primary private IP address* from the range specified by the subnet CIDR.
- The primary private IP address is bound to the primary ENI of the instance. You can't change or remove this address, but you can assign secondary private IP addresses to the primary ENI.
- Any secondary addresses must come from the same subnet that the ENI is attached to.
- It's possible to attach additional ENIs to an instance. Those ENIs may be in a different subnet, but they must be in the same availability zone as the instance.
- As always, any addresses associated with the ENI must come from the subnet to which it is attached.

Amazon Virtual Private Cloud (Amazon VPC)

Attaching Elastic Network Interfaces

– An ENI can exist independently of an instance.

– You can create an ENI first and then attach it to an instance later. For example, you can create an ENI in one subnet and then attach it to an instance as the primary ENI when you launch the instance.

– If you disable the Delete on Termination attribute of the ENI, you can terminate the instance without deleting the ENI. You can then associate the ENI with another instance

– You can also take an existing ENI and attach it to an existing instance as a secondary ENI.

– This lets you redirect traffic from a failed instance to a working instance by detaching the ENI from the failed instance and reattaching it to the working instance.

Amazon Virtual Private Cloud (Amazon VPC)

Internet Gateways

- An *Internet gateway* gives instances the ability to receive a public IP address, connect to the Internet, and receive requests from the Internet.
- When you create a VPC, it does not have an Internet gateway associated with it. You must create an Internet gateway and associate it with a VPC manually.
- You can associate only one Internet gateway with a VPC.
- An Internet gateway is somewhat analogous to an Internet router an Internet service provider may install on-premises. But in AWS, an Internet gateway doesn't behave exactly like a router.
- In a traditional network, you might configure your core router with a default gateway IP address pointing to the Internet router to give your servers access to the Internet.
- An Internet gateway, however, doesn't have a management IP address or network interface. Instead, AWS identifies an Internet gateway by its resource ID, which begins with igw-followed by an alphanumeric string.
- To use an Internet gateway, you must create a *default route* in a *route table* that points to the Internet gateway as a target.

Amazon Virtual Private Cloud (Amazon VPC)

Route Tables

- In AWS, you only have to manage the route table which the implied router uses
- Each route table consists of one or more routes and at least one subnet association.
- Think of a route table as being connected to multiple subnets in much the same way a traditional router would be.
- When you create a VPC, AWS automatically creates a default route table called the *main route table* and associates it with every subnet in that VPC.
- You can use the main route table or create a custom one that you can manually associate with one or more subnets.
- If you do not explicitly associate a subnet with a route table you've created, AWS will implicitly associate it with the main route table.
- A subnet cannot exist without a route table association.

Amazon Virtual Private Cloud (Amazon VPC)

Routes.

- Routes determine how to forward traffic from instances within the subnets associated with the route table. IP routing is destination-based, meaning that routing decisions are based only on the destination IP address, not the source.
- When you create a route, you must provide the destination and the target.
- The destination must be an IP prefix in CIDR notation. The target must be an AWS network resource such as an Internet gateway or an ENI. It cannot be a CIDR
- Every route table contains a *local route* that allows instances in different subnets to communicate with each other.

Destination	Target
172.31.0.0/16	Local

- The local route is the only mandatory route that exists in every route table.
- It's what allows communication between instances in the same VPC.
- Because there are no routes for any other IP prefixes, any traffic destined for an address outside of the VPC CIDR range will get dropped

Amazon Virtual Private Cloud (Amazon VPC)

The Default Route

- To enable Internet access for your instances, you must create a default route pointing to the Internet gateway. After adding a default route, you would end up with this

Destination	Target
172.31.0.0/16	Local
0.0.0.0/0	igw-0e538022a0fddc318

- The 0.0.0.0/0 prefix encompasses all IP addresses, including those of hosts on the Internet. This is why it's always listed as the destination in a default route.
- Any subnet that is associated with a route table containing a default route pointing to an Internet gateway is called a *public subnet*. Contrast this with a *private subnet* that does not have a default route

Amazon Virtual Private Cloud (Amazon VPC)

Security Groups

- A *security group* functions as a firewall that controls traffic to and from an instance by permitting traffic to ingress or egress that instance's ENI
- Every ENI must have at least one security group associated with it. One ENI can have multiple security groups attached, and the same security group can be attached to multiple ENIs.
- In practice, because most instances have only one ENI, people often think of a security group as being attached to an instance. When an instance has multiple ENIs, take care to note whether those ENIs use different security groups
- When you create a security group, you must specify a group name, description, and VPC for the group to reside in. Once you create the group, you specify inbound and outbound rules to allow traffic through the security group.

Amazon Virtual Private Cloud (Amazon VPC)

Inbound Rules

– Inbound rules specify what traffic is allowed into the attached ENI. An inbound rule consists of three required elements:

Source	Protocol	Port Range
198.51.100.10/32	TCP	22
0.0.0.0/0	TCP	443

– When you create a security group, it doesn't contain any inbound rules.

– Security groups use a default-deny approach, ie denies all traffic that is not explicitly allowed by a rule.

– If you want to allow anyone on the Internet to connect to an instance, you'd need an inbound rule to allow all TCP traffic coming in on port 443 (the default port and protocol for HTTPS).

– To manage this instance using SSH, you'd need another inbound rule for TCP port 22. However, you don't want to allow SSH access from just anyone. You need to allow SSH access only from the IP address 198.51.100.10

– The prefix 0.0.0.0/0 covers all valid IP addresses, so using the preceding rule would allow HTTPS access not only from the Internet but from all instances in the VPC as well.

Amazon Virtual Private Cloud (Amazon VPC)

Outbound Rules

- Outbound rules specify what traffic the instance may send via the attached ENI. An outbound rule mirrors an inbound rule and contains three elements.
- When you create a security group, AWS automatically creates the outbound rule listed

Destination	Protocol	Port Range
0.0.0.0/0	All	All

- The main purpose of this rule is to allow the instance to access the Internet and other AWS resources. You may delete this rule, but if you do, the security group won't permit your instance to access the Internet or anything else.

Sources and Destinations

- The source or destination in a rule can be any CIDR or the resource ID of a security group.
- If you specify a security group as the source or destination, any instance with that security group attached will be allowed by the rule.
- This makes it easy to allow instances to communicate with each other by simply assigning the same security group to all of them.

Amazon Virtual Private Cloud (Amazon VPC)

Default Security Group

– Each VPC contains a default security group that you can't delete.
– You don't have to use it, but if you do, you can modify its rules to suit your needs.
– Alternatively, you may opt to create your own custom group and use it instead.

Amazon Virtual Private Cloud (Amazon VPC)

Network Access Control Lists

- a *network access control list* (NACL) functions as a firewall in that it contains inbound and outbound rules to allow traffic based on a source or destination CIDR, protocol, and port. Also, each VPC has a default NACL that can't be deleted.
- a NACL is attached to a subnet. The NACL associated with a subnet controls what traffic may enter and exit that subnet. This means that NACLs can't be used to control traffic between instances in the same subnet. If you want to do that, you have to use security groups.
- A subnet can have only one NACL associated with it. When you create a new subnet in a VPC,
- the VPC's default NACL is associated with the subnet by default.
- You can modify the default NACL, or you can create a new one and associate it with the subnet.
- You can also associate the same NACL with multiple subnets, provided those subnets are all in the same VPC as the NACL.
- Unlike a security group, which is stateful, a NACL is stateless, meaning that it doesn't track the state of connections passing through it.
- This is much like an access control list (ACL) on a traditional switch or router.
- The stateless nature of the NACL is why each one is preconfigured with rules to allow all inbound and outbound traffic

Amazon Virtual Private Cloud (Amazon VPC)

Inbound Rules

- Inbound rules determine what traffic is allowed to ingress the subnet
- The default NACL for a VPC with no IPv6 CIDR comes prepopulated with the two inbound rules

Rule Number	Protocol	Port Range	Source	Action
100	All	All	0.0.0.0/0	Allow
*	All	All	0.0.0.0/0	Deny

- NACL rules are processed in ascending order of the rule number. Rule 100 is the lowest-numbered rule, so it gets processed first. This rule allows all traffic from any source. You can delete or modify this rule or create additional rules before or after it.
- It's designated by an asterisk (*) instead of a number and is always the last rule in the list. You can't delete or otherwise change the default rule. The default rule causes the NACL to deny any traffic that isn't explicitly allowed by any of the preceding rules

Rule Number	Protocol	Port Range	Source	Action
90	TCP	80	0.0.0.0/0	Deny

- This rule denies all TCP traffic with a destination port of 80. Because it's the lowest-numbered rule in the list, it gets processed first. Any traffic not matching this rule would be processed by rule 100, which allows all traffic.

Amazon Virtual Private Cloud (Amazon VPC)

Outbound Rules

the outbound NACL rules follow an almost identical format as the inbound rules

Rule Number	Protocol	Port Range	Destination	Action
100	All	All	0.0.0.0/0	Allow
*	All	All	0.0.0.0/0	Deny

– In most cases you will need these rules whether you use the default NACL or a custom one. Because a NACL is stateless, it won't automatically allow return traffic. Therefore, if you permit HTTPS traffic with an inbound rule, you must also explicitly permit the return traffic using an outbound rule. In this case, rule 100 permits the return traffic

– If you do need to restrict access from the subnet—to block Internet access, for example—you will need to create an outbound rule to allow return traffic over *ephemeral ports*. Ephemeral ports are reserved TCP or UDP ports that clients listen for reply traffic on. As an example, when a client sends an HTTPS request to your instance over TCP port 80, that client may listen for a reply on TCP port 36034. Your NACL's outbound rules must allow traffic to egress the subnet on TCP port 36034.

– The range of ephemeral ports varies by client operating system. Many modern operating systems use ephemeral ports in the range of 49152–65535, but don't assume that allowing only this range will be sufficient. The range for TCP ports may differ from the range for UDP, and older or customized operating systems may use a different range altogether. To maintain compatibility, do not restrict outbound traffic using a NACL. Use a security group instead.

Amazon Virtual Private Cloud (Amazon VPC)

Using Network Access Control Lists and Security Groups Together

– You may want to use a NACL in addition to a security group so that you aren't dependent upon users to specify the correct security group when they launch an instance. Because a NACL is applied to the subnet, the rules of the NACL apply to all traffic ingressing and egressing the subnet, regardless of how the security groups are configured.

– When you make a change to a NACL or security group rule, that change takes effect immediately (practically, within several seconds). Avoid changing security groups and NACLs simultaneously. If your changes don't work as expected, it can become difficult to identify whether the problem is with a security group or NACL. Complete your changes on one before moving to the other. Additionally, be cautious about making changes when there are active connections to an instance, as an incorrectly ordered NACL rule or missing security group rule can terminate those connections. Refer to Exercise 4.5 for a demonstration.

– Note that in a NACL rule you can specify a CIDR only as the source or destination. This is unlike a security group rule wherein you can specify another security group for the source or destination.

Amazon Virtual Private Cloud (Amazon VPC)

Public IP Addresses

- A public IP address is reachable over the public Internet. This is in contrast to RFC 1918 addresses, which cannot be routed over the Internet but can be routed within private networks.
- You need a public IP address for an instance if you want others to directly connect to it via the Internet. You *may* also give an instance a public IP address if you want it to have outbound-only Internet access. You *don't* need a public IP address for your instances to communicate with each other within the VPC infrastructure, as this instance-to-instance communication happens using private IP addresses.
- When you launch an instance into a subnet, you can choose to automatically assign it a public IP. This is convenient, but there are a couple of potential downsides to this approach.
- First, if you forget to choose this option when you launch the instance, you cannot go back and have AWS automatically assign a public IP later. Also, when you stop or terminate the instance, you will lose the public IP address. If you stop and restart the instance, it will receive a different public IP.
- Even if you don't plan to stop your instance, keep in mind that AWS may perform maintenance events that cause your instance to restart. If this happens, its public IP address will change.
- If your instance doesn't need to maintain the same IP address for a long period of time, this approach may be acceptable. If not, you may opt instead for an *elastic IP address*.

Amazon Virtual Private Cloud (Amazon VPC)

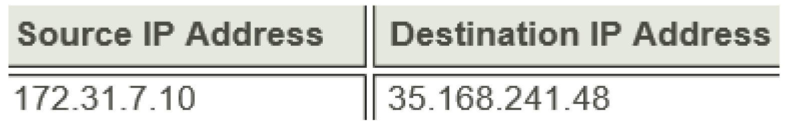
Elastic IP Addresses

- An elastic IP address (EIP) is a type of public IP address that AWS allocates to your account when you request it. Once AWS allocates an EIP to your account, you have exclusive use of that address until you manually release it. Outside of AWS, there's no noticeable difference between an EIP and an automatically assigned public IP
- When you initially allocate an EIP, it is not bound to any instance. Instead, you must associate it with an ENI. You can move an EIP around to different ENIs, although you can associate it with only one ENI at a time. Once you associate an EIP with an ENI, it will remain associated for the life of the ENI or until you disassociate it.
- If you associate an EIP to an ENI that already has an automatically assigned public IP address allocated, AWS will replace the public IP address with the EIP. Complete Exercise 4.7 to practice allocating and using an EIP.

Amazon Virtual Private Cloud (Amazon VPC)

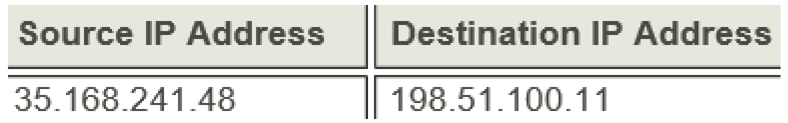
Network Address Translation

- When you associate an ENI with a public IP address, the ENI maintains its private IP address. Associating a public IP with an ENI doesn't reconfigure the ENI with a new address. Instead, the Internet gateway maps the public IP address to the ENI's private IP address using a process called *network address translation* (NAT).
- When an instance with a public IP connects to a host on the Internet, the host sees the traffic as originating from the instance's public IP. For example, assume an instance with a private IP address of 172.31.7.10 is associated with the EIP 35.168.241.48. When the instance attempts to send a packet to the Internet host 198.51.100.11, it will send the following packet to the Internet gateway:



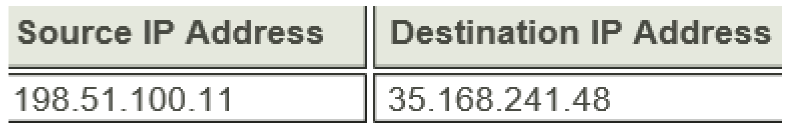
Source IP Address	Destination IP Address
172.31.7.10	35.168.241.48

- The Internet gateway will translate this packet to change the source IP address to the instance's public IP address. The translated packet, which the Internet gateway forwards to the host, looks like this:



Source IP Address	Destination IP Address
35.168.241.48	198.51.100.11

- Likewise, when a host on the Internet sends a packet to the instance's EIP, the Internet gateway will perform network address translation on the incoming packet. The packet that reaches the Internet gateway from the Internet host will look like this:

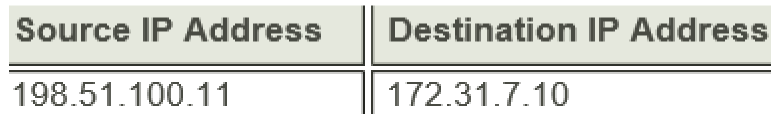


Source IP Address	Destination IP Address
198.51.100.11	35.168.241.48

Amazon Virtual Private Cloud (Amazon VPC)

Network Address Translation

– The Internet gateway will translate this packet, replacing the destination IP address with the instance's private IP address, as follows:



Source IP Address	Destination IP Address
198.51.100.11	172.31.7.10

– Network address translation occurs automatically at the Internet gateway when an instance has a public IP address. You can't change this behavior.

– Network address translation as described here is also sometimes called *one-to-one NAT* because one private IP address gets mapped to one public IP address.

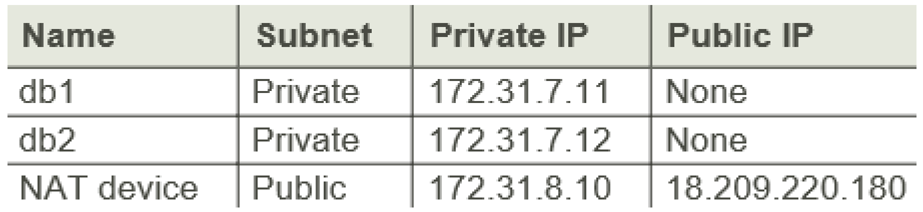
Amazon Virtual Private Cloud (Amazon VPC)

Network Address Translation Devices

NAT gateway

NAT instance

- It allow an instance to access the Internet while preventing hosts on the Internet from reaching the instance directly.
- This is useful when an instance needs to go out to the Internet to fetch updates or to upload data but does not need to service requests from clients.
- When you use a NAT device, the instance needing Internet access does not have a public IP address allocated to it. Incidentally, this makes it impossible for hosts on the Internet to reach it directly.
- Instead, only the NAT device is configured with a public IP. Additionally, the NAT device has an interface in a public subnet.

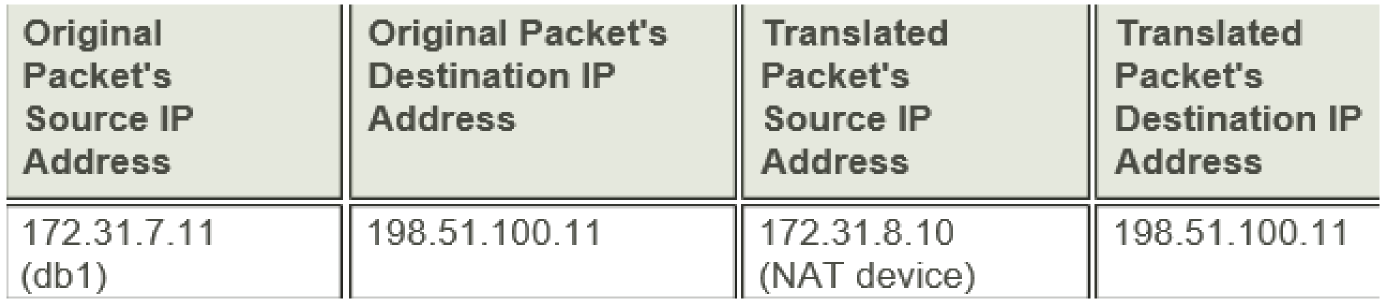


Name	Subnet	Private IP	Public IP
db1	Private	172.31.7.11	None
db2	Private	172.31.7.12	None
NAT device	Public	172.31.8.10	18.209.220.180

- When db1 sends a packet to a host on the Internet with the address 198.51.100.11, the packet must first go to the NAT device. The NAT device translates the packet as follows:

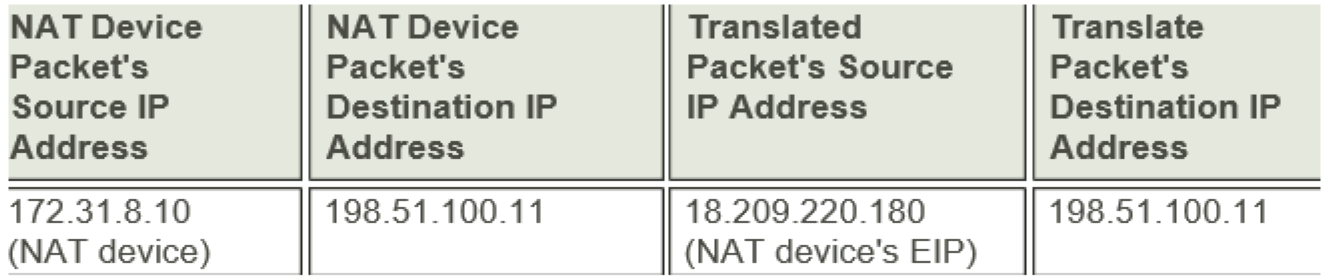
Amazon Virtual Private Cloud (Amazon VPC)

Network Address Translation



Original Packet's Source IP Address	Original Packet's Destination IP Address	Translated Packet's Source IP Address	Translated Packet's Destination IP Address
172.31.7.11 (db1)	198.51.100.11	172.31.8.10 (NAT device)	198.51.100.11

The NAT device then takes the translated packet and forwards it to the Internet gateway. The Internet gateway performs NAT translation on this packet as follows:



NAT Device Packet's Source IP Address	NAT Device Packet's Destination IP Address	Translated Packet's Source IP Address	Translate Packet's Destination IP Address
172.31.8.10 (NAT device)	198.51.100.11	18.209.220.180 (NAT device's EIP)	198.51.100.11

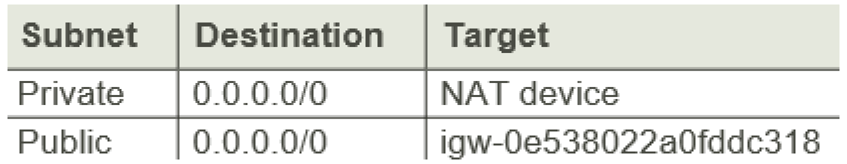
Multiple instances can use the same NAT device, thus sharing the same public IP address for outbound connections. The function that NAT devices perform is also called *port address translation* (PAT).

Amazon Virtual Private Cloud (Amazon VPC)

Configuring Route Tables to Use NAT Devices

Instances that use the NAT device must send Internet-bound traffic to it, while the NAT device must send Internet-bound traffic to an Internet gateway. Hence, the NAT device and the instances that use it must use different default routes. Furthermore, they must also use different route tables and hence must reside in separate subnets

Notice that the instances reside in the Private subnet, and the NAT device is in the Public subnet. The default routes for these subnets would follow the pattern

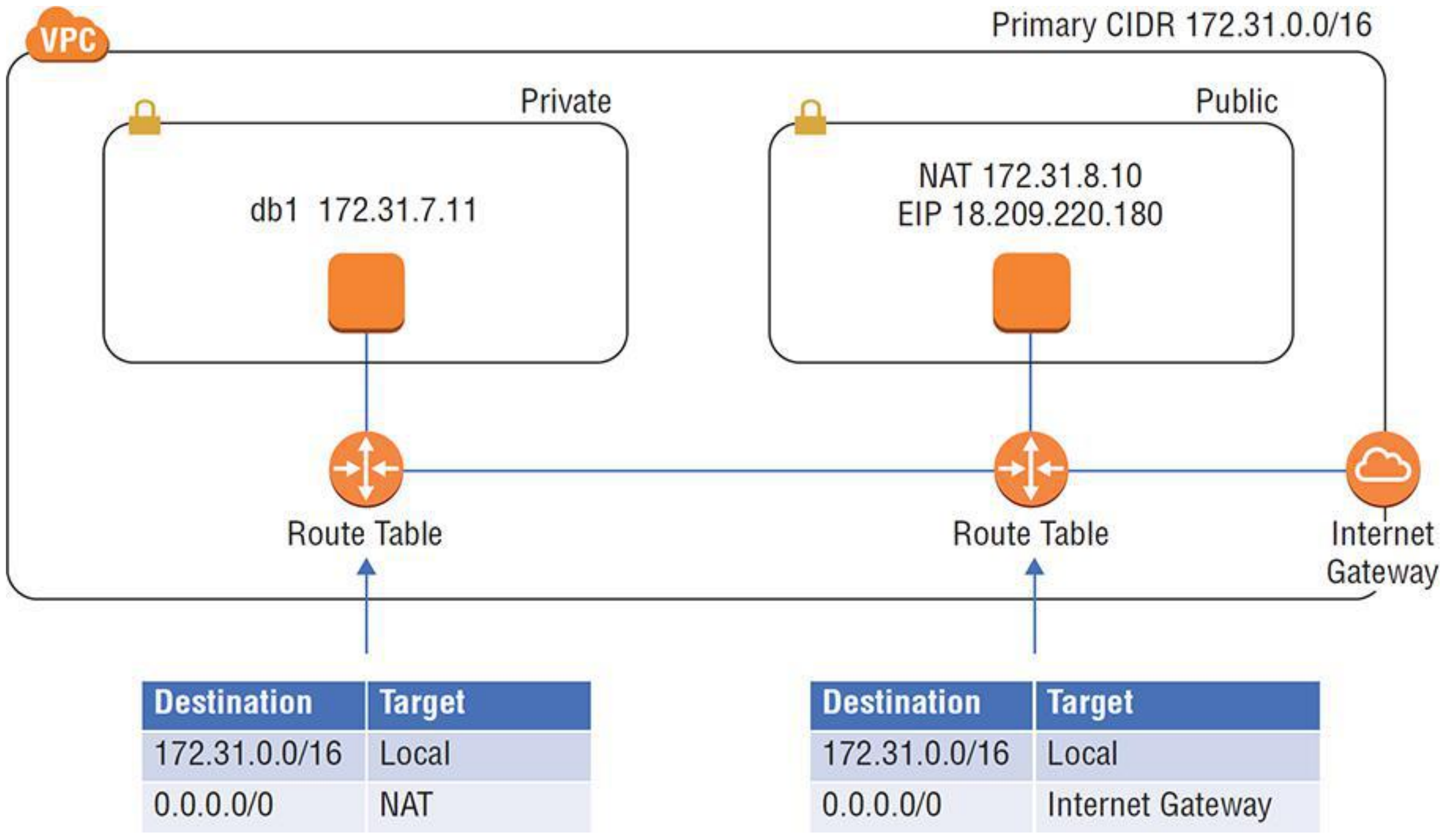


Subnet	Destination	Target
Private	0.0.0.0/0	NAT device
Public	0.0.0.0/0	igw-0e538022a0fddc318

see the relationship between both of the route tables. Recall that a route target must be a VPC resource such as instance, Internet gateway, or ENI. The specific target you choose depends on the type of NAT device you use: a NAT gateway or a NAT instance.

Amazon Virtual Private Cloud (Amazon VPC)

Configuring Route Tables to Use NAT Devices



Destination	Target
172.31.0.0/16	Local
0.0.0.0/0	NAT

Destination	Target
172.31.0.0/16	Local
0.0.0.0/0	Internet Gateway

Amazon Virtual Private Cloud (Amazon VPC)

NAT Gateway

A NAT gateway is a NAT device managed by AWS. Like an Internet gateway, it's a one-size-fits-all resource. It doesn't come in a variety of flavors, and there's nothing to manage or access. It automatically scales to accommodate your bandwidth requirements. You set it and forget it.

When you create a NAT gateway, you must assign it an EIP. A NAT gateway can reside in only one subnet, which must be a public subnet for it to access the Internet. AWS selects a private IP address from the subnet and assigns it to the NAT gateway. For redundancy, you may create additional NAT gateways in different availability zones.

After creating a NAT gateway, you must create a default route to direct Internet-bound traffic from your instances to the NAT gateway. The target you specify will be the NAT gateway ID, which follows the format nat-xxxxxxxx

If you use multiple NAT gateways, you can create multiple default routes, each pointing to a different NAT gateway as the target.

Because a NAT gateway doesn't use an ENI, you can't apply a security group to it. You can, however, apply a NACL to the subnet that it resides in.

Amazon Virtual Private Cloud (Amazon VPC)

NAT Instance

A NAT instance is a normal EC2 instance that uses a preconfigured Linux-based AMI. You have to perform the same steps to launch it as you would any other instance. It functions like a NAT gateway in many respects, but there are some key differences.

Unlike a NAT gateway, a NAT instance doesn't automatically scale to accommodate increased bandwidth requirements. Therefore, it's important that you select an appropriately robust instance type. If you choose an instance type that's too small, you must manually upgrade to a larger instance type.

Also, a NAT instance has an ENI, so you must apply a security group to it. You also must remember to assign it a public IP address. Lastly, you must disable the *source/destination check* on the NAT instance's ENI. This allows the NAT instance to receive traffic addressed to an IP other than its own, and it also allows the instance to send traffic using a source IP that it doesn't own.

One advantage of a NAT instance is that you can use it as a bastion host, sometimes called a *jump host*, to connect to instances that don't have a public IP. You can't do this with a NAT gateway.

You must create a default route to direct Internet-bound traffic to the NAT instance. The target of the default route will be the NAT instance's ID, which follows the format i-0a1674fe5671dcb00.

If you want to guard against instance or availability zone failures, it's not as simple as just spinning up another NAT instance. You cannot create multiple default routes pointing to different NAT instances. If you need this level of resiliency, you're better off using NAT gateways instead.

Amazon Virtual Private Cloud (Amazon VPC)

VPC Peering

- You can configure VPC peering to allow instances in one VPC to communicate with VPCs in another over the private AWS network. You may want to do this if you have instances in different regions that need to communicate. You may also want to connect your instances to another AWS customer's instances
- To enable VPC peering, you must set up a *VPC peering connection* between two VPCs. A VPC peering connection is a point-to-point connection between two and only two VPCs. You can have at most one peering connection between a pair of VPCs. Also, peered VPCs must not have overlapping CIDR blocks.
- With one exception, a VPC peering connection allows only instance-to-instance communication. This means an instance in one VPC can use the peering connection only to connect to another instance in the peered VPC. You can't use it to share Internet gateways or NAT devices. You can, however, use it to share a Network Load Balancer (NLB).
- If you have more than two VPCs you need to connect, you must create a peering connection between each pair. You cannot daisy-chain VPC peering connections together and route through them. This configuration is called transitive routing.
- To use a peering connection, you must create new routes in both VPCs to allow traffic to travel in both directions. For each route, the destination prefix should exist within the destination VPC.

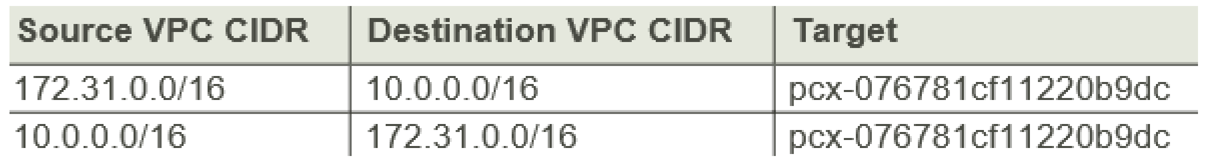
The target of each route must be the peering connection's identifier, which begins with pcx-.

Amazon Virtual Private Cloud (Amazon VPC)

VPC Peering

– Notice that the routes are mirrors of each other. Again, this is to allow traffic in both directions. The destination CIDR doesn't need to exactly match that of the destination VPC. If you want to enable peering only between specific subnets, you can specify the subnet CIDR instead.

– Inter-region VPC peering is not available for some AWS regions. Peering connections between regions have a maximum transmission unit (MTU) of 1500 bytes and do not support IPv6.



Source VPC CIDR	Destination VPC CIDR	Target
172.31.0.0/16	10.0.0.0/16	pcx-076781cf11220b9dc
10.0.0.0/16	172.31.0.0/16	pcx-076781cf11220b9dc

Amazon Virtual Private Cloud (Amazon VPC)

Summary

- The Virtual Private Cloud service provides the networking foundation for EC2 and other AWS services. AWS abstracts some networking components in such a way that their configuration is easier than in a traditional network, but you still need to have a solid grasp of networking fundamentals to architect VPCs.
- In each region, AWS automatically provides a default VPC with default subnets, a main route table, a default security group, and a default NACL. Many use a default VPC for a long time without ever having to configure a VPC from scratch. This makes it all the more important that you as an AWS architect understand how to configure a virtual network infrastructure from scratch. There's a good chance you won't be allowed to modify an infrastructure that was built on top of a default VPC. Instead, you may be tasked with replicating it from the ground up—troubleshooting various issues along the way. Practice what you've learned in this chapter until creating fully functional VPCs becomes second nature to you.
- In a traditional network, you're free to reconfigure server IP addresses, move them to different subnets, and even move them to different physical locations. You have tremendous flexibility to change your plans midstream. When creating a VPC, you don't have that luxury. You must carefully plan your entire application infrastructure up front, not just your network. It's therefore crucial that you understand how all of the VPC and EC2 components fit together.
- You begin by defining a contiguous IP address range for the VPC and representing it as a CIDR. The primary CIDR should ideally be sufficiently large to accommodate all of your instances, but small enough to leave room for a secondary CIDR

Amazon Virtual Private Cloud (Amazon VPC)

Summary

- After that, you have to divvy up your VPC CIDR into subnets. Because a subnet is a container that resides in only one availability zone, you must make up-front decisions about where to place your instances. Once you create an instance in a subnet, you can't move it.
- Prior to launching instances, you need to configure security groups, as every instance's ENI must have at least one attached. One area where you do have some flexibility is with network access control lists. You can associate a NACL with a subnet at any time or forego NACLs altogether.
- If you want your instances to be accessible from the Internet, you must provision an Internet gateway, create a default route, and assign public IP addresses. Those are the basics. If you choose to use a NAT gateway or instance or a VPC peering connection, you'll have to modify multiple route tables

Amazon Virtual Private Cloud (Amazon VPC)

Exam Essentials

- **Be able to determine the correct prefix length for a CIDR block based on the number of IP addresses you need in a VPC or subnet.** Allowed prefix lengths range from /16 to /28. The longer the prefix length, the fewer the number of IP addresses you'll have available.
- **Understand the significance of a subnet.** A subnet is a logical container that holds EC2 instances. Each subnet has a CIDR block derived from the CIDR of the VPC that it resides in. An instance in the subnet must take its private IP address from the subnet's CIDR. But AWS reserves the first four and last IP addresses in every subnet.
- **Know the impact of an availability zone failure.** If a zone fails, every subnet in that zone—and every instance in that subnet—will likewise fail. To tolerate a zone failure, build redundancy into your instance deployments by spreading them across different zones
- **Understand the rules for creating and using elastic network interfaces (ENIs).** Every instance must have a primary network interface with a primary private IP address. Any additional ENI you attach to an instance must be in the same subnet as the primary ENI
- **Be able to create, modify, and use route tables.** Know the purpose of the main route table in a VPC and its relationship to the VPC's subnets. Also understand how to create a public subnet using an Internet gateway and a default route.
- **Know the differences between security groups and network access control lists.** Understand why a stateful security group requires different rules than a stateless NACL to achieve the same result.

Amazon Virtual Private Cloud (Amazon VPC)

Exam Essentials

– **Understand how network address translation works.** Know the difference between NAT that occurs at the Internet gateway and NAT that occurs on a NAT device. NAT that occurs on a NAT device is also called port address translation (PAT). Multiple instances can share a single public IP address using a NAT device.

– **Be able to create and configure VPC peering among multiple VPCs.** Know the limitations of VPC peering connections. VPC peering connections do not support transitive routing or IPv6. Inter-region peering is available for some regions.

Databases

Introduction

- A database allows an application to store, organize, and quickly retrieve data. Although you could use flat files to store data, they become increasingly slow to search as the amount of data grows. By relying on the database to perform these tasks, developers are free to focus on the application without having to directly interact with a filesystem to store and retrieve data.
- Consequently, the availability and performance of a database-backed application depend on the database you choose and how you configure it. Databases come in two flavors: relational and nonrelational. Each of these differs in the way it stores, organizes, and lets you retrieve data, so the type of database you choose depends on the needs of the application.
- In this chapter, you'll learn the differences between these two database types and how to select the right one for your application. You'll also learn how to use the managed database services that AWS provides to get the level of performance and reliability your applications require, as well as how to protect your data and recover it in the event of a database failure.

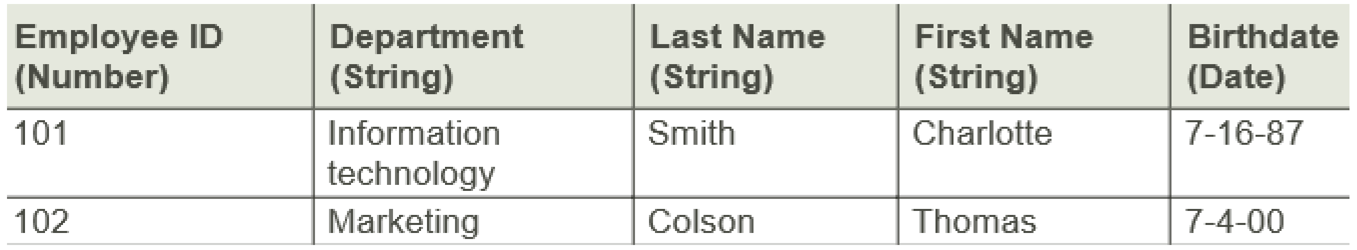
Databases

Relational Databases

A *relational database* contains at least one *table*, which you can visualize as a spreadsheet with columns and rows. In a relational database table, columns may also be called *attributes*, and rows may also be called *records* or *tuples*

Columns and Attributes

– Before you can add data to a relational database table, you must predefine each column's name and what data types it can accept. Columns are ordered, and you can't change the order after you create the table. The ordering creates a relationship between attributes in the table, which is where the term *relational database* comes from.



Employee ID (Number)	Department (String)	Last Name (String)	First Name (String)	Birthdate (Date)
101	Information technology	Smith	Charlotte	7-16-87
102	Marketing	Colson	Thomas	7-4-00

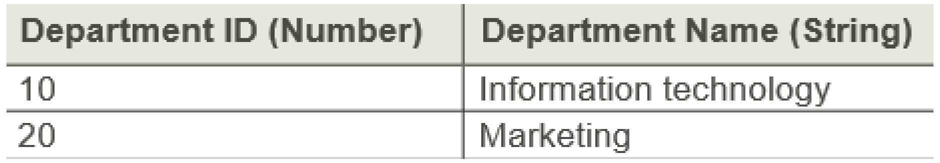
– Data must match the type defined under each column. For example, the Employee ID can't be a letter, because it's defined as a numeric data type. One advantage of a relational database is that you don't have to understand up front how you're going to query the data. As long as the data is there in a consistent format, you can craft queries to get the data you want, the way you want. This makes relational databases good for applications that need to query data in arbitrary columns and customize how that data is presented. For example, you could query the database to return the birthdate of every employee whose first name is Charlotte.

– You can add more columns to a table after creating it. You can also delete columns, but deleting a column entails deleting all of the data stored under the column. If you delete the First Name column, it will remove first name data for all employees in the table.

Databases

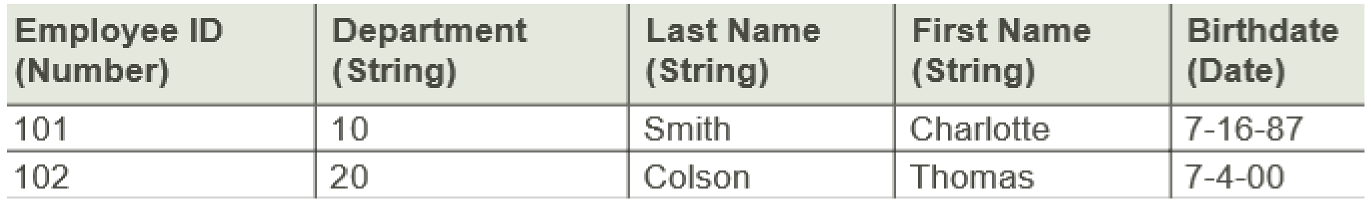
Using Multiple Tables

Storing all data in a single table can lead to unnecessary duplication, needlessly increasing the size of the database and making queries slower. Hence, it's common for applications to use multiple related tables. Using the preceding example, if 50 employees work in the information technology department, the string "Information technology" appears in the table 50 times—once for each record. To avoid this wasted space, you may create a separate table to hold department names



Department ID (Number)	Department Name (String)
10	Information technology
20	Marketing

– Instead of putting the department names in every employee record, you would create a single record for each department in the Departments table. The Employees table would then refer to each department using the Department ID,



Employee ID (Number)	Department (String)	Last Name (String)	First Name (String)	Birthdate (Date)
101	10	Smith	Charlotte	7-16-87
102	20	Colson	Thomas	7-4-00

– In this relationship, the Departments table is the *parent table*, and the Employees table is the *child table*. Each value in the Department column in the Employees table refers to the Department ID in the Departments table. Notice that the data type for the Department column is still a string. Although you could change the data type to a number, it's not necessary. The Department ID in the Departments table is called the *primary key*, and it must be unique in the table so that it can uniquely identify a row. The Employees table refers to this as a *foreign key*

Databases

Storing Data

The INSERT statement allows you to insert data directly into a table. If you need to load a large number of records, you can use the COPY command to copy data from a properly formatted file into the table you specify.

Online Transaction Processing vs. Online Analytic Processing

Depending on its configuration, a relational database can fall into one of two categories: *online transaction processing* (OLTP) or *online analytic processing* (OLAP).

OLTP

OLTP databases are suited to applications that read and write data frequently, on the order of multiple times per second. They are optimized for fast queries, and those queries tend to be regular and predictable. Depending on the size of the database and its performance requirements, an OLTP database may have intense memory requirements so that it can store frequently accessed portions of tables in memory for quick access. Generally, a single server with ample memory and compute power handles all writes to an OLTP database. An OLTP database would be a good candidate for backing an online ordering system that processes hundreds of orders a minute

OLAP

OLAP databases are optimized for complex queries against large data sets. As a result, OLAP databases tend to have heavy compute and storage requirements. In *data warehousing* applications, it's common to aggregate multiple OLTP databases into a single OLAP database. For example, in an OLTP database for an employee management system, employee data may be spread out across multiple tables. At regular but infrequent intervals, a data warehouse would aggregate these tables into a single table in an OLAP database. This makes it easier to write queries against the data and reduces the amount of time it takes to process such a query. With a large OLAP database, it's common for multiple database servers to share the computational load of complex queries. In a process called *partitioning*, each server gets a portion of the database for which it's responsible.

Databases

Amazon Relational Database Service

The Amazon Relational Database Service (RDS) is a managed database service that lets you run relational database systems in the cloud. RDS takes care of setting up the database system, performing backups, ensuring high availability, and patching the database software and the underlying operating system. RDS also makes it easy to recover from database failures, restore data, and scale your databases to achieve the level of performing and availability that your application requires

To deploy a database using RDS, you start by configuring a *database instance*, which is an isolated database environment. A database instance exists in a virtual private cloud (VPC) that you specify, but unlike an EC2 instance, AWS fully manages database instances. You can't SSH into them, and they don't show up under your EC2 instances.

Database Engines

A database engine is simply the software that stores, organizes, and retrieves data in a database. Each database instance runs only one database engine. RDS offers the following six database engines to choose from:

- **MySQL** MySQL is designed for OLTP applications such as blogs and ecommerce. RDS offers the latest MySQL Community Edition versions, including 5.5, 5.6, and 5.7. MySQL offers two storage engines—MyISAM and InnoDB—but you should use the latter as it's the only one compatible with RDS-managed automatic backups.
- **MariaDB** MariaDB is a drop-in binary replacement for MySQL. It was created over concerns about MySQL's future after Oracle acquired the company that developed it. RDS offers many versions of MariaDB, ranging from 10.0.17 through 10.2. MariaDB supports the XtraDB and InnoDB storage engines, but AWS recommends using the latter for maximum compatibility with RDS

Databases

Amazon Relational Database Service

- **Oracle** Oracle is one of the most widely deployed relational database management systems. Some applications expressly require an Oracle database.
- **PostgreSQL** PostgreSQL advertises itself as the most Oracle-compatible open source database. This is a good choice when you have in-house applications that were developed for Oracle but want to keep costs down. RDS offers versions of PostgreSQL ranging from 9.3.12-R1 through 10.4-R1
- **Amazon Aurora** Aurora is Amazon's drop-in binary replacement for MySQL and PostgreSQL. Aurora offers better write performance than both by using a virtualized storage layer that reduces the number of writes to the underlying storage.
- Depending on the edition you choose, Aurora is compatible with PostgreSQL or MySQL import and export tools and snapshots. Aurora is designed to let you seamlessly migrate from an existing deployment that uses either of those two open source databases. For MySQL-compatible editions, Aurora supports only the InnoDB storage engine. Also, the Aurora Backtrack feature for MySQL lets you, within a matter of seconds, restore your database to any point in time within the last 72 hours.
- **Microsoft SQL Server** RDS offers multiple Microsoft SQL Server versions: 2008 R2, 2012, 2014, 2016, and 2017. For the edition, you can choose Express, Web, Standard, and Enterprise. The variety of flavors makes it possible to migrate an existing SQL Server database from an on-prem deployment to RDS without having to perform any database upgrades

Databases

Licensing Considerations

RDS provides two models for licensing the database engine software you run. The *license included* model covers the cost of the license in the pricing for an RDS instance. The *bring your own license* (BYOL) model requires you to obtain a license for the database engine you run.

License Included MariaDB and MySQL use the GNU General Public License (GPL) v2.0, and PostgreSQL uses the PostgreSQL license, all of which allow for free use of the respective software

All versions and editions of Microsoft SQL Server that you run on RDS include a license, as do Oracle Database Standard Edition One (SE1) and Standard Edition Two (SE2).

Bring Your Own License The following Oracle Database editions allow you to bring your own license

- Enterprise Edition (EE)
- Standard Edition (SE)
- Standard Edition One (SE1)
- Standard Edition Two (SE2)

Database Option Groups

Database engines offer various features to help you manage your databases and improve security. *Option groups* let you specify these features—called *options*—and apply them to one or more instances. Options require more memory, so make sure your instances have ample memory and enable only the options you need.

The options available for a database option group depend on the engine. Microsoft SQL Server and Oracle offer *transparent data encryption* (TDE), which causes the engine to encrypt data before writing it to storage. MySQL and MariaDB offer an audit plugin that lets you log user logons and queries run against your databases

Databases

Database Instance Classes

When launching a database instance, you must decide how much processing power, memory, network bandwidth, and disk throughput it needs. RDS offers a variety of database instance classes to meet the diverse performance needs of different databases. If you get it wrong or if your needs change, you can switch your instance to a different class. RDS divides database instance classes into the following three types.

Standard

Memory Optimized

Burst Capable (Burstable)

Storage

Selecting the right storage for your database instance is about more than just ensuring you have enough disk space. You also have to decide how fast the storage must be to meet the performance requirements of your database-backed application.

Understanding Input/Output Operations Per Second

AWS measures storage performance in *input/output operations per second* (IOPS). An input/output (I/O) operation is either a read from or write to storage. All things being equal, the more IOPS you can achieve, the faster your database can store and retrieve data.

RDS allocates you a number of IOPS depending on the type of storage you select, and you can't exceed this threshold. The speed of your database storage is limited by the number of IOPS allocated to it. The amount of data you can transfer in a single I/O operation depends on the page size that the database engine uses. To understand how many IOPS you need, you first need to understand how much disk throughput you need.

Databases

Understanding Input/Output Operations Per Second

MySQL and MariaDB have a page size of 16 KB. Hence, writing 16 KB of data to disk would constitute one I/O operation. Oracle, PostgreSQL, and Microsoft SQL Server use a page size of 8 KB. Writing 16 KB of data using one of those database engines would consume two I/O operations. The larger the page size, the more data you can transfer in a single I/O operation.

Assuming a 16 KB page size, suppose your database needed to read 102,400 KB (100 MB) of data every second. To achieve this level of performance, your database would have to be able to read 6,400 16 KB pages every second. Because each page read counts as one I/O operation, your storage and instance class would need to be able to sustain 6,400 IOPS. Notice the inverse relationship between IOPS and page size: The larger your page size, the fewer IOPS you need to achieve the same level of throughput.

Note Things get interesting when you move beyond a 32 KB page size. If your database engine writes more than 32 KB in a single I/O operation, AWS counts that as more than one I/O operation. For example, reading or writing a 64 KB page would count as two I/O operations. A 128 KB page would count as four I/O operations

The number of IOPS you can achieve depends on the type of storage you select. RDS offers the following three different types of storage.

Databases

General-Purpose SSD

For most databases, *general-purpose SSD* (gp2) storage is sufficient. It's fast, giving you single-digit millisecond latency. You can allocate a volume of up to 16 TB. For each gigabyte of data that you allocate to a volume, RDS allocates that volume a baseline performance of three IOPS, up to a total of 10,000 IOPS per volume. A 20 GB volume would get 60 IOPS, while a 100 GB volume would get 300 IOPS. This means that the larger your volume, the better performance you'll get. Note that the minimum storage volume you can create depends on the database engine. For SQL Server Enterprise and Standard, it's 200 GB, and for all others, it's 20 GB.

The maximum throughput the gp2 storage type offers is 1,280 Mbps (160 MBps). To achieve this, two things have to fall into place. First, your instance must support a disk throughput of at least that much. For example, the db.m4.4xlarge instance class has a maximum throughput of 2,000 Mbps, so it would suffice. Also, you must allocate a sufficient number of IOPS to sustain this throughput. Suppose you're running MariaDB with a page size of 16 KB (128 Kb or 0.128 Mb). To determine the number of IOPS you'll need to sustain 1,280 Mbps of disk throughput, you'd divide the bandwidth by the page size, as follows:

1280 Mbps/0.128 Mb = 10,000 IOPS

To achieve 1,280 Mbps disk throughput to a volume, it would need 10,000 IOPS allocated. Note that this is the maximum number of IOPS possible with gp2. To achieve this many IOPS, your volume would have to be 3,333.3 GB or about 3.34 TB.

If you think you might occasionally need up to 3000 IOPS, but don't need a lot of storage, you don't have to over-allocate storage just to get your desired number of IOPS. Volumes smaller than 1 TB can temporarily burst to 3,000 IOPS. The duration of the burst is determined by the following formula:

Databases

General-Purpose SSD

Burst duration in seconds = (Credit balance)/[3,000 – 3 * (storage size in GB)]

When you initially boot a database instance, you get a credit balance of 5,400,000 IOPS. Anytime your instance uses IOPS above and beyond its baseline, it will dip into the credit balance. Once the credit balance is depleted, you can no longer burst. For example, with a 200 GB volume, the burst duration would be 2,250 seconds or 37.5 minutes.

The credit balance is replenished at a rate of one baseline IOPS every second. For example, if you have a 200 GB volume with a baseline IOPS of 600, your credit balance increases by 600 IOPS per second, up to the maximum of 5,400,000.

Databases

Provisioned IOPS SSD (io1)

If you're not excited about the rather confusing math involved with using gp2 storage, RDS provides a more straightforward option. *Provisioned IOPS SSD* lets you simply allocate the number of IOPS you need when you create your instance. There's no concept of bursting in io1 storage. The number of IOPS you provision is what you get and what you pay for, whether you use it or not. This makes it useful for OLTP databases that require consistent low-latency performance

If you use a standard or memory optimized instance class, RDS guarantees that you'll achieve within 10 percent of the provisioned IOPS for 99.9 percent of the year. That means you may get less than your specified number of IOPS for only about 2 hours and 45 minutes out of the year.

Assuming that you pick a standard instance with 4,000 Mbps throughput and use a database engine with a 16 KB page size, you can achieve up to 31,250 IOPS. To get this, you should provision 32,000 IOPS when you create the instance, as you must specify provisioned IOPS in increments of 1,000.

The maximum number of IOPS you can achieve and how much storage you can allocate are constrained by the database engine you select. Oracle, PostgreSQL, MariaDB, MySQL, and Aurora let you choose 100 GB to 16 TB of storage and allocate 1,000 to 40,000 provisioned IOPS. Microsoft SQL Server gives you up to 16 TB of storage and lets you choose between 1,000 and 32,000 provisioned IOPS. The ratio of storage in gigabytes to IOPS must be at least 50:1. For example, if you want 32,000 IOPS, you must provision at least 640 GB of storage.

Databases

Magnetic Storage

RDS offers magnetic storage for backward compatibility with older instances. It's limited to a maximum size of 4 TB and 1,000 IOPS

Read Replicas

If your database instance doesn't meet your performance requirements, you have a few options, depending on where the bottleneck is.

As mentioned previously, if your memory, compute, network speed, or disk throughput are the issue, you can scale your database instance vertically by upgrading to a larger instance class. *Scaling vertically*—also called *scaling up*—is a straightforward approach. You simply throw more resources at your database instance and don't have to make any changes to your application or databases.

Scaling horizontally, also known as *scaling out*, entails creating additional database instances called read replicas. All database engines except for Oracle and Microsoft SQL Server support read replicas. Aurora exclusively supports a specific type of read replica called an *Aurora replica*.

A read replica is another database instance that services only queries against the database. A read replica takes some of the query load off of the *master database instance*, which remains solely responsible for writing data to the database. Read replicas are useful for read-heavy applications

You can have up to five read replicas and up to fifteen Aurora replicas. Data from the master is asynchronously replicated to each read replica, meaning that there's a delay between when the data is written to the database by the master and when it shows up on the replica. This makes read replicas unsuitable for disaster recovery. For MySQL, you can set a replication delay.

Databases

Amazon Relational Database Service

When you create a read replica, RDS gives you a read-only endpoint which is a domain name that resolves only to your read replica. If you have multiple replicas, RDS will load balance the connection to one of them. If you have reporting and analysis tools that only need to read data, you would point them to a read-only endpoint.

A read replica and the master may be in different availability zones, and even in different regions. In the event of the master instance failing, you can promote a read replica to the master. But keep in mind that because of the asynchronous nature of replication, you may lose data this way.

Databases

High Availability (Multi AZ)

To keep your database continuously available in the event of a database instance outage, you can deploy multiple database instances in different availability zones using what RDS calls a *multi-AZ deployment*. In a multi-AZ deployment, you have a *primary database instance* in one availability zone that handles reads and writes to the database, and you have a *standby database instance* in a different availability zone. If the primary instance experiences an outage, it will failover to the standby instance, usually within two minutes.

Here are a few possible causes for a database instance outage:

- Availability zone outage
- Changing a database instance type
- Patching of the instance's operating system

You can configure multi-AZ when you create a database instance or later. All database engines support multi-AZ but implement it slightly differently.

If you enable multi-AZ after creating your instance, you'll experience a significant performance hit, so be sure to do it during a maintenance window.

Databases

Multi-AZ with Oracle, PostgreSQL, MariaDB, MySQL, and Microsoft SQL Server

In this multi-AZ deployment, all instances must reside in the same region. RDS synchronously replicates data from the primary to the standby instance. This replication can introduce some latency, so be sure to use EBS-optimized instances and provisioned IOPS SSD storage.

The standby instance is not a read replica and cannot serve read traffic. Your application connects to the endpoint domain name for the primary. When a failover occurs, RDS changes the DNS record of the endpoint to point to the standby. The only thing your application has to do is reconnect to the endpoint.

If using the bring your own license model for Oracle, you must possess a license for both the primary and standby instances.

For MySQL and MariaDB, you can create a multi-AZ read replica in a different region. This lets you failover to a different region.

Databases

Automated Snapshots

RDS can automatically create snapshots of your instances daily during a 30-minute backup window. You can customize this window or let RDS choose it for you. Because taking a snapshot impacts performance, choose a time when your database is least busy. If you let RDS select the backup window, it will randomly select a 30-minute window within an 8-hour block that varies by region

Enabling automatic backups enables *point-in-time recovery*, which archives database change logs to S3 every 5 minutes. In the event of a failure, you'll lose only up to five minutes' worth of data. Restoring to a point-in-time can take hours, depending on how much data is in the transaction logs.

RDS keeps automated snapshots for a limited period of time and then deletes them. You can choose a *retention period* between 1 day and 35 days. The default is 7 days. To disable automated snapshots, set the retention period to zero. Note that disabling automated snapshots immediately deletes all existing automated snapshots and disables point-in-time recovery. Also, if you change the retention period from zero to any other value, it will trigger an immediate snapshot.

You can also manually take a snapshot of your database instance. Unlike automated snapshots, manual snapshots stick around until you delete them. If you delete an instance, RDS will prompt you to take a final snapshot. It will also prompt you to retain automated snapshots. RDS will keep the final snapshot and all manual snapshots. If you choose not to retain automated backups, it will immediately delete any automated snapshots.

Databases

Maintenance Items

Because RDS is a managed service, it's the responsibility of AWS to handle patching and upgrades. AWS routinely performs these maintenance items on your database instances.

Maintenance items include operating system security and reliability patches. These generally occur once every few months. Database engine upgrades also may occur during a maintenance window. When AWS begins to support a new version of a database engine, you can choose to upgrade to it. Major version upgrades may contain database changes that aren't backward compatible. As such, if you want a major version upgrade, you must apply it manually. AWS may automatically apply minor version changes that are nonbreaking.

You can determine when these maintenance tasks take place by specifying a 30-minute weekly maintenance window. The window cannot overlap with the backup window. Even though the maintenance window is 30 minutes, it's possible for tasks to run beyond this.

Amazon Redshift

Redshift is a managed data warehouse solution designed for OLAP databases. Although it's based on PostgreSQL, it's not part of RDS. Redshift uses columnar storage, meaning that it stores the values for a column close together. This improves storage speed and efficiency and makes it faster to query data from individual columns. Redshift supports ODBC and JDBC database connectors.

Redshift uses compression encodings to reduce the amount of size each column takes up in storage. You can apply compression manually on a column-by-column basis. Or if you use the copy command to import data from a file into a Redshift database, Redshift will determine which columns to compress.

Databases

Compute Nodes

A Redshift cluster contains one or more *compute nodes* that are divided into two categories. *Dense compute* nodes can store up to 326 TB of data on magnetic storage. *Dense storage* nodes can store up to 2 PB of data on fast SSDs.

If your cluster contains more than one compute node, Redshift also includes a *leader node* to coordinate communication among the compute nodes, as well as to communicate with clients. A leader node doesn't incur any additional charges.

Data Distribution Styles

Rows in a Redshift database are distributed across compute nodes. How the data is distributed depends on the distribution style. In *EVEN* distribution, the leader node spreads the data out evenly across all compute nodes. This is the default style. *KEY* distribution spreads the data according to the value in a single column. Columns with the same value are stored on the same node. In the *ALL* distribution, every table is distributed to every compute node.

Databases

Nonrelational (No-SQL) Databases

Nonrelational databases are designed to consistently handle tens of thousands of transactions per second. Although they can store the same data you'd find in a relational database, they're optimized for so-called unstructured data. Unstructured data is an unfortunate term, as all data you store in any database has some structure. A more accurate description would be multistructured data. The data you store in a nonrelational database can vary in structure, and that structure can change over time.

Nonrelational and relational databases have many elements in common. Nonrelational databases—also known as no-SQL databases—consist of collections that are confusingly also sometimes called *tables*. Within a table, you store items, which are similar to rows or tuples in a relational database. Each item consists of at least one attribute, which is analogous to a column in a SQL database. An attribute consists of a unique name called a *key*, a data type, and a value. Attributes are sometimes called *key-value pairs*.

Storing Data

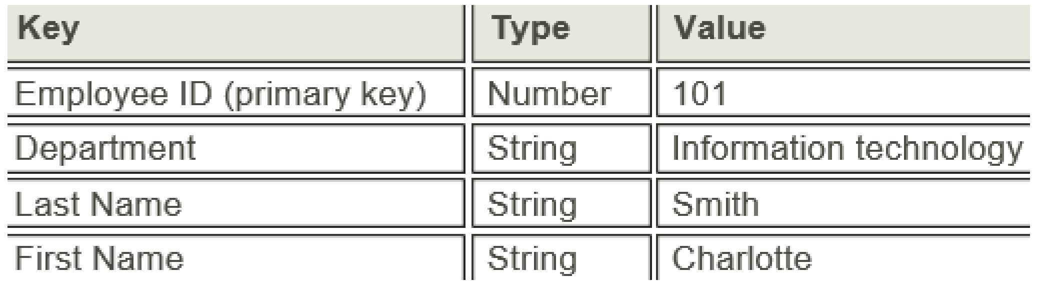
One of the biggest differences between a relational and nonrelational database is that nonrelational databases are *schemaless* and don't require all items in a table to have the same attributes. Each item requires a primary key attribute whose value must be unique within the table. The purpose of the primary key is to uniquely identify an item and provide a value by which to sort items. Nonrelational databases are flexible when it comes to the type of data you can store. With the exception of the primary key attribute, you don't have to define attributes when you create a table. You create attributes on the fly when you create or modify an item. These attributes are unordered and hence have no relation to each other, which is they're called nonrelational.

Databases

Nonrelational databases do not give you a way to split data across tables and then merge it together at query time. Therefore, an application will generally keep all of its data in one table. This can lead to the duplication of data, which in a large database can incur substantial storage costs.

Querying Data Database Engines

The trade-off for having flexibility to store unstructured data comes in terms of being more limited in your queries. Nonrelational databases are optimized for queries based on the primary key. Queries against other attributes are slower, making nonrelational databases inappropriate for complex or arbitrary queries. Prior to creating a table, you need to understand the exact queries that you're going to need to perform against the data. Consider the following item:



Key	Type	Value
Employee ID (primary key)	Number	101
Department	String	Information technology
Last Name	String	Smith
First Name	String	Charlotte

If you wanted to list every department that has an employee named Charlotte, it would be difficult to obtain this using a nonrelational database. Because items are sorted by Employee ID, the system would have to scan through every item to locate all items that have an attribute First Name with a value of Charlotte. And because the data in each item is unstructured, it may require searching through every attribute. It would then have to determine which of these items contain a Department attribute. Such a query would be slow and computationally expensive.

Databases

Types of Nonrelational Databases

You may hear nonrelational databases divided into categories such as key-value stores, document-oriented stores, and graph databases. But all nonrelational databases are key-value store databases.

A document-oriented store is a particular application of a nonrelational database that analyzes the contents of a document stored as a value and extracts metadata from it.

A graph database analyzes relationships between attributes in different items. This is different than a relational database that enforces relationships between records. A graph database discovers these relationships in unstructured data.

Databases

Dynamo DB

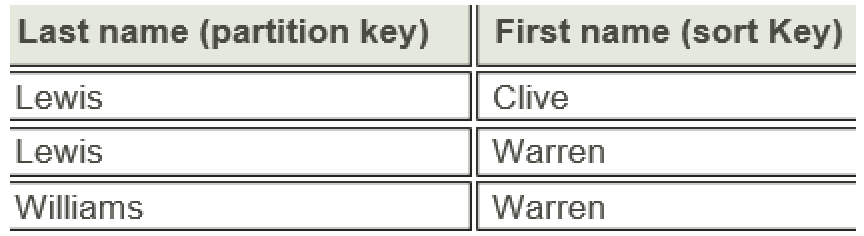
DynamoDB is a managed nonrelational database service that can handle thousands of reads and writes per second. It achieves this level of performance by spreading your data across multiple *partitions*. A partition is an allocation of storage for a table, and it's backed by solid-state drives in multiple availability zones

Partition and Hash Keys

When you create a table, you must specify a primary key and a data type. Because the primary key uniquely identifies an item in the table, its value must be unique within the table. There are two types of primary keys you can create

A *partition key*, also known as a *hash key*, is a primary key that contains a single value. When you use only a partition key as a primary key, it's called a *simple primary key*. Good candidates for a partition key would be an email address, a unique username, or even a randomly generated identifier. A partition key can store no more than 2,048 B.

A primary key can also be a combination of two values: a partition key and a *sort (*or *range) key*. This is called a *composite primary key*. The partition key doesn't have to be unique, but the combination of the partition key and sort key must be unique. For example, a person's last name could be the partition key, while the first name could be the sort key. Using this approach, you could use the following values for a composite primary key for a table:



Last name (partition key)	First name (sort Key)
Lewis	Clive
Lewis	Warren
Williams	Warren

Databases

Dynamo DB

Partition and Hash Keys

Neither the last name Lewis nor the first name Warren is unique in the table. But combining the partition and sort keys together creates a unique primary key.

DynamoDB distributes your items across partitions based on the primary key. Using the preceding example, items with the last name Lewis would all be stored on the same partition. DynamoDB would arrange the items in ascending order by the sort key. Note that a sort key can store no more than 1,024 B.

When a lot of read or write activity occurs against an item, the partition the item exists in is said to be a *hot partition*. This can negatively affect performance. To avoid hot partitions, try to make your partition keys as unique as possible. For example, if you're storing log entries, using the current date as a partition key will result in a different hot partition every day. Instead, you may consider using a timestamp that changes frequently.

Databases

Attributes and Items

Each key-value pair composes an attribute, and one or more attributes make up an item. DynamoDB can store an item size of up to 400 KB, which is roughly equivalent to 50,000 English words!

At a minimum, every item contains a primary key and corresponding value. When you create an attribute, you must define the data type. Data types fall into the following three categories:

Scalar A *scalar data type* can have only one value. The string data type can store up to 400 KB of Unicode data with UTF-8 encoding. A string must always be greater than zero.

The number data type stores positive or negative numbers up to 38 significant digits. DynamoDB trims leading and trailing zeros.

The binary data type stores binary data in base-64 encoded format. Like the string type, it's limited by the maximum item size to 400 KB.

The Boolean data type can store a value of either true or false.

The null data type is for representing an attribute with an undefined or unknown value. Oddly, it must contain a value of null.

Set A *set data type* holds an unordered list of scalar values. The values must be unique within a set, and a set must contain at least one value. You can create number sets, string sets, and binary sets.

Document *Document data types* are designed to hold different types of data that fall outside the constraints of scalar and set data types. You can nest document types together up to 32 levels deep.

A list document type can store an ordered collection of values of any type. For example, you could include the following in the value of a list document:

Databases

Dynamo DB

Chores: ["Make coffee", Groceries: ["milk", "eggs", "cheese"], "Pay bills", Bills: [water: [60], electric: [100]]]

Notice that the Chores list contains string data, numeric data, and nested lists.
A map data type can store an unordered collection of key-value pairs in a format similar to JavaScript Object Notation (JSON). As with a list, there are no restrictions on the type of data you can include. The following is an example of a map that contains a nested list and a nested map:

```
{ Day: "Friday",
Chores: [ "Make coffee",
 "Groceries", {
 milk: { Quantity: 1 },
eggs: { Quantity: 12 }
}
"Mow the lawn"],
}
```

Databases

Dynamo DB

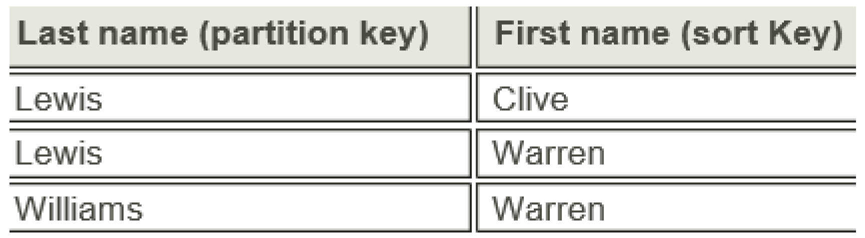
DynamoDB is a managed nonrelational database service that can handle thousands of reads and writes per second. It achieves this level of performance by spreading your data across multiple *partitions*. A partition is an allocation of storage for a table, and it's backed by solid-state drives in multiple availability zones

Partition and Hash Keys

When you create a table, you must specify a primary key and a data type. Because the primary key uniquely identifies an item in the table, its value must be unique within the table. There are two types of primary keys you can create

A *partition key*, also known as a *hash key*, is a primary key that contains a single value. When you use only a partition key as a primary key, it's called a *simple primary key*. Good candidates for a partition key would be an email address, a unique username, or even a randomly generated identifier. A partition key can store no more than 2,048 B.

A primary key can also be a combination of two values: a partition key and a *sort (*or *range) key*. This is called a *composite primary key*. The partition key doesn't have to be unique, but the combination of the partition key and sort key must be unique. For example, a person's last name could be the partition key, while the first name could be the sort key. Using this approach, you could use the following values for a composite primary key for a table:



Last name (partition key)	First name (sort Key)
Lewis	Clive
Lewis	Warren
Williams	Warren

Databases

Throughput Capacity

When creating a table, you must specify the number of reads and writes per second your application will require. This is called *provisioned throughput*. DynamoDB reserves partitions based on the number of *read capacity units* (RCUs) and *write capacity units* (WCUs) you specify when creating a table.

When you read an item from a table, that read may be *strongly consistent* or *eventually consistent*. A strongly consistent read always gives you the most up-to-date data, while an eventually consistent read may produce stale data that does not reflect data from a recent write operation. Whether you use strongly or eventually consistent reads depends on whether your application can tolerate reading stale data. You need to understand whether you need strongly or eventually consistent reads when deciding how much throughput to provision.

For an item up to 4 KB in size, one RCU buys you one strongly consistent read per second. To read an 8 KB item every second using a strongly consistent read, you'd need two RCUs.

If you use an eventually consistent read, one RCU buys you two eventually consistent reads per second. To read an 8 KB item every second using an eventually consistent read, you'd need only one RCU.

When it comes to writing data, one WCU gives you one write per second for an item up to 1 KB in size. This means if you need to write 100 items per second, each item being less than 1 KB, you'd need to provision 100 WCUs. If you need to write 10 items per second, each item being 2 KB, then you'd need 20 WCUs.

The throughput capacity you specify is an upper limit of what DynamoDB delivers. If you exceed your capacity, DynamoDB may throttle your request and yield an "HTTP 400 (Bad request)" error. AWS SDKs have built-in logic to retry throttled requests, so having a request throttled won't prevent your application from reading or writing data, but it will slow it down.

Databases

Auto Scaling

If you're unsure exactly how much throughput you need to provision for a table or if you anticipate your throughput needs will vary over time, you can configure Auto Scaling to automatically increase your provisioned throughput when it gets close to hitting a defined threshold.

To configure Auto Scaling, you specify a minimum and maximum RCU and WCU. You also specify a desired utilization percentage. DynamoDB will automatically adjust your RCU and WCU to keep your utilization at this percentage. For example, suppose you set a utilization of 70 percent, a minimum RCU of 10, and a maximum RCU of 50. If you consume 21 RCU, Auto Scaling will adjust your provisioned capacity to around 30 RCU. If your consumption drops to 14, Auto Scaling will reduce your provisioned throughput to 20 RCU.

Setting the right utilization is a balancing act. The higher you set your utilization, the more likely you are to exceed your provisioned capacity. If that happens, your requests may get throttled. On the other hand, if you set your utilization too low, you will end up paying for capacity you don't need.

Reserved Capacity

If you need 100 or more WCU or RCU, you can purchase reserved throughput capacity to save money. You must reserve RCU and WCU separately, and you're limited to 100,000 units of each. You have to pay a one-time fee and commit to a period of one or three years

Databases

Reading Data

DynamoDB provides two different operations to let you read data from a table. A *scan* lists all items in a table. It's a read-intensive operation and can potentially consume all of your provisioned capacity units. A *query* returns an item based on the value of the partition key. When performing a query, the value of the partition key you search for must exactly match that of an item. If your table contains a sort key, you may optionally query by the sort key as well. For the sort key, you have more flexibility. You can search by exact value, a value greater than or less than the key, a range of values, or the beginning of the value.

Secondary Indexes

Secondary indexes solve two issues with querying data from DynamoDB. When you query for a particular item, you must specify a partition key exactly. For example, the Author table you created earlier has LastName as the partition key and FirstName as the sort key. Secondary indexes let you look up data by an attribute other than the table's primary key. Think of a secondary index as a copy of some of the attributes in a table. The table that the index gets its data from is called the *base table*.

When you create a secondary index, you can choose which attributes get copied from the base table into the index. These are called *projected attributes*. A secondary index always includes the partition and sort key attributes from the base table. You can choose to copy just the partition and sort keys and their values, the keys plus other attributes, or everything. This lets you extract only the data you need. There are two types of secondary indexes.

Databases

Global Secondary Index

You can create a *global secondary index* (GSI) any time after creating a table. In a global secondary index, the partition and hash keys can be different than the base table. The same rules for choosing a primary key still apply. You want the primary key of your index to be as unique as possible. If you use a composite primary key, items having partition keys with the same value will be stored on the same partition.

When reading from a global secondary index, reads are always eventually consistent. If you add an item to a table, it may not immediately get copied to the secondary index.

Databases

Summary

Whether you implement a relational or nonrelational database depends solely on the application that will use it. Relational databases have been around a long time, and many application developers default to modeling their data to fit into a relational database. Applications use database-specific SDKs to interact with the database, so often the needs of the application mandate the specific database engine required. This is why AWS RDS offers six of the most popular database engines and sports compatibility with a wide range of versions. The idea is to let you take an existing database and port it to RDS without having to make any changes to the application.

The nonrelational database is a more recent invention. DynamoDB is Amazon's proprietary nonrelational database service. Unlike applications designed for relational databases, an application designed for a nonrelational database generally cannot be ported from an on-premises deployment to DynamoDB without some code changes. You're therefore more likely to find cloud-native applications using DynamoDB. When developing or refactoring an application to use DynamoDB, there's a good chance that developers will need to consult with you regarding how to design the database. It's critical in this case that you understand how to choose partition and sort keys and data types and how to allocate throughput capacity to meet the performance needs of the application.

Regardless of which database you use, as with any AWS service, you as an AWS architect are responsible for determining your performance and availability requirements and implementing them properly.

Databases

Exam Essentials

Understand the differences between relational and nonrelational databases. A relational database requires you to specify attributes up front when you create a table. All data you insert into a table must fit into the predefined attributes. It uses the Structured Query Language (SQL) to read and write data, and so it's also called a SQL database. A nonrelational database only requires you to specify a primary key attribute when creating a table. All items in a table must include a primary key but can otherwise have different attributes. Nonrelational—also called no-SQL databases—store unstructured data.

Know the different database engines RDS supports. RDS supports all of the most popular database engines—MySQL, MariaDB, Oracle, PostgreSQL, Amazon Aurora, and Microsoft SQL Server. Understand the difference between the bring-your-own-license and license-included licensing models. Know which database engines support which licensing models.

Be able to select the right instance class and storage type given specific storage requirements. Memory and storage tend to be the constraining factors for relational databases, so it's crucial that you know how to choose the right instance class and storage type based on the performance needs of a database. Know the three different instance classes: standard, memory optimized, and burstable. Also know how these relate to the three different storage types: general-purpose SSD (gp2), provisioned IOPS SSD (io1), and magnetic.

Understand the differences between multi-AZ and read replicas. Both multi-AZ and read replicas involve creating additional database instances, but there are some key differences. A read replica can service queries, while a standby instance in a multi-AZ deployment cannot. A master instance asynchronously replicates data to read replicas, while in a multi-AZ configuration, the primary instance synchronously replicates data to the standby. Understand how Aurora replicas work and how Aurora multi-AZ differs from multi-AZ with other database engines.

Databases

Exam Essentials

Be able to determine the appropriate primary key type for a DynamoDB table. DynamoDB tables give you two options for a primary key. A simple primary key consists of just a partition key and contains a single value. DynamoDB distributes items across partitions based on the value in the partition key. When using a simple primary key, the partition key must be unique within a table. A composite primary key consists of a partition key and a sort key. The partition key does not have to be unique, but the combination of the partition key and the sort key must be.

Know how DynamoDB throughput capacity works. When you create a table, you must specify throughput capacity in write capacity units and read capacity units. How many read capacity units a read consumes depends on two things: whether the read is strongly or eventually consistent and how much data you read within one second. For an item of up to 4 KB in size, one strongly consistent read consumes one read capacity unit. Eventually consistent reads consume half of that. When it comes to writes, one write capacity unit lets you write up to one 1 KB item per second.

Authentication and Authorization—AWS Identity and Access Management

Introduction

– Your AWS resources are probably your company's crown jewels, so you definitely don't want to leave them unprotected. But you also can't lock them down so tightly that even your admins and customers can't get in.

– Finding the perfect balance is possible. Getting there will have a lot to do with the way you *authenticate* user requests to confirm they're legitimate and then *authorize* no more and no less than the exact access they'll need. On AWS, authentication and authorization are primarily handled by Identity and Access Management (IAM).

– In this chapter, you're going to learn about IAM identities—which are sometimes described as *principals*. An identity represents an AWS user or a role. Roles are identities that can be temporarily assigned to an application, service, user, or group.

– Identities can also be federated. That is, users or applications without AWS accounts can be authenticated and given temporary access to AWS resources using an external service such as Kerberos, Microsoft Active Directory, or the Lightweight Directory Access Protocol (LDAP).

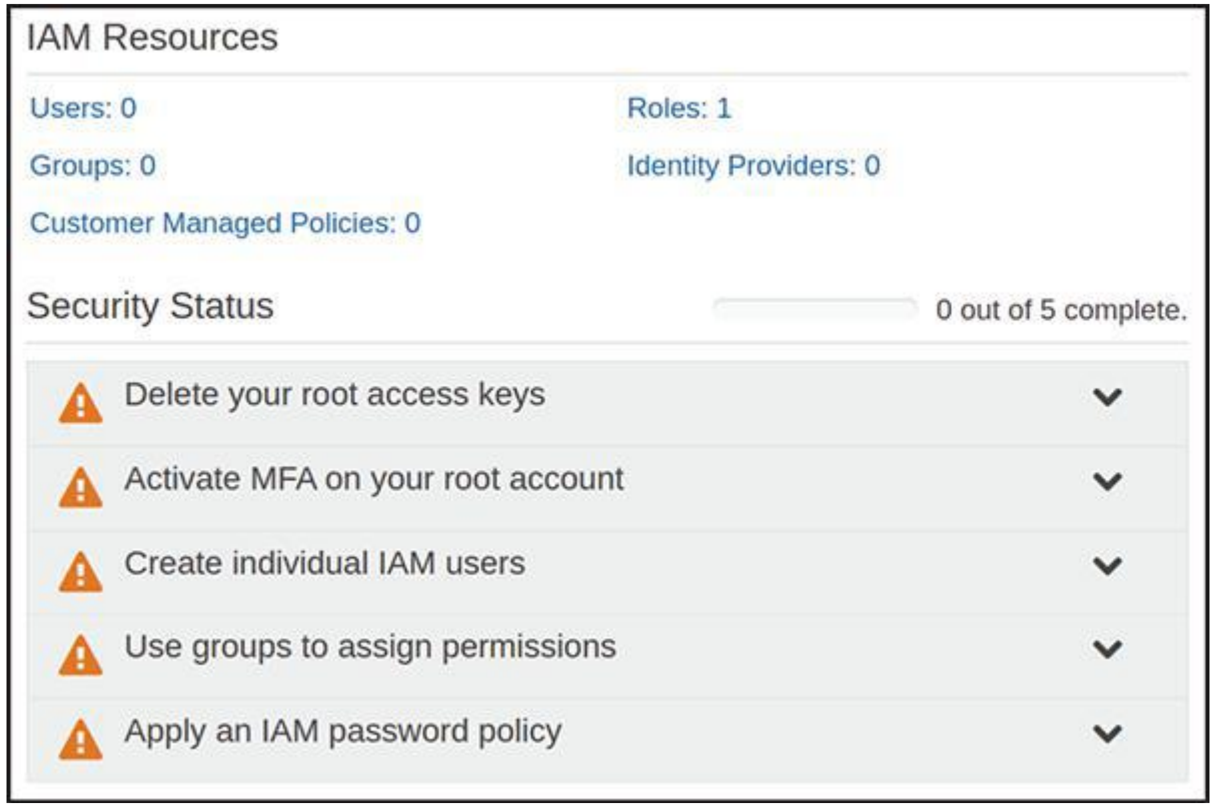
– Identities are controlled by attaching policies that precisely define the way they'll be able to interact with all the resources in your AWS account. You can attach policies to either principals (identity-based policies) or resources (resource-based policies).

Authentication and Authorization—AWS Identity and Access Management

IAM Identities

The one identity that comes with every new AWS account is the root user. By default, root has full rights over all the services and resources associated with your account. This makes sense since there would otherwise be no way for you to get things done with the services that lie beyond root control. On the other hand, having everything in the hands of this one user makes root an attractive target for hackers: they only need to get the root password or access keys to compromise the entire account.

To reduce your exposure to this vulnerability, AWS suggests that you heavily protect your root account and delegate specific powers for day-to-day operations to other users. Figure 6.1 shows the checklist of Amazon's recommended actions in the Security Status section of a typical account's IAM home page.



IAM Resources

Users: 0
Roles: 1
Groups: 0
Identity Providers: 0
Customer Managed Policies: 0

Security Status — 0 out of 5 complete.

- Delete your root access keys
- Activate MFA on your root account
- Create individual IAM users
- Use groups to assign permissions
- Apply an IAM password policy

Authentication and Authorization—AWS Identity and Access Management

IAM Policies

Your first job should be to understand how policies are used to control the behavior of IAM identities. An IAM policy is a document that identifies one or more *actions* as they relate to one or more AWS *resources*. Finally, the policy document determines the *effect* permitted by the action on the resource. The value of an effect will be either Allow or Deny.

A policy might, for instance, allow (the effect) the creation of buckets (the action) within S3 (the resource). Of course, the way resources and actions are defined will vary according to the particular service you're working with.

IAM provides hundreds of preset policies that can be viewed from the Policies page that's accessed from the IAM Dashboard. You can filter policies using key words to narrow down your search. You can also create your own policy using the tools available through the Create policy page in the Dashboard or by manually crafting your own using JSON-formatted text.

Authentication and Authorization—AWS Identity and Access Management

- *The following example is a JSON AdministratorAccess policy document from the IAM Dashboard interface. Note how it allows the identity holding the policy to perform any (*) action on any (*) resource in your account.*

```
{ "Version": "2012-10-17",
 "Statement": [
 { "Effect": "Allow",
"Action": "*",
"Resource": "*"
}
]
}
```

Authentication and Authorization—AWS Identity and Access Management

Any action that's not explicitly allowed by a policy will be denied. But wait, if all actions are implicitly denied, why would you ever need to explicitly invoke “deny” within a policy? Such policies could be useful in cases where a user requires access to most—but not all—resources within a domain. “Deny” can single out exactly those resources that should remain off-limits.

When you associate a policy document with an IAM identity, that identity will now be bound to the policy's powers and limitations. A single IAM policy can be associated with any number of identities, and a single identity can have as many as 10 managed policies (each no greater than 6,144 characters) attached to it.

This raises the potential for trouble: what happens if two policies are associated with a single identity conflict? What if, for instance, one policy permits a user to create new S3 buckets and the second forbids it? AWS always resolves such conflicts by denying the action in question.

You should also bear in mind that, in general, an explicit deny effect will always overrule an allow.

Authentication and Authorization—AWS Identity and Access Management

User and Root Accounts

- The best way to protect your root account is to lock it down by doing the following
- Delete any access keys associated with root
- Assign a long and complex password and store it in a secure password vault.
- Enable multifactor authentication (MFA) for the root account.
- Wherever possible, don't use root to perform administration operations

Before all that, however, you must create at least one new user and assign it enough authority to get the necessary work done. Typically, that will involve giving your main admin user the AdministratorAccess policy, through which you'll be able to use that user to create other users, groups, and roles—each with just enough power to perform its specific task.

Note : You may be forgiven for wondering why giving a user the AdministratorAccess policy is any safer than leaving your root account in active service. After all, both seem to have complete control over all your resources, right? Wrong, because there are some powers that even an AdministratorAccess holder doesn't have, including the ability to create or delete account-wide budgets and enable MFA Delete on an S3 bucket.

Authentication and Authorization—AWS Identity and Access Management

When you create a new IAM user, you'll have the option of applying a password policy. This policy can be set to enforce the minimum lengths and complexity along with the maximum lapsed time between password resets. Getting your team members to use only higher-quality passwords is an important security precaution.

Users can open the My Security Credentials page (accessed via the pull-down menu beneath the user's name in the console) to manage their security settings. As you can see in Figure 6.2, that page includes sections for the following

- » Updating a password
- » Activating or managing MFA
- » Generating or deleting access keys for managing your AWS resources through the AWS CLI or programming SDKs
- » Generating key pairs for authenticating signed URLs for your Amazon CloudFront distributions
- » Generating X.509 certificates to encrypt Simple Object Access Protocol (SOAP) requests to those AWS services that allow it

Note SOAP requests to S3 and Amazon Mechanical Turk are an exception to this rule, as they use regular access keys rather than X.509 certificates.

Retrieving your 12-digit AWS Account ID and, for use with legacy S3 ACLs, your canonical user ID

Authentication and Authorization—AWS Identity and Access Management

Access Keys

Access keys provide authentication for programmatic or CLI-based access. Rather than having to find a way for your application to input a traditional username and password, you can make your local environment aware of the access key ID and secret access key. Using that information, your application or command can pass authentication data along with each request.

You learned about access keys in action in Chapter 2, "Amazon Elastic Compute Cloud and Amazon Elastic Block Store." You should also become familiar with the AWS access key lifecycle and how it should be managed.

Deactivating Unused Keys

Each active access key is a potential source of account vulnerability. Audit the keys associated with each of your users from time to time, and if you can confirm that any of these keys are not currently being used, deactivate them. If you have no plans to use them in the future, delete them altogether.

Key Rotation

Best practices require that you regularly retire older access keys because the longer a key has been in use, the greater the chance that it's been compromised. Therefore, it makes good sense to set a limit—perhaps 30 days—beyond which keys must be deleted and replaced by new ones.

Key rotation is automated for IAM roles used by EC2 resources to access other AWS services. But if your keys are designated for your own applications, things can be more complicated. It's a good idea to follow this protocol:

Authentication and Authorization—AWS Identity and Access Management

- Generate a new access key for each of your users. Users can also be given the ability to manage their own keys.
- Update your application settings to point to the new keys.
- Deactivate (but don't delete) the old keys.
- Monitor your applications for a few days to make sure all the updates were successful. The CLI command

```
aws iam get-access-key-last-used --access-key-id ABCDEFGHIJKLMNOP
```

- can be used to determine whether any applications are still using an old key
- When you're sure you have everything, delete the old keys

Key rotation can be enforced by including rotation in the password policy associated with your IAM user accounts.

Authentication and Authorization—AWS Identity and Access Management

Groups

Creating individual users with just the permissions they'll need to perform their duties can be a secure and efficient way to manage AWS operations. But as your account becomes busier and more complex, it can also become a nightmare to manage.

Say, for instance, you've given some level of access to a half a dozen admins and a similar number of developers. Manually assigning appropriate access policies to each user can be time-consuming. But just imagine how much fun you'll have if you need to *update* permissions across the entire team—perhaps applying one set of changes for just the developers and another for the admins.

The solution for such scenarios is to create a separate IAM group for each class of user and then associate each of your users with the group that fits their job description. You could create one group for developers, another for admins, and a third for your design team.

Now, whenever you need to make changes to a particular class's access profile, rather than having to edit each user account, you just update the appropriate group. If you're adding an Elastic Beanstalk workload, you can open it up for the entire developers' group with a single action. If you're no longer running your video content through Amazon Elastic Transcoder, you can remove access from the design group.

Authentication and Authorization—AWS Identity and Access Management

Roles

An IAM role is a temporary identity that a user or service seeking access to your account resources can request. This kind of authorization can solve a lot of logistical problems.

You might have users who will occasionally need to shut down and restart EC2 instances after an update. To prevent accidental shutdowns, however, you'd rather they didn't normally have that power. Much the way sudo

works on a Linux or macOS machine—or the runas command syntax on Windows—you can allow your user to assume an authorizing role for only as long as it's needed

You might also want to temporarily give users from another AWS account—or users who sign in using a federated authentication service—access to resources on this account. An IAM role (which by default expires after 12 hours) is often the best mechanism for making this work.

You create a new role by defining the *trusted entity* you want given access. There are four categories of trusted entity: an AWS service; another AWS account (identified by its account ID); a web identity who authenticates using a login with Amazon, Amazon Cognito, Facebook, or Google; and SAML 2.0 federation with a SAML provider you define separately.

Once your entity is defined, you give it permissions by creating and attaching your own policy document or assigning one or more preset IAM policies. When a trusted entity assumes its new role, AWS issues it a time-limited security token using the AWS Security Token Service (STS).

Authentication and Authorization—AWS Identity and Access Management

Authentication Tools

AWS provides a wide range of tools to meet as many user and resource management needs as possible. The IAM functionality that you've already seen is only part of the story. The Amazon Cognito, AWS Managed Microsoft AD, and AWS Single Sign-On services are for handling user authentication, while AWS Key Management Service (KMS), AWS Secrets Manager, and AWS CloudHSM simplify the administration of encryption keys and authentication secrets.

Amazon Cognito

Cognito provides mobile and web app developers with two important functions.

- Through Cognito's *user pools*, you can add user sign-up and sign-in to your applications.
- Through Cognito's *identity pools*, you can give your application users temporary, controlled access to other services in your AWS account.

Building a new user pool involves defining how you want your users to identify themselves when they sign up (attributes such as address or birth date) and sign in (username or email address). You can also set minimum requirements for password complexity, multifactor authentication, and email verification.

When you set up an identity pool, you define the pool from which your users can come (using Cognito, AWS, federated, or even unauthenticated identities). You then create and assign an IAM role to the pool. Once your pool is live, any user whose identity matches your definition will have access to the resources specified in the role.

Authentication and Authorization—AWS Identity and Access Management

AWS Managed Microsoft AD

Managed Microsoft AD is actually accessed through the AWS Directory Service, as are a number of directory management tools like Amazon Cloud Directory and Cognito. (Cloud Directory is a way to store and leverage hierarchical data like lists of an organization's users or hardware assets.) What the Directory Service tools all share in common is the ability to handle large stores of data and integrate them into AWS operations.

Technically, Managed Microsoft AD is called *AWS Directory Service for Microsoft Active Directory*. But whichever way you refer to it, the goal is to have Active Directory control the way Microsoft SharePoint, .NET, and SQL Server–based workloads running in your VPC connect to your AWS resources. It's also possible to connect your AWS services to an on-premises Microsoft Active Directory using AD Connector.

Managed Microsoft AD domain controllers run in two VPC availability zones. As a managed service, AWS automatically takes care of all necessary infrastructure administration, including data replication and software updates.

Authentication and Authorization—AWS Identity and Access Management

AWS Single Sign-On

Single sign-on (SSO) allows you to provide users with streamlined authentication and authorization through an existing Microsoft Active Directory configured within AWS Directory Service. The service works across multiple AWS accounts within AWS Organizations. SSO also supports access to popular applications such as Salesforce, Box, and Office 365 in addition to custom apps that support Security Assertion Markup Language (SAML) 2.0.

AWS Organizations, by the way, is a service that can manage policy-based controls across multiple AWS accounts. Companies with more than one AWS account can use AWS Organizations to unify and integrate the way their assets are exposed and consumed no matter how distributed they might be.

Authentication and Authorization—AWS Identity and Access Management

AWS Key Management Service

As you saw regarding encryption keys for EBS volumes in Chapter 2 and relating to both server-side and client-side encryption for S3 buckets in Chapter 3, "Amazon Simple Storage Service and Amazon Glacier Storage," KMS deeply integrates with AWS services to create and manage your encryption keys

The value of KMS lies in how it provides fully managed and centralized control over your system-wide encryption. The service lets you create, track, rotate, and delete the keys that you'll use to protect your data. For regulatory compliance purposes, KMS is integrated with AWS CloudTrail, which records all key-related events.

Key creation and administration happens through the console, AWS CLI, or SDKs. Key administration powers can be assigned to individual IAM users, groups, or roles.

AWS Secrets Manager

You already know that you can manage identity authentication to AWS services using IAM roles. However, roles won't help you securely pass credentials—referred to as *secrets* in this context—to third-party services or databases

Instead, the passwords and third-party API keys for many of the resources your applications might need can be handled by the AWS Secrets Manager. Rather than having to hard-code secrets into your code and then having to regularly update them when they change, with Secrets Manager, you can deliver the most recent credentials to applications on request. The manager will even automatically take care of credential rotation.

Authentication and Authorization—AWS Identity and Access Management

AWS CloudHSM

CloudHSM (where the HSM stands for "hardware security module") launches virtual compute device clusters to perform cryptographic operations on behalf of your web server infrastructure. One typical goal is to off-load the burden of generating, storing, and managing cryptographic keys from your web servers so their resources can be focused exclusively on serving your applications.

While CloudHSM provides a service that's similar to AWS KMS, according to AWS documentation (https://aws.amazon.com/cloudhsm/faqs/), it is particularly useful for the following:

Keys stored in dedicated, third-party validated hardware security modules under your exclusive control

Federal Information Processing Standards (FIPS) 140-2 compliance

Integration with applications using Public Key Cryptography Standards (PKCS)#11, Java JCE (Java Cryptography Extension), or Microsoft CNG (Cryptography API: Next Generation) interfaces

High-performance in-VPC cryptographic acceleration (bulk crypto)

You activate an HSM cluster by running the CloudHSM client as a daemon on each of your application hosts. The client is configured to fully encrypt communication with the HSM.

Authentication and Authorization—AWS Identity and Access Management

Summary

The IAM root user that's automatically enabled on a new AWS account should ideally be locked down and not used for day-to-day account operations. Instead, you should give individual users the precise permissions they'll need to perform their jobs.

All user accounts should be protected by strong passwords, multifactor authentication, and the use of encryption certificates and access keys for resource access.

Once authenticated, a user can be authorized to access a defined set of AWS resources using IAM policies. It's a good practice to associate users with overlapping access needs into IAM groups, where their permissions can be centrally and easily updated. Users can also be assigned temporary IAM roles to give them the access they need, when they need it.

Access keys should be regularly audited to ensure that unused keys are deleted and active keys are rotated at set intervals.

Identities (including users, groups, and roles) can be authenticated using a number of AWS services, including Cognito, Managed Microsoft AD, and single sign-on. Authentication secrets are managed by services such as AWS Key Management Service (KMS), AWS Secrets Manager, and AWS CloudHSM.

Authentication and Authorization—AWS Identity and Access Management

Exam Essentials

Understand how to work with IAM policies. You can build your own custom IAM policies or select preset policies and apply them to identities to control their access to AWS resources.

Understand how to protect your AWS account's root user. You should lock down your root user and instead delegate day-to-day tasks to specially defined users.

Understand how to effectively and securely manage user access. This includes the efficient use of IAM groups and roles and appropriately applying both preset and custom IAM policies.

Understand how to optimize account access security. You can use IAM administration tools to enforce the use of strong passwords and MFA, along with properly managed (rotated) access keys and tokens.

Be familiar with the various AWS authentication and integration tools. Cognito lets you manage your application's users, and Managed Microsoft AD applies Active Directory domains to compatible applications running in your VPC. Both permit federated identities (where users using external authentication services can be authenticated for AWS resources) through external providers.

CloudTrail, CloudWatch, and AWS Config

Introduction

– CloudTrail, CloudWatch, and AWS Config are three AWS services that can help you ensure the continuing health, performance, and security of your AWS resources and applications. These services collectively help you keep an eye on your AWS environment by performing the following operational tasks:.

– **Tracking Performance** Understanding how your AWS resources are performing can tell you if they're powerful enough to handle the load you're throwing at them or if you need to scale up or out. By tracking performance over time, you can identify spikes in usage and determine whether those spikes are temporarily exhausting your resources. For example, if an EC2 instance maxes out its CPU utilization during times of high usage, you may need to upgrade to a larger instance type or add more instances.

– **Detecting Application Problems** Some application problems may be latent or hidden and go unreported by users. Checking application logs for warnings or errors can alert you to latent problems early on. For example, the words *exception* or *warning* in a log may indicate a data corruption, a timeout, or other issue that needs to be investigated before it results in a catastrophic failure.

CloudTrail, CloudWatch, and AWS Config

Introduction

– **Detecting Security Problems** Users having excessive permissions—or worse, accessing resources they're not supposed to—pose a security risk. Tracking user activities and auditing user permissions can help you reduce your risk and improve your security posture.

– **Logging Events** Maintaining a log of every single action that occurs against your AWS resources can be invaluable in troubleshooting and security investigations. Problems and breaches are inevitable and sometimes go undiscovered until months later. Having a detailed record of who did what and when can help you understand the cause, impact, and scope of an issue. Just as importantly, you can use this information to prevent the same problem in the future.

– **Maintaining an Inventory of AWS Resources** Understanding your existing resources, how they're configured, and their relationships and dependencies can help you understand how proposed changes will impact your current environment. Maintaining an up-to-date inventory can also help you ensure compliance with baselines your organization has adopted. Tracking your resource configurations over time makes it easy to satisfy auditing requirements that demand documentation on how a resource was configured at a point in time in the past.

CloudTrail, CloudWatch, and AWS Config

Introduction

– CloudTrail, CloudWatch, and AWS Config are separate services that can be configured independently. They can also work together to provide a comprehensive monitoring solution for your AWS resources, applications, and even on-premises servers.

  - CloudTrail keeps detailed logs of every read or write action that occurs against your AWS resources, giving you a trail that includes what happened, who did it, when, and even their IP address.
  - CloudWatch collects numeric performance metrics from AWS and non-AWS resources such as on-premises servers. It also collects and stores log files from these resources and lets you search them easily, and it provides alarms that can send you a notification or take an action when a metric crosses a threshold.
  - AWS Config tracks how your AWS resources are configured and how they change over time. You can view how your resources are related to one another and how they were configured at any time in the past. You can also compare your resource configurations against a baseline that you define and have AWS Config alert you when a resource falls out of compliance.

CloudTrail, CloudWatch, and AWS Config

CloudTrail

- An *event* is a record of an action that a principal performs against an AWS resource. CloudTrail logs read and write actions against AWS services in your account, giving you a detailed record including the action, the resource affected and its region, who performed the action, and when.
- CloudTrail logs both API and non-API actions. Non-API actions include logging into the management console. API actions include launching an instance, creating a bucket in S3, and creating a virtual private cloud (VPC). These are API events regardless of whether they're performed in the AWS management console, with the AWS Command Line Interface, with an AWS SDK, or by another AWS service. CloudTrail classifies events into *management events and data events*.

CloudTrail, CloudWatch, and AWS Config

Management Events

- Management events include operations that a principal executes (or attempts to execute) against an AWS resource. AWS also calls management events *control plane operations*.
- Management events are further grouped into write-only and read-only events. Write-only events include API operations that modify or might modify resources. For example, the Run Instance API operation may create a new EC2 instance and would be logged, regardless of whether the call was successful. Write-only events also include logging into the management console as the root or an IAM user. CloudTrail does not log unsuccessful root logins. Read-only events include API operations that read resources but can't make changes, such as Describe Instances API operation that returns a list of EC2 instances.

CloudTrail, CloudWatch, and AWS Config

Data Events

- Data events track two types of data plane operations that tend to be high volume: S3 object-level activity and Lambda function executions. For S3 object-level operations, CloudTrail distinguishes read-only and write-only events. GetObject is a read-only event, while DeleteObject and PutObject are write-only events.

Event History

- By default, CloudTrail logs 90 days of management events and stores them in a viewable, searchable, and downloadable database called the *event history*. The event history does not include data events.
- CloudTrail creates a separate event history for each region containing only the activities that occurred in that region. But events for global services such as IAM and Route 53 are included in the event history of every region.

CloudTrail, CloudWatch, and AWS Config

Trails

- If you want to store more than 90 days of event history or if you want to customize the types of events CloudTrail logs—for example, by excluding specific services or actions, or including data events—you can create a *trail*
- A trail is a configuration that records specified events and delivers them as CloudTrail log files to an S3 bucket of your choice. A log file contains one or more log entries in JavaScript Object Notation (JSON) format. A log entry represents a single action against a resource and includes detailed information about the action including, but not limited to, the following:
- Event Time The date and time of the action, given in universal coordinated time (UTC). Log entries in a log file are sorted by timestamp, but events with the same timestamp are not necessarily in the order in which the events occurred.
- User Identity Detailed information about the principal that initiated the request. This may include the type of principal (e.g., IAM role or user), its Amazon resource name (ARN), and IAM username.

CloudTrail, CloudWatch, and AWS Config

- eventSource The global endpoint of the service against which the action was taken
- EventName The name of the API operation
- Aws Region The region the resource is located in. For global services, this is always us-east-1.
- Source IPAddress The IP address of the requester

Creating a Trail

- You can choose to log events from a single region or all regions. If you apply the trail to all regions, then whenever AWS launches a new region, they'll automatically add it to your trail.
- You can create up to five trails for a single region. A trail that applies to all regions will count against this limit in each region. For example, if you create a trail in us-east-1 and then create another trail that applies to all regions, CloudTrail will consider you to have two trails in the us-east-1 region.
- After creating a trail, it can take up to 15 minutes between the time CloudTrail logs an event and the time it writes a log file to the S3 bucket.

CloudTrail, CloudWatch, and AWS Config

Logging Management and Data Events

- When you create a trail, you can choose whether to log management events, data events, or both. If you log management events, you must choose whether to log read-only events, write-only events, or both. This allows you to log read-only and write-only events to separate trails
- If you create a trail using the web console and log management events, the trail will automatically log global service events also. These events are logged as occurring in the us-east-1 region. This means that if you create multiple trails using the web console, global events will be logged to each trail. To avoid these duplicate events, you can disable logging global service events on an existing trail using the AWS CLI command aws cloudtrail update-trail --name mytrail --no-include-global-service-events. Alternately, if a trail is configured to log to all regions and you reconfigure it to log only to a single region, CloudTrail will disable global event logging for that trail. Another option is to forego the web console and create your trails using the AWS CLI, including the --no-include-global-service-events flag

CloudTrail, CloudWatch, and AWS Config

- If you create a single-region trail to log data events, you can log only Lambda functions that exist in that region. If you create a trail that applies to all regions, you can log Lambda functions in any region. You're limited to selecting a total of 250 individual objects per trail, including Lambda functions and S3 buckets and prefixes. If you have more than 250 objects to log, you can instruct CloudTrail to log all Lambda functions or all S3 buckets.
- Note Don't log data events on the bucket which is storing your CloudTrail logs. Doing so would create an infinite loop!

CloudTrail, CloudWatch, and AWS Config

Log File Integrity Validation

- CloudTrail provides a means to ensure that no log files were modified or deleted after creation. During quiet periods of no activity, it also gives you assurance that no log files were delivered, as opposed to being delivered and then maliciously deleted. This is useful in forensic investigations where someone with access to the S3 bucket may have tampered with the log file. For example, when an attacker hacks into a system, it's common for them to delete or modify log files to cover their tracks.
- With log file integrity validation enabled, every time CloudTrail delivers a log file to the S3 bucket, it calculates a cryptographic hash of the file. This hash is a unique value derived from the contents of the log file itself. If even one byte of the log file changes, the entire hash changes. Hashes make it easy to detect when a file has been modified.

CloudTrail, CloudWatch, and AWS Config

Log File Integrity Validation

– Every hour, CloudTrail creates a separate file called a *digest file* that contains the cryptographic hashes of all log files delivered within the last hour. CloudTrail places this file in the same bucket as the log files but in a separate folder. This allows you to set different permissions on the folder containing the digest file to protect it from deletion. CloudTrail also cryptographically signs the digest file using a private key that varies by region and places the signature in the file's S3 object metadata.

– Each digest file also contains a hash of the previous digest file, if it exists. If there are no events to log during an hour long period, CloudTrail still creates a digest file. This lets you assert that no log files were delivered during the period.

Log File Integrity Validation

- You can validate the integrity of CloudTrail log and digest files by using the AWS CLI. You must specify the ARN of the trail and a start time. The AWS CLI will validate all log files from the starting time to the present. For example, to validate all log files written from January 1, 2018, to the present, you'd issue the following command: aws cloudtrail validate-logs --trail-arn arn:aws:cloudtrail:us-east-1:account-id:trail/benpiper-trail --start-time 2018-01-01T00:00:00Z.
- Note : By default, log and digest files are encrypted using Amazon server-side encryption with Amazon S3-managed encryption keys (SSE-S3). For additional security, you can choose instead to use server-side encryption with AWS KMS-managed keys (SSE-KMS) for log files. The customer master key (CMK) you use must be in the same region as the bucket. Digest files are always encrypted with SSE-S3.

CloudTrail, CloudWatch, and AWS Config

CloudWatch

– CloudWatch functions as a metric repository that lets you collect, retrieve, and graph numeric performance metrics from AWS and non-AWS resources. All AWS resources automatically send their metrics to CloudWatch. These metrics include EC2 instance CPU utilization, EBS volume read and write IOPS, S3 bucket sizes, and DynamoDB consumed read and write capacity units. Optionally, you can send custom metrics to CloudWatch from your applications and on-premises servers. CloudWatch Alarms can send you a notification or take an action based on the value of those metrics. CloudWatch Logs lets you collect, store, view, and search logs from AWS and non-AWS sources. You can also extract custom metrics from logs, such as the number of errors logged by an application or the number of bytes served by a web server.

CloudWatch Metrics

- CloudWatch organizes metrics into *namespaces*. Metrics from AWS services are stored in AWS namespaces and use the format AWS/service to allow for easy classification of metrics. For example, AWS/EC2 is the namespace for metrics from EC2, and AWS/S3 is the namespace for metrics from S3.
- You can think of a namespace as a container for metrics. Namespaces help prevent confusion of metrics with similar names. For example, CloudWatch stores the WriteOps metric from the Relational Database Service (RDS) in the AWS/RDS namespace, while the EBS metric VolumeWriteOps goes in the AWS/EBS namespace. You can create custom namespaces for custom metrics. For example, you may store metrics from an Apache web server under the custom namespace Apache. Metrics exist only in the region in which they were created.

CloudWatch Metrics

A metric functions as a variable and contains a time-ordered set of data points. Each data point contains a timestamp, a value, and optionally a unit of measure. Each metric is uniquely defined by a namespace, a name, and optionally a dimension. A dimension is a name-value pair that distinguishes metrics with the same name and namespace from one another. For example, if you have multiple EC2 instances, CloudWatch creates a CPUUtilization metric in the AWS/EC2 namespace for each instance. To uniquely identify each metric, AWS assigns it a dimension named InstanceId with the value of the instance's resource identifier.

CloudTrail, CloudWatch, and AWS Config

Basic and Detailed Monitoring

- How frequently an AWS service sends metrics to CloudWatch depends on the monitoring type the service uses. Most services support basic monitoring, and some support *basic monitoring* and *detailed monitoring*.
- Basic monitoring sends metrics to CloudWatch every five minutes. EC2 provides basic monitoring by default. EBS uses basic monitoring for gp2 volumes.
- EC2 collects metrics every minute but sends only the five-minute average to CloudWatch. How EC2 sends data points to CloudWatch depends on the hypervisor. For instances using the Xen hypervisor, EC2 publishes metrics at the end of the five-minute interval. For example, between 13:00 and 13:05 an EC2 instance has the following CPU Utlization metric values measured in percent: 25, 50, 75, 80, and 10. The average CPU Utilization over the five-minute interval is 48. Therefore, EC2 sends the CPU utilization metric to CloudWatch with a timestamp of 13:00 and a value of 48.

CloudTrail, CloudWatch, and AWS Config

Basic and Detailed Monitoring

– For instances using the Nitro hypervisor, EC2 sends a data point every minute during a five-minute period, but a data point is a rolling average. For example, at 13:00, EC2 records a data point for CPU utilization metric with a value of 25. EC2 sends this data point to CloudWatch with a timestamp of 13:00. At 13:01, EC2 records another data point with a value of 50. It averages this new data point with the previous one to get a value of 37.5. It then sends this new data point to CloudWatch, but with a timestamp of 13:00. This process continues for the rest of the five-minute interval.

– Services that use detailed monitoring publish metrics to CloudWatch every minute. More than 70 services support detailed monitoring including EC2, EBS, RDS, DynamoDB, ECS, and Lambda. EBS defaults to detailed monitoring for io1 volumes.

CloudTrail, CloudWatch, and AWS Config

Regular and High-Resolution Metrics

- The metrics generated by AWS services have a timestamp resolution of no less than one minute. For example, a measurement of CPU Utilization taken at 14:00:28 would have a timestamp of 14:00. These are called *regular-resolution* metrics. For some AWS services, such as EBS, CloudWatch stores metrics at a five-minute resolution. For example, if EBS delivers a VolumeWriteBytes metric at 21:34, CloudWatch would record that metric with a timestamp of 21:30.
- CloudWatch can store custom metrics with up to one second resolution. Metrics with a resolution of less than one minute are *high-resolution metrics*. You can create your own custom metrics using the PutMetricData API operation. When publishing a custom metric, you can specify the timestamp to be up to two weeks in the past or up to two hours into the future. If you don't specify a timestamp, CloudWatch creates one based on the time it received the metric in UTC.

Expiration

- You can't delete metrics in CloudWatch. Metrics expire automatically, and when a metric expires depends on its resolution. Over time, CloudWatch aggregates higher-resolution metrics into lower-resolution metrics.
- A high-resolution metric is stored for three hours. After this, all the data points from each minute-long period are aggregated into a single data point at one-minute resolution. The high-resolution data points simultaneously expire and are deleted. After 15 days, five data points stored at one-minute resolution are aggregated into a single data point stored at five-minute resolution. These metrics are retained for 63 days. At the end of this retention period, 12 data points from each metric are aggregated into a single 1-hour resolution metric and retained for 15 months. After this, the metrics are deleted.
- To understand how this works, consider a VolumeWriteBytes metric stored at five-minute resolution. CloudWatch will store the metric at this resolution for 63 days, after which it will convert the data points to one-hour resolution. After 15 months, CloudWatch will delete those data points permanently.

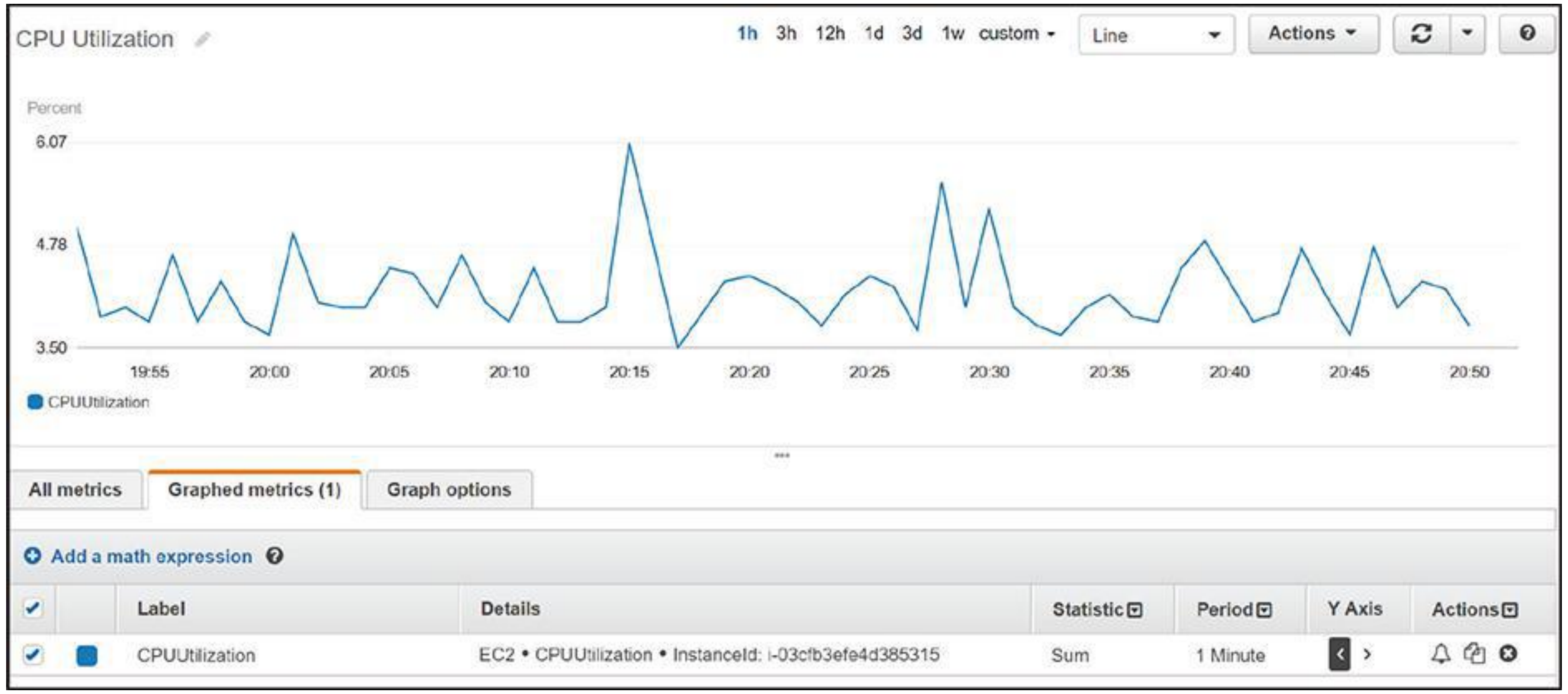
Graphing Metrics

- CloudWatch lets you visualize your metrics by graphing data points over time. This is useful for showing trends and changes over time, such as spikes in usage.
- CloudWatch can perform statistical analysis on data points over a period of time and graph the results as a time series. You can choose from the following *statistics*:
- **Sum** The total of all data points in a period
- **Minimum** The lowest data point in a period
- **Maximum** The highest data point in a period
- **Average** The average of all data points in a period
- **Sample count** The number of data points in a period
- **Percentile** The data point of the specified percentile. You can specify a percentile of up to two decimal places. For example, 99.44 would yield the lowest data point that's higher than 99.44 percent of the data points in the period. You must specify a percentile statistic in the format p99.44

CloudTrail, CloudWatch, and AWS Config

Graphing Metrics

- To graph a metric, you must specify the metric, the statistic, and the period. The period can be from 1 second to 30 days, and the default is 60 seconds.
- If you want CloudWatch to graph a metric as is, use the Sum statistic and set the period equal to the metric's resolution. For example, if you're using detailed monitoring to record the CPU Utilization metric from an EC2 instance, that metric will be stored at one-minute resolution. Therefore, you would graph the CPU Utilization metric over a period of one minute using the Sum statistic. To get an idea of how this would look



✓		Label	Details	Statistic	Period	Y Axis	Actions
✓	🟦	CPUUtilization	EC2 • CPUUtilization • InstanceId: i-03cfb3efe4d385315	Sum	1 Minute	‹ ›	

CloudTrail, CloudWatch, and AWS Config

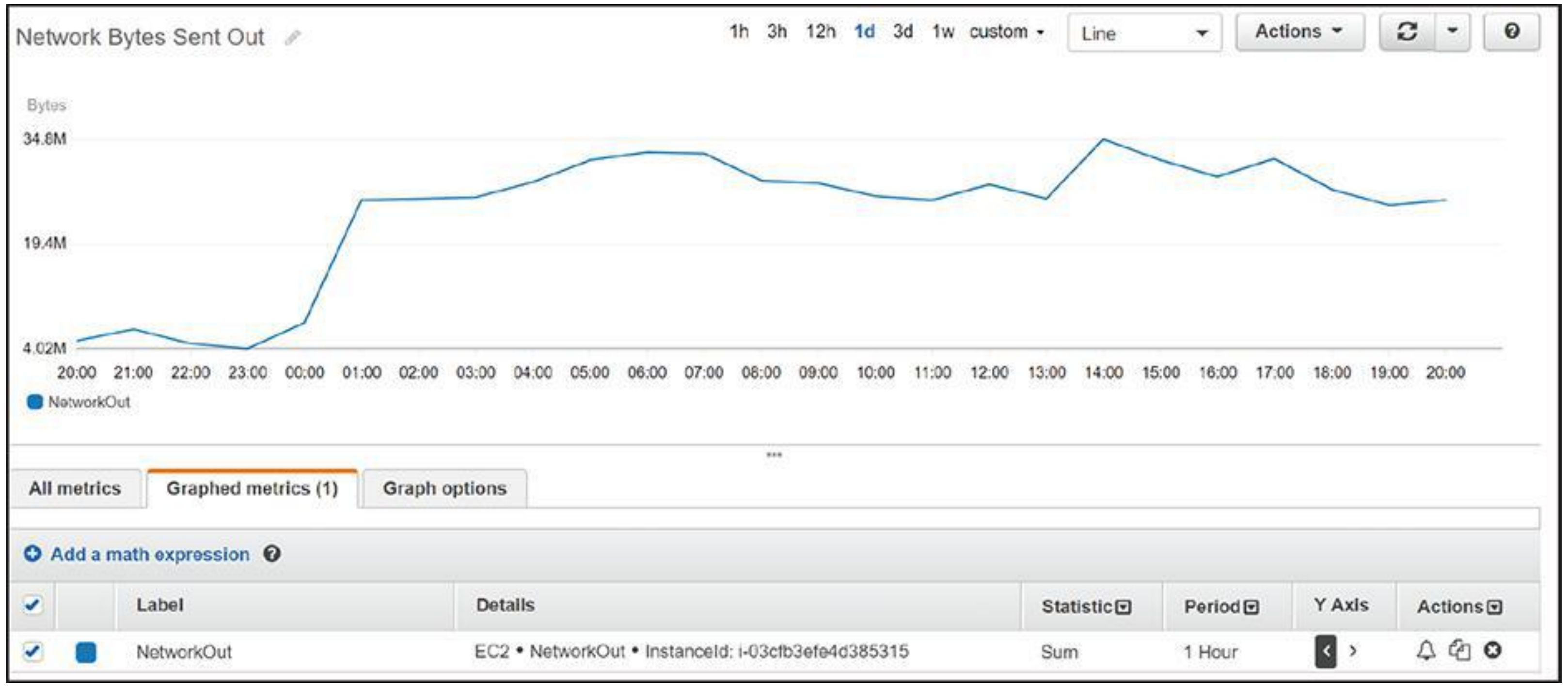
Graphing Metrics

- Note that the statistic applies only to data points within a period. In the preceding example, the Sum statistic adds all data points within a one-minute period. Because CloudWatch stores the metric at one-minute resolution, there is only one data point per minute. Hence, the resulting graph shows each individual metric data point
- The *time range* in the preceding graph is set to one hour, but you can choose a range of time between 1 minute and 15 months. Choosing a different range doesn't change the time series, but only how it's displayed.
- Which statistic you should choose depends on the metric and what you're trying to understand about the data. CPU utilization fluctuates and is measured as a percentage, so it doesn't make much sense to graph the sum of CPU utilization over, say, a 15 minute period. It does, however, make sense to take the average CPU utilization over the same timeframe. Hence, if you were trying to understand long-term patterns of CPU utilization, you may use the Average statistic over a 15-minute period

CloudTrail, CloudWatch, and AWS Config

Graphing Metrics

– The NetworkOut metric in the AWS/EC2 namespace measures the number of bytes sent by an instance during the collection interval. To understand peak hours for network utilization, you may graph the Sum statistic over a one-hour period and set the time range to one day, as shown



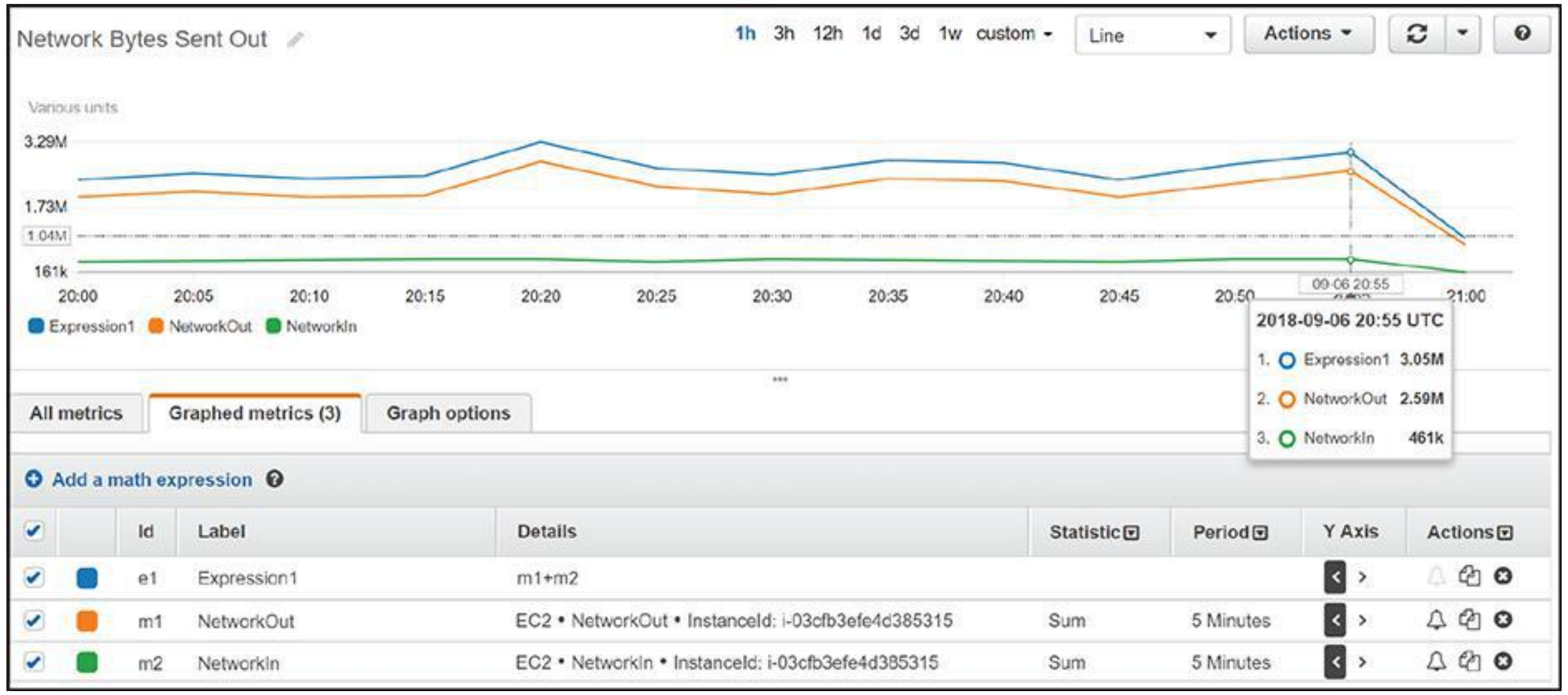
		Label	Details	Statistic	Period	Y Axis	Actions
✓	🟦	NetworkOut	EC2 • NetworkOut • InstanceId: i-03cfb3efe4d385315	Sum	1 Hour	< >	

– The Details column shows information that uniquely identifies the metric: the namespace, metric name, and the metric dimension, which in this case is InstanceID

CloudTrail, CloudWatch, and AWS Config

Metric Math

– CloudWatch lets you perform various mathematical functions against metrics and graph them as a new time series. This is useful for when you need to combine multiple metrics into a single time series by using arithmetic functions, which include addition, subtraction, multiplication, division, and exponentiation. For example, you might divide the AWS/ Lambda Invocations metrics by the Errors metric to get an error rate. Complete Exercise 7.2 to create a CloudWatch graph using *metric math*.



2018-09-06 20:55 UTC
1. Expression1 3.05M
2. NetworkOut 2.59M
3. NetworkIn 461k

✓		Id	Label	Details	Statistic	Period	Y Axis	Actions
✓	🟦	e1	Expression1	m1+m2			< >	
✓	🟧	m1	NetworkOut	EC2 • NetworkOut • InstanceId: i-03cfb3efe4d385315	Sum	5 Minutes	< >	
✓	🟩	m2	NetworkIn	EC2 • NetworkIn • InstanceId: i-03cfb3efe4d385315	Sum	5 Minutes	< >	

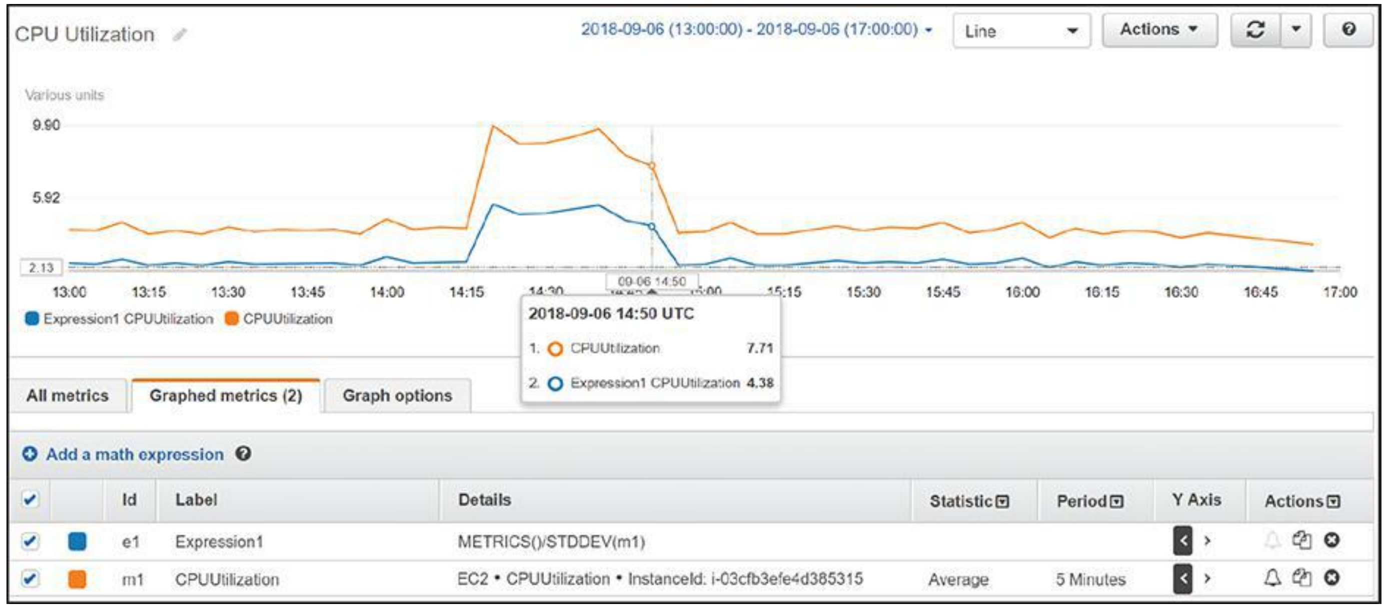
Metric Math

- In addition to arithmetic functions, CloudWatch provides the following statistical functions that you can use in metric math expressions:
  - AVG—Average
  - MAX—Maximum
  - MIN—Minimum
  - STDDEV—Standard deviation
  - SUM—Sum
- Statistical functions return a scalar value, not a time series, so they can't be graphed. You must combine them with the Metrics function, which returns an array of time series of all selected metrics. For instance, referring to step 6 in Exercise 7.2, you could replace the expression M1+M2 with the expression SUM (METRICS()) to achieve the same result.

CloudTrail, CloudWatch, and AWS Config

Metric Math

– As another example, suppose you want to compare the CPU utilization of an instance with the standard deviation. You would first graph the AWS/EC2 metric CPU Utilzation for the instance in question. You'd then add the metric math expression METRICS ()/STDDEV(m1) where m1 is the time series for the CPU Utilizaiton metrics



2018-09-06 14:50 UTC

1. CPUUtilization 7.71
2. Expression1 CPUUtilization 4.38

✓		Id	Label	Details	Statistic	Period	Y Axis	Actions
✓	🟦	e1	Expression1	METRICS()/STDDEV(m1)				
✓	🟧	m1	CPUUtilization	EC2 • CPUUtilization • InstanceId: i-03cfb3efe4d385315	Average	5 Minutes		

– Keep in mind that the function STDDEV(m1) returns a scalar value—the standard deviation of all data points for the CPU Utilization metric. You must therefore use the METRICS function in the numerator to yield a time series that CloudWatch can graph.

CloudTrail, CloudWatch, and AWS Config

CloudWatch Logs

– CloudWatch Logs is a feature of CloudWatch that collects logs from AWS and non-AWS sources, stores them, and lets you search and even extract custom metrics from them. Some common uses for CloudWatch Logs include receiving CloudTrail logs, collecting application logs from an instance, and logging Route 53 DNS queries

Log Streams and Log Groups

– CloudWatch Logs stores *log events* that are records of activity recorded by an application or AWS resource. For CloudWatch Logs to understand a log event, the event must contain a timestamp and a UTF-8 encoded event message.

– CloudWatch Logs stores log events from the same source in a *log stream*. The source may be an application or AWS resource. For example, if you have multiple instances running a web server that generates access logs, each instance would send that log to CloudTrail as a separate stream. You can manually delete log streams, but not individual log events.

CloudTrail, CloudWatch, and AWS Config

Log Streams and Log Groups

- CloudWatch organizes log streams into *log groups*. A stream must exist in only one log group. To organize related log streams, you can place them into the same log group. There's no limit to the number of streams in a group.
- You may define the retention settings for a log group, choosing to keep log events from between 1 day to 10 years or indefinitely, which is the default setting. The retention settings apply to all log streams in a log group. You can manually export a log group to an S3 bucket for archiving.

CloudTrail, CloudWatch, and AWS Config

Metric Filters

- You can use *metric filters* to extract data from logs in a log group to create CloudWatch metrics. You can create a metric filter to track the occurrences of a string at a particular position in a log file. For example, you may want to track the number of 404 Not Found errors that appear in an Apache web server application log. You would create a metric filter to track the number of times the string "404" appears in the HTTP status code section of the log. Every time CloudWatch Logs receives a log event that matches the filter, it increments a custom metric. You might name such a metric HTTP 404 Errors and store it in the custom Apache namespace. Of course, you can then graph this metric in CloudWatch.
- Metric values must be numeric. You can use a metric filter to extract a numeric value, such as the number of bytes transferred in a request, and store that in a metric. But you can't use a metric filter to extract a non-numeric string, such as an IP address, from a log and store it as a metric. You can, however, increment a metric when the metric filter matches a specific string.

CloudTrail, CloudWatch, and AWS Config

Metric Filters

- Metric filters apply to log groups, and you can create a metric filter only after creating the group. Metric filters are not retroactive and will not generate metrics based on log events that CloudWatch recorded before the filter's creation.

CloudWatch Agent

- The CloudWatch Agent collects logs from EC2 instances and on-premises servers running Linux or Windows operating systems. The agent can also collect performance metrics, including metrics that EC2 doesn't natively produce, such as memory utilization. Metrics generated by the agent are custom metrics and are stored in a custom namespace that you specify.

- **Note** The CloudWatch Agent is different than the legacy CloudWatch Logs agent, which sends only logs and not metrics. AWS recommends using the CloudWatch agent

CloudTrail, CloudWatch, and AWS Config

Sending CloudTrail Logs to CloudWatch Logs

- You can configure CloudTrail to send a trail log to a CloudWatch Logs log stream. This lets you search and extract metrics from your trail logs. Remember that CloudTrail generates trail logs in JSON format and stores them in an S3 bucket of your choice, but it doesn't provide a way to search those logs. CloudWatch understands JSON format and makes it easy to search for specific events. For example, to search for failed console logins, you would filter the log stream using the following syntax:
- {$.eventSource = "signin.amazonaws.com" && $.responseElements.ConsoleLogin = "Failure"}
- You must specify a log group and an IAM role to use. CloudTrail can create the role for you and then assume the role to create the log group. CloudTrail then begins automatically delivering new trail logs to CloudWatch Logs. Delivery isn't instant, and it can take a few minutes before trail logs show up in CloudWatch Logs. Complete Exercise 7.3 to configure your existing CloudTrail to deliver trail logs to CloudWatch Logs.

CloudTrail, CloudWatch, and AWS Config

Sending CloudTrail Logs to CloudWatch Logs

– CloudTrail does not send log events larger than 256 KB to CloudWatch Logs. Hence, a single RunInstance call to launch 500 instances would exceed this limit. Therefore, make sure you break up large requests if you want them to be available in CloudWatch Logs.

CloudTrail, CloudWatch, and AWS Config

CloudWatch Alarms

- A *CloudWatch alarm* watches over a single metric and performs an action based on its value over a period of time. The action CloudWatch takes can include things such as sending an email notification, rebooting an instance, or executing an Auto Scaling action.
- To create an alarm, you first define the metric you want CloudWatch to monitor. In much the same way CloudWatch doesn't graph metrics directly but graphs metric statistics over a period, CloudWatch alarms does not directly monitor metrics. Instead, it performs statistical analysis of a metric over time and monitors the result. This is called the *data point to monitor*.

CloudTrail, CloudWatch, and AWS Config

Data Point to Monitor

– Suppose you want to monitor the average of the AWS/EBS VolumeReadOps metric over a 15-minute period. The metric has a resolution of 5 minutes. You would choose Average for the statistic and 15 minutes for the period. Every 15 minutes CloudWatch would take three metric data points—one every 5 minutes—and would average them together to generate a single data point to monitor.

– **Note** You should set the period equal to or greater than the resolution of the metric. If you set a lower period, such as one minute, CloudWatch will look for a data point every minute, but because the metric updates only once every five minutes, it will count four of the five metrics as missing. This will result in the alarm not working properly.

CloudTrail, CloudWatch, and AWS Config

Data Point to Monitor

– If you use a percentile for the statistic, you must also select whether to ignore data points until the alarm collects a statistically significant number of data points. What constitutes statistically significant depends on the percentile. If you set the percentile to .5 (p50) or greater, you must have 10/1(1-percentile) data points to have a statistically significant sample. For instance, if you use the p80 statistic, a statistically significant number of data points would be 10/(1-.8) or 50. If the percentile is less than .5, you need 10/percentile data points to have a statistically significant sample. Supposing you were using the p25 statistic, you'd need 10/(.25) or 40 data points. If you choose to ignore data points before you have a statistically significant sampling, CloudWatch will not evaluate any of them. In other words, your alarm will be effectively disabled.

CloudTrail, CloudWatch, and AWS Config

Threshold

– The threshold is the value the data point to monitor must meet or cross to indicate something is wrong. You define a threshold by specifying a value and a condition. If you want to trigger an alarm when CPU Utilzation meets or exceeds 50 percent, you would set the threshold for that alarm to >= 50. Or if you want to know when CPUCreditBalance falls below 800, you would set the threshold to < 800.

Alarm States

– An alarm can be in one of the three following states at any given time:

ALARM

– The data points to alarm have crossed and remained past a defined threshold for a period of time.

OK

– The data points to alarm have not crossed and remained past a defined threshold for a period of time

INSUFFICIENT_DATA

– The alarm hasn't collected enough data to determine whether the data points to alarm have crossed a defined threshold.

CloudTrail, CloudWatch, and AWS Config

Threshold

- New alarms always start out in an INSUFFICIENT_DATA state It's important to remember that an ALARM state doesn't necessarily indicate a problem, and an state doesn't necessarily indicate the absence of a problem. Alarm states track only whether the data points to alarm have crossed and remained past a threshold for a period of time.

Data Points to Alarm and Evaluation Period

- The period of time a data point to monitor must remain crossing the threshold to trigger an alarm state change depends on the *data points to alarm*. As an example, if the period is five minutes and the data points to alarm is three, then the data points to monitor must cross and remain crossing the threshold for 15 minutes before the alarm goes into an Alarm State
- There are cases where you may want to trigger the alarm if a data point to monitor crosses the threshold periodically but doesn't remain past it. For this, you can set an evaluation period that's equal to or greater than the data points to alarm. Suppose you want to trigger an alarm if the data points to monitor cross the threshold for three out of five data points. You would set the evaluation period to 5. The alarm would trigger if *any* three of the latest five data points exceed the threshold. The exceeding values don't have to be consecutive. This is called an *m out of n alarm*, where *m* is the data point to alarm and *n* is the evaluation period. The evaluation period can't exceed 24 hours

Data Points to Alarm and Evaluation Period

- There are cases where you may want to trigger the alarm if a data point to monitor crosses the threshold periodically but doesn't remain past it. For this, you can set an evaluation period that's equal to or greater than the data points to alarm. Suppose you want to trigger an alarm if the data points to monitor cross the threshold for three out of five data points. You would set the evaluation period to 5. The alarm would trigger if *any* three of the latest five data points exceed the threshold. The exceeding values don't have to be consecutive. This is called an *m out of n alarm*, where *m* is the data point to alarm and *n* is the evaluation period. The evaluation period can't exceed 24 hours
- To give an illustration, let's say you create an alarm with a threshold of >= 40. The data points to alarm is 2, and the evaluation period is 3. Now suppose CloudWatch evaluates the following three consecutive data points: 46, 39, and 41. Two of the three data points exceed the threshold, so the alarm will transition to the ALARM state.

CloudTrail, CloudWatch, and AWS Config

Data Points to Alarm and Evaluation Period

- Following that, CloudWatch evaluates the consecutive data points 45, 30, and 25. Two of the three data points fall below the threshold, so the alarm transitions to an OK state. Notice that CloudWatch must evaluate three data points (the evaluation period) before it changes the alarm state.
- **Note** When monitoring metrics for EC2 instances using the Nitro hypervisor, make sure the evaluation period is at least 2. Triggering an alarm based on a single data point may cause result in false positives because of the way EC2 delivers Nitro metrics to CloudWatch.

Missing Data

- Missing data can occur during an evaluation period. This may happen if you detach an EBS volume from or stop an instance. CloudWatch offers the following four options for how it evaluates periods with missing data:

CloudTrail, CloudWatch, and AWS Config

Missing Data

As Missing

- This treats the period missing data as if it never happened. For instance, if over four periods the data points are 41, no data, 50, and 25, CloudWatch will remove the period with no data from consideration. If the evaluation period is 3, it will evaluate only three periods with the data points 41, 50, and 25. It will not consider the missing data as occurring in the evaluation period. This is the default setting.

Not Breaching

- Missing data points are treated as not breaching the threshold. Consider a three out of four alarm with a threshold <40, given the same data points: 41, no data, 50, and 25. Even though two values are breaching, the alarm would not trigger because the missing data is assumed to be not breaching.

Breaching

- CloudWatch treats missing data as breaching the threshold. Using the preceding illustration, the alarm would trigger, as three of the four values would exceed the threshold.

Ignore

- The alarm doesn't change state until it receives the number of consecutive data points specified in the data points to alarm setting.

CloudTrail, CloudWatch, and AWS Config

Actions

- You can configure an alarm to take an action when it transitions to a given state. You're not limited to just when an alarm goes into an ALARM state. You can also have CloudWatch take an action when the alarm transitions to an OK state. This is useful if you want to receive a notification when CPU utilization is abnormally high and another when it returns to normal. You can also trigger an action when an alarm monitoring an instance goes into an INSUFFICIENT_DATA state, which may occur when the instance shuts down. You can choose from the following actions:
- **Notification Using Simple Notification Service** The Simple Notification Service (SNS) uses communication channels called *topics*. A topic allows a sender or *publisher* to send a notification to one or more recipients called *subscribers*.
- A subscriber consists of a protocol and an endpoint. The protocol can be HTTP, HTTPS, Simple Queue Service (SQS), Lambda, a mobile push notification, email, email-JSON, or short message service (SMS). The endpoint depends on the protocol. In the case of email or email-JSON, the endpoint would be an email address. In the case of SQS, it would be a queue. The endpoint for HTTP or HTTPS would be a URL.

CloudTrail, CloudWatch, and AWS Config

Actions

– When creating an alarm action, you specify an SNS topic. When the alarm triggers, it sends a notification to the topic. SNS takes care of relaying the notification to the subscribers of the topic.

Auto Scaling Action

– If you're using Auto Scaling, you can create a simple Auto Scaling policy to add or remove instances. The policy must exist before you can select it as an alarm action.

EC2 Action

– You can stop, terminate, reboot, or recover an instance in response to an alarm state change. You might choose to monitor the AWS/EC2 STATUSCHECKFAILED_INSTANCE metric, which returns 1 if there's a problem with the instance, such as memory exhaustion, file system corruption, or incorrect network or startup configuration. Such issues could be corrected by rebooting the instance.

CloudTrail, CloudWatch, and AWS Config

EC2 Action

- You might also monitor the StatusCheckedFailed_System metric, which returns 1 when there's a problem that requires AWS involvement to repair, such as a loss of network connectivity or power, hypervisor problems, or hardware failure. In response to this, the recover action migrates the instance to a new host and restarts it with the same settings. In-memory data is lost during this process.
- EC2 actions are available only if the metric you're monitoring includes the Instance Id as a dimension, as the action you specify will take place against that instance. Using an EC2 action requires a service-linked role named AWSServiceRoleForCloudWatchEvents , which CloudWatch can create for you when you create the alarm.
- Although a single alarm can have multiple actions, it will take those actions only upon transition to a single state that you specify when configuring the alarm. You cannot, for instance, set one action when an alarm transitions to Alarm state and another action when it transitions to an OK state. Instead, you'd have to create two separate alarms.

CloudTrail, CloudWatch, and AWS Config

AWS Config

– AWS Config tracks how your AWS resources are configured at a point in time. Resources are entities that you create and manage using the management console, AWS CLI, or AWS SDKs. Think of AWS Config as a time machine. You can use it to see what a resource configuration looked like at some point in the past versus what it looks like now.

– It can also show you how your resources are related to one another so that you can see how a change in one resource might impact another. For instance, suppose you create an EBS volume, attach it to an instance, and then later detach it. You can use AWS Config to see not only exactly when you created the volume but when it was attached to and detached from the instance.

– CloudTrail logs events, and CloudWatch can alert on events, but only AWS Config gives you a holistic view of your resources and how they were configured at any point in time. AWS Config can help you with the following objectives:

CloudTrail, CloudWatch, and AWS Config

AWS Config

Security

- AWS Config can notify you whenever a resource configuration changes, alerting you to potential breaches. You can also see what users had which permissions at a given time.

Easy Audit Reports

- You can provide a configuration snapshot report from any point in time.

Troubleshooting

- You can analyze configurations around the time a problem started. AWS Config makes it easy to spot misconfigurations and how a problem in one resource might impact another.

Change Management

- AWS Config lets you see how a potential change to one resource could impact another. For example, if you plan to change a security group, you can use AWS Config to quickly see all instances that use it.

CloudTrail, CloudWatch, and AWS Config

The Configuration Recorder

– The *configuration recorder* is the workhorse of AWS Config. It discovers your existing resources, records how they're configured, monitors for changes, and tracks those changes over time. By default, it monitors all items in the region in which you configure it. It can also monitor the resources of global services such as IAM. If you don't want to monitor all resources, you can select specific resource types to monitor, such as EC2 instances, IAM users, S3 buckets, or DynamoDB tables. You can have only one configuration recorder per region.

Configuration Items

– The configuration recorder generates a *configuration item* for each resource it monitors. The configuration item contains the specific settings for that resource at a point in time, as well as the resource type, its ARN, and when it was created. The configuration item also includes the resource's relationships to other resources. For example, a configuration item for an EBS volume would include the instance ID of the instance it was attached to at the time the item was recorded. AWS Config maintains configuration items for every resource it tracks, even after the resource is deleted. Configuration items are stored internally in AWS Config, and you can't delete them manually.

CloudTrail, CloudWatch, and AWS Config

Configuration History

AWS Config uses configuration items to build a configuration history for each resource. A configuration history is a collection of configuration items for a given resource over time. A configuration history includes details about the resources, such as when it was created, how it was configured at different points in time, and when it was deleted, if applicable. It also includes any related API logged by CloudTrail.

Every six hours in which a change occurs to a resource, AWS Config delivers a configuration history file to an S3 bucket that you specify. The S3 bucket is part of what AWS calls the delivery channel. Files are grouped by resource type. The configuration history for all EC2 instances goes into one file, while the history for EBS volumes goes into another. The files are timestamped and kept in separate folders by date. You can also view configuration history directly from the AWS Config console. You can optionally add an SNS topic to the delivery channel to have AWS Config notify you immediately whenever there's a change to a resource.

CloudTrail, CloudWatch, and AWS Config

Configuration Snapshots

A configuration snapshot is a collection of all configuration items from a given point in time. Think of a configuration snapshot as a configuration backup for all monitored resources in your account.

Using the AWS CLI you can have AWS Config deliver a configuration snapshot to the bucket defined in your delivery channel. By default, the delivery channel is named default, so to deliver a configuration snapshot, you would manually issue the following command:

```
aws configservice deliver-config-snapshot --delivery-channel-name default
```

AWS Config can automatically deliver a configuration snapshot to the delivery channel at regular intervals. You can't set automatic delivery of this in the console but must configure this using the CLI. To do this, you must also specify a JSON file containing at a minimum the following items

CloudTrail, CloudWatch, and AWS Config

Configuration Snapshots

Delivery channel name (default)

S3 bucket name

Delivery frequency of the configuration snapshot

The file may also optionally contain an SNS ARN. Let's refer to the following file as

CloudTrail, CloudWatch, and AWS Config

deliveryChannel.json:

```
{ "name": "default",
 "s3BucketName": "my-config-bucket-us-east-1",
 "snsTopicARN": "arn:aws:sns:us-east-1:account-id:config-topic",
"configSnapshotDeliveryProperties": {
"deliveryFrequency": "TwentyFour_Hours"
}
}
```

The delivery frequency can be every hour or every 24, 12, 6, or 3 hours. To reconfigure the delivery channel according to the settings in the deliveryChannel.json file, you'd issue the following command:

```
aws configservice put-delivery-channel --delivery-channel file://deliveryChannel.json
```

To verify that the configuration change succeeded, issue the following command:

```
aws configservice describe-delivery-channels
```

If the output matches the configuration settings in the file, then the configuration change was successful.

CloudTrail, CloudWatch, and AWS Config

Monitoring Changes

The configuration recorder generates at least one new configuration item every time a resource is created, changed, or deleted. Each new item is added to the configuration history for the resource as well as the configuration history for the account

A change to one resource will trigger a new configuration item for not only the changed resource but also related resources. For instance, removing a rule in a security group causes the configuration recorder to create a new item for the security group and every instance that uses that security group.

Although you can't delete configuration items manually, you can configure AWS Config to keep configuration items between 30 days to 7 years. Seven years is the default. Note that the retention period does not apply to the configuration history and configuration snapshot files AWS Config delivers to S3.

CloudTrail, CloudWatch, and AWS Config

Starting and Stopping the Configuration Recorder

You can start and stop the configuration recorder at any time using the web console or the CLI. During the time the configuration recorder is stopped, it doesn't monitor or record changes. But it does retain existing configuration items. To stop it using the following CLI command, you must specify the configuration recorder's name, which is default.

```
aws configservice stop-configuration-recorder --configuration-recorder-name default
```

To start the recorder, issue the following command:

```
aws configservice start-configuration-recorder --configuration-recorder-name default
```

CloudTrail, CloudWatch, and AWS Config

Recording Software Inventory

AWS Config can record software inventory changes on EC2 instances and on-premises servers. This includes the following:

Applications

AWS components like the CLI and SDKs

The name and version of the operating system

IP address, gateway, and subnet mask

Firewall configuration

Windows updates

To have AWS Config track these changes, you must enable inventory collection for the server using the EC2 Systems Manager. You also must ensure AWS Config monitors the SSM : ManagedInstanceInventory resource type

Managed and Custom Rules

In addition to monitoring resources changes, AWS Config lets you specify rules to define the optimal baseline configuration for your resources. AWS Config also provides customizable, predefined rules that cover a variety of common scenarios. For example, you may want to verify that CloudTrail is enabled, that every EC2 instance has an alarm tracking the CPU Utilzation metric, that all EBS volumes are encrypted, or that multifactor authentication (MFA) is enabled for the root account. If any resources are noncompliant, AWS Config flags them and generates an SNS notification.

When you activate a rule, AWS Config immediately checks monitored resources against the rules to determine whether they're compliant. After that, how often it reevaluates resources is based on how the rule is configured. A reevaluation can be triggered by configuration changes or periodically. Periodic checks can occur every hour or every 3, 6, 12, or 24 hours. Note that even if you turn off the configuration recorder, periodic rules will continue to run.

CloudTrail, CloudWatch, and AWS Config

Summary

You must configure CloudWatch and AWS Config before they can begin monitoring your resources. CloudTrail automatically logs only the last 90 days of management events even if you don't configure it. It's therefore a good idea to configure these services early on in your AWS deployment.

CloudWatch, CloudTrail, and AWS Config serve different purposes, and it's important to know the differences among them and when each is appropriate for a given use case.

CloudWatch tracks performance metrics and can take some action in response to those metrics. It can also collect and consolidate logs from multiple sources for storage and searching, as well as extract metrics from them.

CloudTrail keeps a detailed record of activities performed on your AWS account for security or auditing purposes. You can choose to log read-only or write-only management or data events.

CloudTrail, CloudWatch, and AWS Config

Summary

AWS Config records resource configurations and relationships past, present, and future. You can look back in time to see how a resource was configured at any point. AWS Config can also compare current resource configurations against rules to ensure that you're in compliance with whatever baseline you define.

CloudTrail, CloudWatch, and AWS Config

Exam Essentials

Know how to configure the different features of CloudWatch. CloudWatch receives and stores performance metrics from various AWS services. You can also send custom metrics to CloudWatch. You can configure alarms to take one or more actions based on a metric. CloudWatch Logs receives and stores logs from various resources and makes them searchable.

Know the differences between CloudTrail and AWS Config. CloudTrail tracks events, while AWS Config tracks how those events ultimately affect the configuration of a resource. AWS Config organizes configuration states and changes by resource, rather than by event.

Understand how CloudWatch Logs integrates with and complements CloudTrail. CloudTrail can send trail logs to CloudWatch Logs for storage, searching, and metric extraction.

Understand how SNS works. CloudWatch and AWS Config send notifications to an Amazon SNS topic. The SNS topic passes these notifications on to a subscriber, which consists of a protocol and endpoint. Know the various protocols that SNS supports.

The Domain Name System and Network Routing—Amazon Route 53 and Amazon CloudFront

Introduction

- Making your cloud resources accessible to the people and machines that will consume them is just as important as creating them in the first place. Facilitating network access to your content generally has three goals: availability, speed, and accurately getting the right content to the right users.
- The Domain Name System (DNS) is primarily concerned with making your resources available across a network using a human-readable name. The Internet's DNS infrastructure ensures that content living behind IP addresses can be reliably associated with human-readable domain names—like amazon.com.
- Amazon's Route 53 is built to manage domain services, but it can also have a significant impact on the precision and speed with which resources are delivered.
- CloudFront, Amazon's global content delivery network (CDN), can take both speed and accuracy a few steps further.

The Domain Name System and Network Routing—Amazon Route 53 and Amazon CloudFront

The Domain Name System

- DNS is responsible for mapping human-readable domain names (like example.com) to the machine-readable IP addresses (like 93.184.216.34) they represent.
- Whenever you launch a new network-facing service on AWS—or anywhere else—and want it to make it accessible through a readable name, you need to satisfy some configuration requirements. But before you can learn to do that, it's important to be familiar with the basic concepts on which name services are built.
- In this section, we'll define the key elements of DNS infrastructure, particularly the way they're used within Amazon Route 53.

The Domain Name System and Network Routing—Amazon Route 53 and Amazon CloudFront

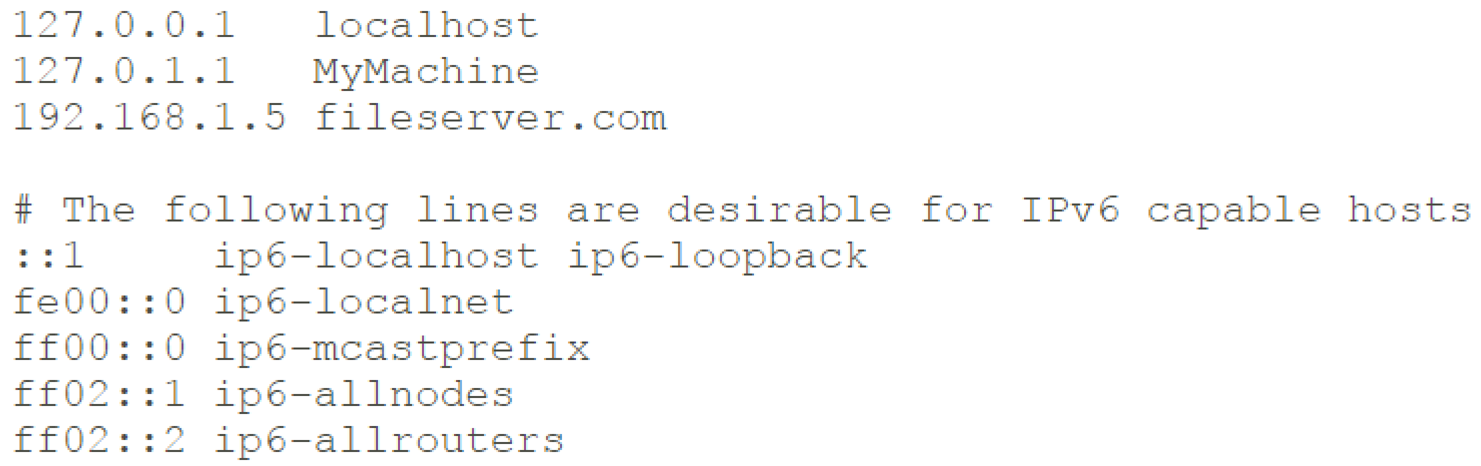
Namespaces

- The addressing structure organizing the billions of objects making up the Internet is managed through naming conventions. If, for instance, there was more than one website called amazon.com or more than one resource identified by the IP address 205.251.242.103, then things would quickly get chaotic. So, there's got to be a reliable, top-down administration authority.
- The Internet naming system is maintained within the domain name hierarchy namespace, which controls the use of human-readable names. It's a *hierarchy* in the sense that the Internet can be segmented into multiple smaller namespaces through the assignment of blocks of public or private IP addresses or through the use of top-level domains (TLDs).
- Both the Internet Protocol and the domain name hierarchy are administrated through the Internet Corporation for Assigned Names and Numbers (ICANN).

The Domain Name System and Network Routing—Amazon Route 53 and Amazon CloudFront

Name Servers

– Associating a domain name like amazon.com with its actual IP address is the job of a name server. All computers will have a simple name server database available locally. That database might contain entries associating *hostnames* (like local host) with an appropriate IP address. The following code is an example of the /etc/hosts file from a typical Linux machine. It includes a line that allows you to enter fileserver.com in your browser to open the home page of a web server running within the local network with the IP 192.168.1.5.



```
127.0.0.1   localhost
127.0.1.1   MyMachine
192.168.1.5 fileserver.com

# The following lines are desirable for IPv6 capable hosts
::1     ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
```

The Domain Name System and Network Routing—Amazon Route 53 and Amazon CloudFront

– If a query isn't satisfied by a local name server, it will be forwarded to one of the external DNS name servers specified in your computer's network interface configuration. Such a configuration might point to a public DNS server like Google's at 8.8.8.8 or OpenDNS at 208.67.222.222. Their job is to provide an IP address matching the domain name you entered so your application (a web browser, for instance) can complete your request.

Domains and Domain Names

– In terms of Internet addressing, a domain is one or more servers, data repositories, or other digital resources identified by a single domain name. A *domain name* is a name that's been registered for the domain that's used to direct network requests to the domain's resources.

Domain Registration

– Top-level name servers must be made aware of a new domain name before they can respond to related queries. Propagating domain name data among name servers is the job of a domain name registrar. Registrars work with registry operators like VeriSign so that domain registrations should be globally authoritative. Among its other roles, Amazon Route 53 acts as a domain name registrar.

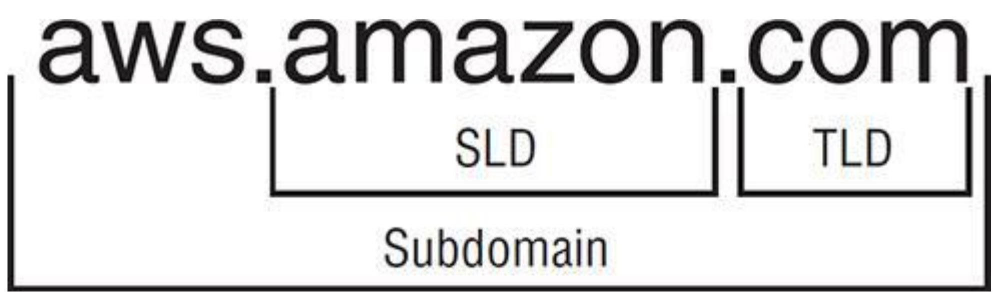
The Domain Name System and Network Routing—Amazon Route 53 and Amazon CloudFront

Domain Layers

– A domain name is made up of multiple parts. The rightmost text of every domain address (like .com or .org) indicates the top-level domain (TLD). The name to the left of the TLD (the *amazon* part of amazon.com) is called the second-level domain (SLD). This SLD designation would also refer to the unique second-level domains used by some countries, like the .co of co.uk that's used for UK-based businesses.

– A subdomain identifies a subset of a domain's resources. Web and email servers from the administration department of a college, for instance, might all use the administration.school.edu name. Thus, dean@administration.school.edu might be a valid email address, while administration.school.edu/apply.pdf administration.school.edu server.

– www.school.edu, api. School.edu and ftp.school.edu are all common examples of subdomains (sometimes referred to as *hosts*). Figure illustrates the parts of a simple subdomain.



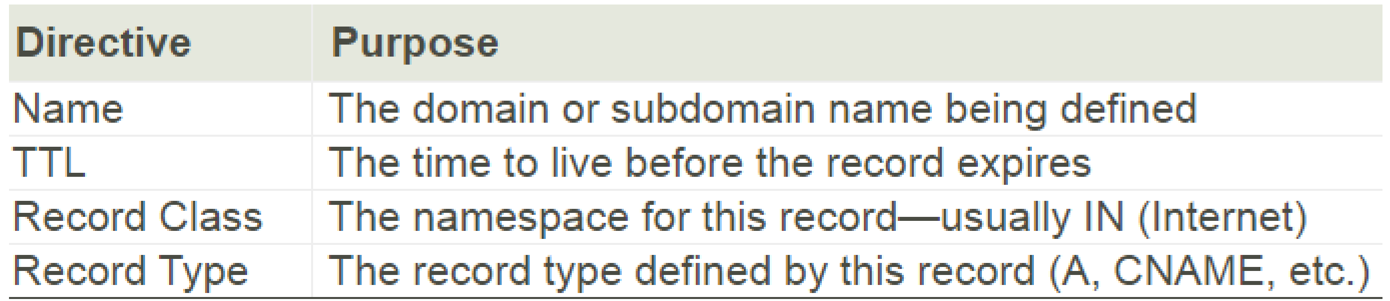
The Domain Name System and Network Routing—Amazon Route 53 and Amazon CloudFront

Fully Qualified Domain Names

- Based on the default DNS settings on many systems, the system's default domain name will be automatically appended when resolving requests for partial domain names. As an example, a request for workstation might be resolved as workstation.localhost. If, however, you want to request a domain name as is, without anything being appended to it, you'll need to use a fully qualified domain name (FQDN).
- An FQDN contains the absolute location of the domain including, at the least, a subdomain and the TLD. In addition, convention will often require a trailing dot after the TLD—which represents the domain root—to confirm that this is, indeed, an FQDN. Addresses in DNS zone files, for instance, will fail without a trailing dot. Here's how that might look:
- Administration.school.edu

Zones and Zone Files

- A *zone* (or hosted zone as Route 53 calls it) is a subset of a DNS domain. A *zone file* is a text file that describes the way resources within the zone should be mapped to DNS addresses within the domain. The file consists of resource records containing the data fields listed in



Directive	Purpose
Name	The domain or subdomain name being defined
TTL	The time to live before the record expires
Record Class	The namespace for this record—usually IN (Internet)
Record Type	The record type defined by this record (A, CNAME, etc.)

The Domain Name System and Network Routing—Amazon Route 53 and Amazon CloudFront

Introduction

- This following example is a typical resource record defining the name server (NS) of the example.com domain as an Internet record (IN) with a one-hour TTL (1h) whose value is

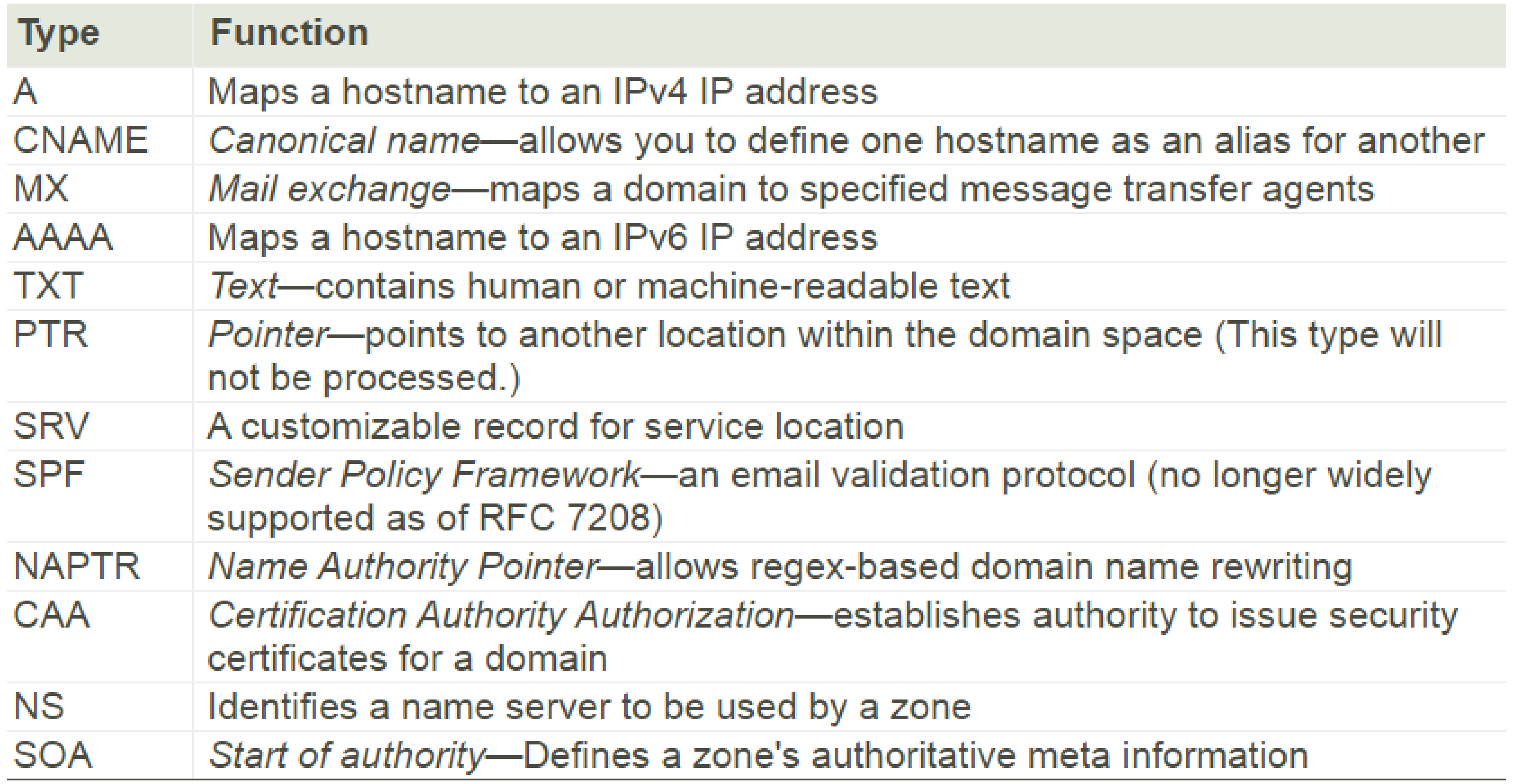
```
      ns-750.awsdns-30.net.:
example.com. 1h IN NS ns-750.awsdns-30.net.
```

Record Types

- The record type you enter in a zone file's resource record will determine how the record's data is formatted and how it should be used. There are currently around 40 types in active use, but Table 8.2 explains only the more common ones that also happen to be offered by Route 53.
- Amazon's Route 53 is built to manage domain services, but it can also have a significant impact on the precision and speed with which resources are delivered.
- CloudFront, Amazon's global content delivery network (CDN), can take both speed and accuracy a few steps further.

The Domain Name System and Network Routing—Amazon Route 53 and Amazon CloudFront

Some common DNS record types



Type	Function
A	Maps a hostname to an IPv4 IP address
CNAME	*Canonical name*—allows you to define one hostname as an alias for another
MX	*Mail exchange*—maps a domain to specified message transfer agents
AAAA	Maps a hostname to an IPv6 IP address
TXT	*Text*—contains human or machine-readable text
PTR	*Pointer*—points to another location within the domain space (This type will not be processed.)
SRV	A customizable record for service location
SPF	*Sender Policy Framework*—an email validation protocol (no longer widely supported as of RFC 7208)
NAPTR	*Name Authority Pointer*—allows regex-based domain name rewriting
CAA	*Certification Authority Authorization*—establishes authority to issue security certificates for a domain
NS	Identifies a name server to be used by a zone
SOA	*Start of authority*—Defines a zone's authoritative meta information

The Domain Name System and Network Routing—Amazon Route 53 and Amazon CloudFront

Alias Records

- It's also possible to route traffic from one domain to another using an alias record. While the use of alias records has not yet been standardized across providers, Route 53 makes them available within record sets, allowing you to connect directly with network-facing resources running on AWS.

Amazon Route 53

- With those DNS basics out of the way, it's time to turn our attention back to AWS. Route 53 provides more than just basic DNS services. In fact, it focuses on four distinct areas: domain registration, DNS management, availability monitoring (health checks), and routing policies (traffic management).
- In case you're curious, the "53" in Route 53 reflects the fact that DNS traffic uses network port 53 to do its job.

Domain Registration

- It's also possible to route traffic from one domain to another using an alias record. While the use of alias records has not yet been standardized across providers, Route 53 makes them available within record sets, allowing you to connect directly with network-facing resources running on AWS.

The Domain Name System and Network Routing—Amazon Route 53 and Amazon CloudFront

Domain Registration

- While there's nothing stopping you from registering your domains through any ICANN-accredited registrar—like GoDaddy—you can just as easily use Route 53. For domains that will be associated with AWS infrastructure, using Route 53 for registration can in fact help simplify your operations.
- You can transfer registration of an existing domain from your current registrar by unlocking the domain transfer setting in the registrar's admin interface and then requesting an authorization code. You'll supply that code to Route 53 when you're ready to do the transfer.
- If you'd prefer to leave your domain with its current registrar, you can still use Route 53 to manage your DNS configuration. Simply copy the name server addresses included in your Route 53 record set and paste them as the new name server values in your registrar's admin interface.

The Domain Name System and Network Routing—Amazon Route 53 and Amazon CloudFront

DNS Management

- However you registered your site name, until you find some way to connect your domain with the resources it's supposed to represent, it won't do you a lot of good. On Route 53, it's the way you create and configure a hosted zone that determines what your users will be shown when they invoke your domain name in a browser, email client, or programmatically.
- Within Route 53, rather than importing a preconfigured zone file (which is certainly an option), you'll usually configure your hosted zones directly via the console or AWS CLI.
- When you create a new hosted zone and enter your domain name, you'll need to tell Route 53 whether you'd like this zone to be publicly or privately hosted. Routing for a privately hosted zone will be available only within the AWS VPCs that you specify. If you want your resources to be accessible to external users (which is true of most domains), then you'll create a publicly hosted zone.
- Route 53 will automatically create an SOA record and provide four name server addresses. From there, it'll be your job to create new record sets defining the relationships between your domain and any subdomains you choose to create, and the resources you want made available. Exercise 8.1 guides you through the process of setting up a functioning hosted zone.

The Domain Name System and Network Routing—Amazon Route 53 and Amazon CloudFront

Availability Monitoring

- You should always be aware of the status of the resources you have running: closing your eyes and just hoping everything is fine has never been a winning strategy. Route 53 offers tools to monitor the health of the resources it's managing and ways to work around problems.
- When you create a new record set, you're given the option of choosing a routing policy (which we'll discuss at length a bit later in this chapter). Selecting any policy besides Simple will make it possible to associate your policy with a health check. A health check, which you create, configure, and name separately, will regularly test the resource that's represented by your record set to confirm it's healthy.
- If everything's OK, Route 53 will continue routing traffic to that resource. But if a preset number of tests pass without an acceptable response, Route 53 will assume the resource is offline and can be set to redirect traffic to a backup resource. Exercise 8.2 illustrates how you can configure a health check that you can use with record sets.

The Domain Name System and Network Routing—Amazon Route 53 and Amazon CloudFront

Routing Policies

- In a world where your users all happily access a single, reliable server instance hosting your application, complicated routing policies make little sense. But since that's not the way things usually work in the real world, you'll generally want to apply flexible protocols to ensure that you're always providing the most reliable and low-latency service possible.
- Route 53 routing policies provide this kind of functionality at the domain level that can be applied globally across all AWS regions. To make the most of their power, you'll need to understand what choices are available and how they work.
- The simplest of the policies is, predictably, called simple. It routes all requests to the IP address or domain name you assign it. Simple is the default routing policy for new record sets.

The Domain Name System and Network Routing—Amazon Route 53 and Amazon CloudFront

Weighted Routing

- A weighted policy will route traffic among multiple resources according to the ratio you set. To explain that better, imagine you have three servers (or load balancers representing three groups of servers) all hosting instances of the same web application. One of the servers (or server groups) has greater unused compute and memory capacity and can therefore handle far more traffic. It would be inefficient to send equal numbers of users to each of the servers. Instead, you can assign the larger server a numeric weight of, say, 50 and then 25 to the other two. That would result in half of all requests being sent to the larger server and 25 percent to each of the others.
- To configure a weighted policy in Route 53, you would create a separate record set for each of your servers, give the same value for the set ID of each of the record sets, and then enter an instance-appropriate numeric value for the weight. The matching set IDs tell Route 53 that those record sets are meant to work together.

The Domain Name System and Network Routing—Amazon Route 53 and Amazon CloudFront

Latency Routing

- Latency-based routing lets you leverage resources running in multiple AWS regions to provide service to clients from the instances that will deliver the best experience. Practically this means that, for an application used by clients in both Asia and Europe, you could place parallel resources in, say, the ap-southeast-1 and eu-west-1 regions. You then create a record set for each resource using latency-based policies in Route 53, with one pointing to your ap-southeast-1 resource and the other to eu-west-1
- Assuming you gave both record sets the same value for Set ID, Route 53 will know to direct requests for those resources to the instance that will provide the lowest latency for each client.

Failover Routing

- A failover routing policy will direct traffic to the resource you identify as *primary* as long as health checks confirm that the resource is running properly. Should the primary resource go offline, subsequent traffic will be sent to the resource defined within a second record set and designated as *secondary*. As with other policies, the desired relationship between record sets is established by using matching set ID values for each set.

The Domain Name System and Network Routing—Amazon Route 53 and Amazon CloudFront

Geolocation Routing

- Unlike latency policies that route traffic to answer data requests as *quickly* as possible, geolocation uses the continent, country, or U.S. state where the request originated to decide *what resource to send*. This can help you focus your content delivery, allowing you to deliver web pages in customer-appropriate languages, restrict content to regions where it's legally permitted, or generate parallel sales campaigns.
- You should be aware that Route 53 will sometimes fail to identify the origin of a requesting IP address. You might want to configure a default record to cover those cases.

Multivalue Answer Routing

- It's possible to combine a health check configuration with multivalue routing to make a deployment more highly available. Each multivalue-based record set points to a single resource and can be associated with a health check.
- As many as eight records can be pointed to parallel resources and connected to each other through matching set ID values. Route 53 will use the health checks to monitor resource status and randomly route traffic among the healthy resources.

The Domain Name System and Network Routing—Amazon Route 53 and Amazon CloudFront

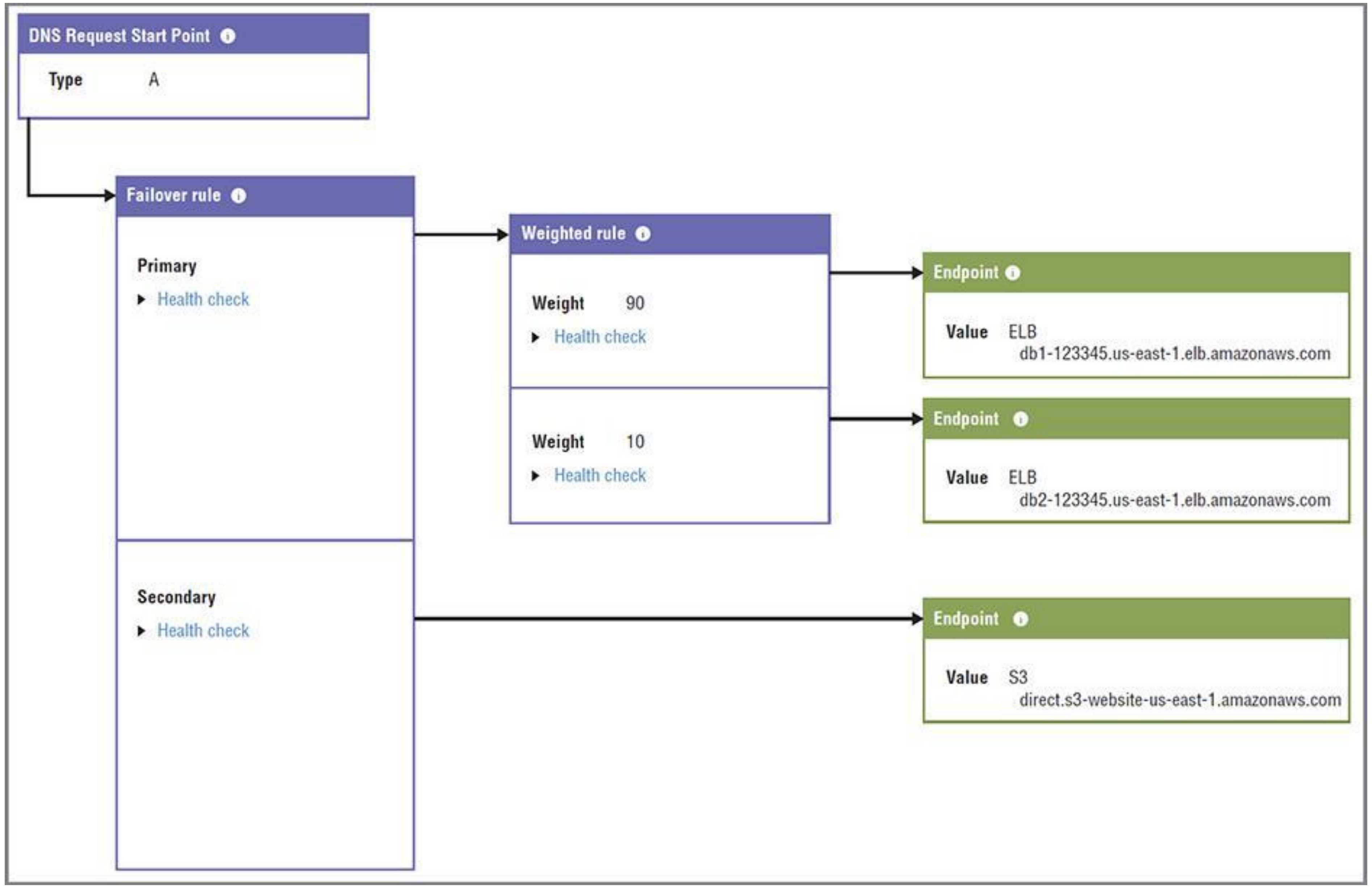
Traffic Flow

- Route 53 Traffic Flow is a console-based graphic interface that allows you to visualize complex combinations of routing policies as you build them. Figure 8.2 shows what a Traffic Flow configuration can look like.
- .

Multivalue Answer Routing

- It's possible to combine a health check configuration with multivalue routing to make a deployment more highly available. Each multivalue-based record set points to a single resource and can be associated with a health check.
- As many as eight records can be pointed to parallel resources and connected to each other through matching set ID values. Route 53 will use the health checks to monitor resource status and randomly route traffic among the healthy resources.

The Domain Name System and Network Routing—Amazon Route 53 and Amazon CloudFront



The Domain Name System and Network Routing—Amazon Route 53 and Amazon CloudFront

- Because it so seamlessly integrates routing policies with all the possible resource endpoints associated with your AWS account, Traffic Flow can make it simpler to quickly build a sophisticated routing structure. You can also use and customize routing templates.
- Traffic Flow offers a routing policy that, as of the time of writing, is not available as a regular Route 53 policy: Geoproximity Routing. Geoproximity gives you the precision of geolocation routing but at a far finer level. Geoproximity routing rules can specify geographic areas by their relationship to either a particular longitude and latitude or to an AWS region. In both cases, setting a bias score will define how widely beyond your endpoint you want to apply your rule..

The Domain Name System and Network Routing—Amazon Route 53 and Amazon CloudFront

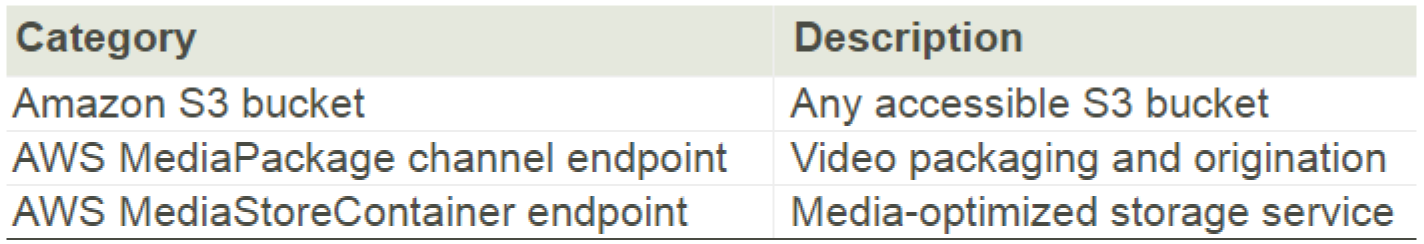
Amazon CloudFront

- As you've seen, Route 53 can be used to optimize the way DNS requests are processed. But Amazon's global content delivery network (CDN) CloudFront can also help solve one of the primary problems addressed by Route 53: getting your content into the hands of your users as quickly as possible.
- A CDN maintains a network of physical edge locations placed geographically close to the end users who are likely to request content. When you configure the way you want your content delivered—as part of what AWS calls a CloudFront distribution—you define how you want your content distributed through that network and how it should then be delivered to your users.
- How do your users "know" to address their requests to your CloudFront endpoint address rather than getting it directly from your resources? Normally, they don't. But when you use Route 53 to configure your domain to direct incoming DNS requests to the CloudFront distribution, users will be automatically routed appropriately.
- When a request is made, CloudFront assesses the user's location and calculates which endpoint will be available to deliver the content with the lowest latency. If this is the first time this content has been requested through the endpoint, the content will be copied from the origin server (an EC2 web server instance, perhaps, or the contents of an S3 bucket). Delivery of subsequent requests will—because the cached copy is still stored at the endpoint—be much faster.

The Domain Name System and Network Routing—Amazon Route 53 and Amazon CloudFront

Amazon CloudFront

- The kind of CloudFront distribution you create will depend on the kind of media you're providing. For web pages and graphic content, you'll select a Web distribution. For video content that's stored in S3 buckets that can use Adobe's Real-Time Messaging Protocol (RTMP), CloudFront's RTMP distribution makes the most sense.
- As you briefly saw in Chapter 3, when you configure your distribution, you'll have the option of adding a free AWS Certificate Manager (ACM) SSL/TLS encryption certificate to your distribution. This will protect your content as it moves between CloudFront and your users' devices from network sniffers and man-in-the-middle attacks.
- **Permitted CloudFront origins**



Category	Description
Amazon S3 bucket	Any accessible S3 bucket
AWS MediaPackage channel endpoint	Video packaging and origination
AWS MediaStoreContainer endpoint	Media-optimized storage service

The Domain Name System and Network Routing—Amazon Route 53 and Amazon CloudFront

AWS CLI Example

The following command will list all the hosted zones in Route 53 within your AWS account.
Among the data that will be returned is an Id for each zone. You can take the value
of Id and pass it to the get-hosted-zone command to return record set details for that zone.
$ aws route53 list-hosted-zones $ aws route53 get-hosted-zone --id
/hostedzone/Z38LGIZCB3CSZ3

The Domain Name System and Network Routing—Amazon Route 53 and Amazon CloudFront

Summary

- The DNS system manages Internet resource addressing through two top-level name spaces: the Internet Protocol (IP addresses) and the domain name hierarchy. The Internet Corporation for Assigned Names and Numbers (ICANN) administrates name servers and domain name registrars (like Route 53) through registry operators (like VeriSign).
- A fully qualified domain name consists of a TLD (like org or com) and an SLD (like amazon). FQDNs are also often given a trailing dot representing "root.“
- DNS configuration details are organized as hosted zones, where resource record sets are created to control the way you want inbound domain traffic to be directed. That behavior can be defined through any one of a number of record types like A, CNAME, and MX.
- A newly created Route 53 hosted zone will contain a start of authority (SOA) record set and a list of name servers to which requests can be directed. You'll need to create at least one record set so Route 53 will know from which domain name to expect requests.
- Route 53 can be configured to regularly monitor the run status of your resources using health checks. Besides alerting you to problems, health checks can also be integrated with Route 53 routing policies to improve application availability.

The Domain Name System and Network Routing—Amazon Route 53 and Amazon CloudFront

Summary

- Weighted routing policies let you direct traffic among multiple parallel resources proportionally according to their ability to handle it.
- Latency routing policies send traffic to multiple resources to provide the lowest-latency service possible.
- Failover routing monitors a resource and, on failure, reroutes subsequent traffic to a backup resource.
- Geolocation routing assesses the location of a request source and directs responses to appropriate resources.
- Amazon CloudFront is a CDN that caches content at edge locations to provide low-latency delivery of websites and digital media.

The Domain Name System and Network Routing—Amazon Route 53 and Amazon CloudFront

Exam Essentials

- **Understand how DNS services enable predictable and reliable network communication.** DNS registration ensures that domain names are globally unique and accessible. Domain requests are resolved using name servers—both local and remote.
- **Understand DNS naming conventions.** You should be familiar with "parsing" domain names. Each of the TLD, SLD, and subdomain/host sections will be read by DNS clients in a predictable way.
- **Be familiar with the key DNS record types.** Recognize the function of key record types, including A and AAAAA (address records for IPv4 and IPv6), CNAME (canonical name record or alias), MX (mail exchange record), NS (name server record), and SOA (start of authority record).
- **Be familiar with Route 53 routing policies.** Recognize the function of key routing policies, including simple (for single resources), weighted (routing among multiple resources by percentage), latency (low-latency content delivery), failover (incorporate backup resources for higher availability), and geolocation (respond to end user's location).
- **Understand how to create a CloudFront distribution.** CloudFront distributions can be configured to deliver content through low-latency, geographically-based content to end users.

Client Logo

Thank You