AWS Certified SysOps Administrator - Associate (SOA-C01)

Course Navigation

Next Sections

Linux Academy

AWS Certified SysOps Administrator - Associate (SOA-C01)

Course Navigation

Monitoring and Metrics
Section 8

Management, Governance, and Cost Controls
Section 9

Previous Sections

Linux Academy

Security on AWS

Shared Responsibility Model

Customer

Responsible for security **IN** the cloud

- Customer data
- Platform, application, and IAM
- OS patching on EC2
- Antivirus
- Network, and firewall configuration
- Multi-factor authentication
- Password and key rotation
- Security groups
- Resource-based policies
- Access control lists
- VPC
- Operating system-level patches
- Data in transit and at rest

AWS

Responsible for security **OF** the cloud

- Regions, availability zones, and edge locations
- Physical server level and below
- Fire/power/climate management
- Storage device decommissioning according to industry standards
- Personnel security
- Network device security and ACLs
- API access endpoints use SSL for secure communication
- DDoS protection
- EC2 instances and spoofing protection (ingress/egress filtering)
- Port scanning against rules even if it's your own environment
- EC2 instance hypervisor isolation
  - Instances on the same physical device are separated at the hypervisor level; they are independent of each other
- Underlying OS patching on Lambda, RDS, DynamoDB, and other managed services; customer focuses on security

IAM Users

Root User

- The user created when an AWS account is created.
- The credentials are the email and password used when signing up for an AWS account.
- By default, the root user has *full* administrative rights and access to every part of the account.

Best Practices for Root User

- The root user should *not* be used for daily work and administration:
  - Another user should be created for daily work that has admin rights.
- The root user account should not have access keys; delete them if they exist.
- The root user should always use *multi-factor authentication (MFA)*, like Google Authenticator.

IAM Users

- A new user has an *implicit deny* for all AWS services, requiring a policy be added to grant them access.
- Users receive unique credentials (username, password, and possibly access keys).
- Users can have IAM policies applied directly to them, or they can be a member of a group that has policies attached.
- With policies, an explicit deny always overrides an explicit allow from attached policies:
  - For instance, all policies attached to a user will be ignored if a single deny-all policy is added.

Best Practices for Users

- *Never* store or "pass" your access credentials to an EC2 instance — use SSH forwarding.
- MFA *can* and *should* be used for user accounts.
- Access credentials are unique and should never be shared.

Next

Back to Main

IAM Groups

- Allow for policy assignments to multiple users at the same time
- Permissions may be assigned to group
- More organized and efficient way to manage users and policies
- Cannot be nested
- Users may be a member of multiple groups

Best Practices for IAM Groups

- Organize users by function (e.g., DB admins, developers, architects, etc.)
- Assign IAM policies to groups, not individual users

Back

IAM Policies

- A document that states one or more permissions (JSON formatted).
- An explicit deny always overrides an explicit allow.
- This allows for the use of a deny-all policy to quickly restrict *all* access a user may have.

Managed Policies

- AWS provides pre-built policy templates to assign to users and groups. Examples include:
  - **Administrator access**: Full access to *all* AWS resources
  - **Power user access**: Admin access except it does not allow user/group management
  - **Read-only access**: Only view AWS resources (e.g., user can only view what is in an S3 bucket)

Custom Policies

- You can also create custom IAM permission policies using the policy generator or write them from scratch.
- More than one policy can be attached to a user or group at the same time.
- Policies cannot be directly attached to AWS resources (such as an EC2 instance).

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "s3:ListBucket",
    "Resource": "arn:aws:s3:::example_bucket"
  }
}
```

Course Navigation

Security on AWS
Section 1

Shared Responsibility Model
IAM: Users and Groups
IAM: Policies
IAM: Roles
IAM: Multi-Factor Authentication (MFA)
Amazon S3: Bucket Policies
S3: Data Integrity
Amazon VPC: Security Groups and NACLs
AWS STS: Federation
Amazon Inspector
AWS KMS Essentials
AWS Certificate Manager
AWS Web Application Firewall (WAF)
AWS Trusted Advisor

Compute
Section 2

IAM Roles

- Temporary security credentials in AWS managed by Secure Token Service (STS).
- Another entity can **assume** the specific permissions defined by the role.
- These entities include:
  - AWS resources (such as an EC2 instance)
  - A user outside of our AWS account who needs temporary access

Roles with AWS Services

- Roles must be used because policies cannot be directly attached to AWS services.
- Services can only have **one** role attached at a time.
- You should **never** pass or store credentials to an EC2 instance — instead, use roles.
- Example: An EC2 instance needs to be able to read data from an S3 bucket:
  - The instance assumes a role with S3 read-only permissions from IAM.
  - The instance can then read objects from the bucket specified in the role's policies.
- You may change roles on running EC2 instances through the console and CLI.

Other Uses of Roles

- Cross-Account Access (Delegation):
  - Provide access to another AWS user from another account.
- Identity Federation:
  - Users outside AWS can assume a role for temporary access to AWS accounts and resources.
  - These users assume an **identity provider** access role.
  - Example identity providers:
    - Active Directory
    - Single sign-on providers, like Facebook, Google, Amazon, etc.

Course Navigation

Security on AWS
Section 1

Shared Responsibility Model
IAM: Users and Groups
IAM: Policies
IAM: Roles
IAM: Multi-Factor Authentication (MFA)
Amazon S3: Bucket Policies
S3: Data Integrity
Amazon VPC: Security Groups and NACLs
AWS STS: Federation
Amazon Inspector
AWS KMS Essentials
AWS Certificate Manager
AWS Web Application Firewall (WAF)
AWS Trusted Advisor

Compute
Section 2

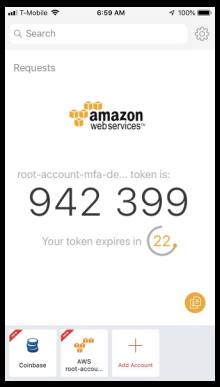
IAM Multi-Factor Authentication (MFA)

What Is Multi-Factor Authentication?

- A security method that requires multiple separate authentications.
- One authentication option we have with AWS uses time-based codes.
- Familiar example of MFA:
  - You go to an ATM to withdraw money from your bank account.
  - This requires both the physical card and a PIN.
  - This example uses two-factor authentication, which is a form of MFA.

AWS Scenario

- Enable MFA in order to access the AWS console:
  - Users type in their username and password as well as a time-based code:
    - The username and password are not enough to be authenticated.
    - The time-based code can be on the user's computer, smartphone, or a device they carry around.
- This should be turned on for users who have access to the console.
- **MFA Delete** for S3 objects can be used to mitigate accidental deletions.



Virtual MFA

YubiKey

Gemalto Token

Course Navigation

Security on AWS
Section 1

Shared Responsibility Model
IAM: Users and Groups
IAM: Policies
IAM: Roles
IAM: Multi-Factor Authentication (MFA)
Amazon S3: Bucket Policies
S3: Data Integrity
Amazon VPC: Security Groups and NACLs
AWS STS: Federation
Amazon Inspector
AWS KMS Essentials
AWS Certificate Manager
AWS Web Application Firewall (WAF)
AWS Trusted Advisor

Compute
Section 2

Amazon S3: Bucket Policies

JSON statement used to allow or deny permissions across objects in a single bucket

Elements of a Bucket Policy

Effect
- Defines whether to allow or deny the action

Action
- Actions we want to allow or deny
- **Important**: An explicit deny always overrides an explicit allow

Resource
- Used to identify resources (like a bucket or object) with Amazon Resource Names (ARNs)

Principal
- An account or user that this policy applies to
- Specific to S3 bucket policies, not user policies

Sid (Optional)
- Statement identifier that provides a way to include information about an individual statement

Condition (Optional)
- Specify conditions for when the policy is in effect
- Example: *PutObject* permission requiring objects to be stored using server-side encryption

Allow PutObject

Require Server-Side Encryption

Course Navigation

Security on AWS
Section 1

Shared Responsibility Model
IAM: Users and Grou
IAM: Policies
IAM: Roles
IAM: Multi-Factor Authentication (MFA
Amazon S3: Bucket Policies
S3: Data Integrity
Amazon VPC: Security Groups and NACLs
AWS STS: Federation
Amazon Inspector
AWS KMS Essentials
AWS Certificate Mana
AWS Web Application Firewall (WAF)
AWS Trusted Advisor

Compute
Section 2

Amazon S3: Bucket Policies

JSON statement used to allow or deny permissions across objects in a single bucket

Elements of a Bucket Policy

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Sid": "PutObjectAcl",
    "Effect": "Allow",
    "Principal": {
      "AWS": [
        "arn:aws:iam::111122223333:tom",
        "arn:aws:iam::444455556666:chris"
      ]
    },
    "Action": [
      "s3:PutObject",
      "s3:PutObjectAcl"
    ],
    "Resource": [
      "arn:aws:s3:::examplebucket/*"
    ]
  }]
}
```

llow

mazon

nation about an

ored using server-side

Allow PutObject

Require Server-Side Encryption

Course Navigation

Security on AWS
Section 1

Shared Responsibility Model
IAM: Users and
IAM: Policies
IAM: Roles
IAM: Multi-Facto Authentication
Amazon S3: Buc Policies
S3: Data Integrity
Amazon VPC: Sec Groups and NACl
AWS STS: Federat
Amazon Inspectc
AWS KMS Essenti
AWS Certificate M
AWS Web Application Firewall (WAF)
AWS Trusted Advisor

Compute
Section 2

Amazon S3: Bucket Policies

JSON statement used to allow or deny permissions across objects in a single bucket

Elements of a Bucket Policy

Effect

```
{
  "Version": "2012-10-17",
  "Statement": [{
      "Sid": "PutObject",
      "Effect": "Deny",
      "Principal": "*",
      "Action": [
        "s3:PutObject"
      ],
      "Resource": [
        "arn:aws:s3:::examplebucket/*"
      ],
      "Condition": {
        "StringNotEquals": {
          "s3:x-amz-server-side-encryption": "AES256"
        }
      }
    ]
}
```

- Example: *PutObject* permission requiring objects to be stored using server-side encryption

Allow PutObject

Require Server-Side Encryption

Course Navigation

Security on AWS
Section 1

Shared Responsibility Model
IAM: Users and Groups
IAM: Policies
IAM: Roles
IAM: Multi-Factor Authentication (MFA)
Amazon S3: Bucket Policies
S3: Data Integrity
Amazon VPC: Security Groups and NACLs
AWS STS: Federation
Amazon Inspector
AWS KMS Essentials
AWS Certificate Manager
AWS Web Application Firewall (WAF)
AWS Trusted Advisor

Compute
Section 2

Versioning
- Enable to store new versions for every modification or deletion
  - Helps with accidental deletion by creating a version for deleted objects

Replication
- Objects are replicated across Availability Zones automatically
- Standard and Infrequent Access options at different price points

Multi-Factor Authentication Delete (MFA Delete)
- Uses MFA to prevent accidental deletion of objects
  - Requires **Versioning** enabled on a bucket
- We can enforce the use of MFA in order to permanently delete an object
- Only root accounts (the bucket owner) can access this feature

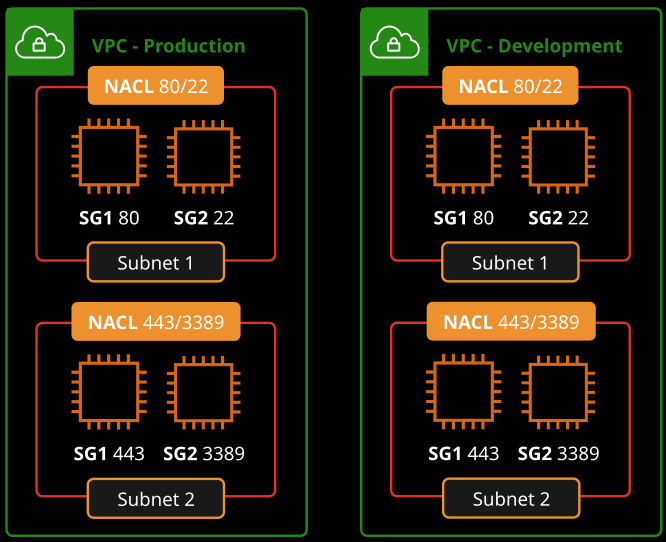
```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Sid": "MFADelete",
    "Effect": "Deny",
    "Principal": "*",
    "Action": "s3:*",
    "Resource": "arn:aws:s3:::examplebucket/taxdocuments/*",
    "Condition": {
      "Null": {
        "aws:MultiFactorAuthAge": true
      }
    }
  }]
}
```

VPC (Virtual Private Cloud)
- Isolate workloads into separate VPCs (based on application, department, test, dev, etc.)

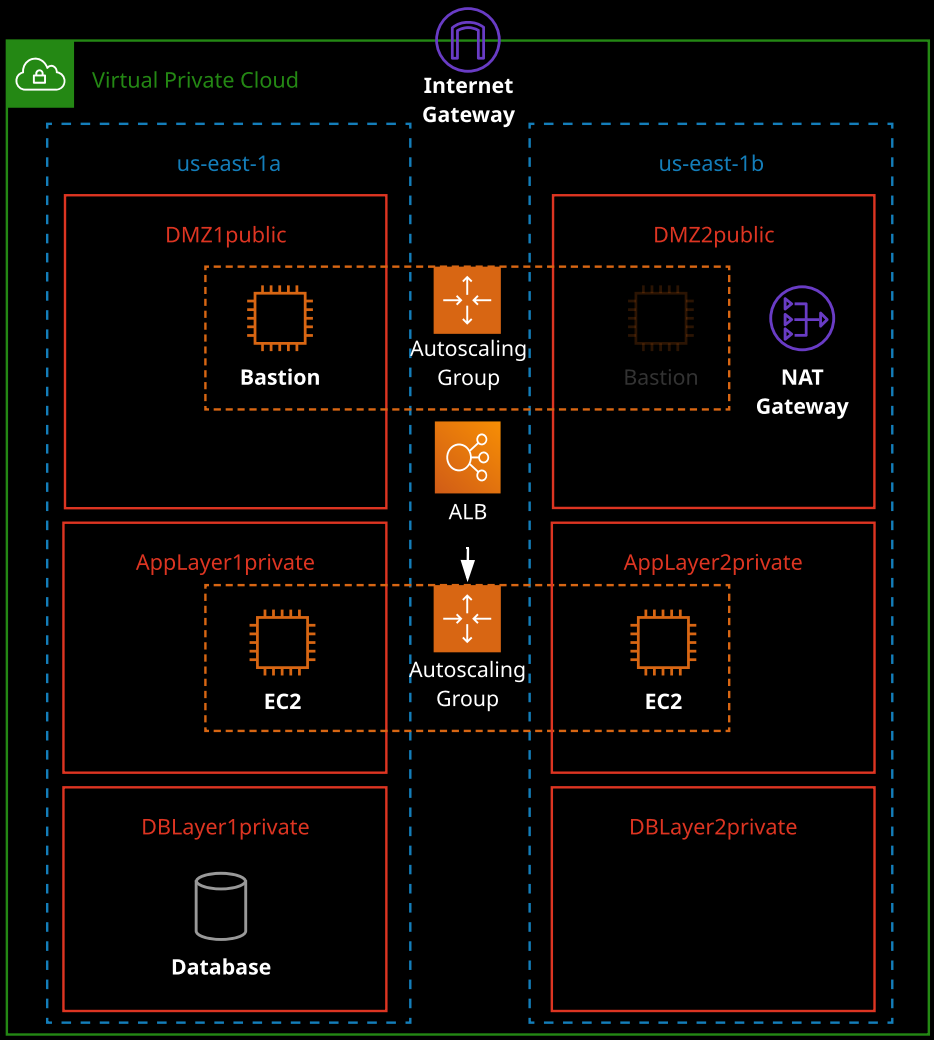
Security Groups
- Group instances with similar functions
- Stateful = every allowed TCP or UDP port will be allowed in both directions

NACLs (Network Access Control Lists)
- Stateless = inbound and outbound rules are separate, no dependencies
- Granular control over IP protocols (allow and deny rules for inbound and outbound evaluated in order)
- Work with security groups (NACL applies for the whole subnet, security groups apply to members)
- Ephemeral ports: Client requests depending on OS (ports 1024-65535)

Host-Based Firewalls: OS-level firewalls as needed



Example VPC


Virtual Private Cloud
Internet Gateway
us-east-1a
us-east-1b
DMZ1public
DMZ2public
Bastion
Autoscaling Group
Bastion
NAT Gateway
ALB
AppLayer1private
AppLayer2private
EC2
Autoscaling Group
EC2
DBLayer1private
DBLayer2private
Database

Course Navigation

Security on AWS
Section 1

Shared Responsibility Model
IAM: Users and Groups
IAM: Policies
IAM: Roles
IAM: Multi-Factor Authentication (MFA)
Amazon S3: Bucket Policies
S3: Data Integrity
Amazon VPC: Security Groups and NACLs
AWS STS: Federation
Amazon Inspector
AWS KMS Essentials
AWS Certificate Manager
AWS Web Application Firewall (WAF)
AWS Trusted Advisor

Compute
Section 2

Secure Token Service (STS) is an extension of IAM that allows for management of temporary security credentials for IAM users or federated users.

- It allows for granular control of how long the access remains active
  - Fifteen minutes to one hour (default = 1 hour)
- Credentials are not stored with the user or service granted temporary access
  - A token is attached to the access request or API call
- Beneficial in a number of ways
  - Low risk of credentials being exposed (not distributed)
  - Do not have to create IAM identities for every user
  - Because they are temporary in nature, there is no need to rotate keys
- STS uses a single endpoint: *https://sts.amazonaws.com*
  - This single endpoint resides in `us-east-1` (N. Virginia)
  - Latency can be reduced by using STS API calls to regions that support them
  - Temporary credentials have global scope, just like IAM

Identity Federation

Federation: Providing a non-AWS user temporary AWS access by linking that user's identity across multiple identity systems

Federation with Third-Party Providers:

- Most commonly used in web and mobile applications
- Amazon Cognito allows for creation of unique identities for users
- Uses identity providers to federate them
  - Facebook, Google, Amazon, etc.

Establishing Single Sign-On (SSO) Using SAML 2.0:

- Most commonly used in enterprise environments with an existing directory system
  - Active Directory, etc.
- Federated users can access AWS resources using their corporate domain accounts
- Federation also aids user management by allowing central management of accounts

Course Navigation

Security on AWS
Section 1

Shared Responsibility Model
IAM: Users and Groups
IAM: Policies
IAM: Roles
IAM: Multi-Factor Authentication (MFA)
Amazon S3: Bucket Policies
S3: Data Integrity
Amazon VPC: Security Groups and NACLs
AWS STS: Federation
Amazon Inspector
AWS KMS Essentials
AWS Certificate Manager
AWS Web Application Firewall (WAF)
AWS Trusted Advisor

Compute
Section 2

Inspector can:
- Analyze the behavior of your AWS resources
- Test network accessibility and security state
- Assesses for security vulnerabilities and deviations from best practices

Target: A collection of EC2 instances

Assessment Template: Composed of security rules and produces a list of findings

Assessment Run: Applying the assessment template to a target

Findings: Security report, organized by severity level

Features:
- Configuration scanning and activity monitoring engine
  - Determines what a target looks like, its behavior, and any dependencies it may have
  - Identifies security and compliance issues
- Built-in content library
  - Rules and reports built into Inspector
  - Best practice, common compliance standard, and vulnerability evaluations
  - Detailed recommendations for resolving issues
- API automation
  - Allows for security testing to be included in the development and design stages

NOTE: AWS does not guarantee that following the provided recommendations will resolve every potential security issue.

EC2

Inspector agent installed
Tag: Key/Value

Inspector

- Security best practice
- Runtime behavior analysis
- Common Vulnerabilities/Exposures
- CIS Security Config Benchmarks

Course Navigation

Security on AWS
Section 1

Shared Responsibility Model
IAM: Users and Groups
IAM: Policies
IAM: Roles
IAM: Multi-Factor Authentication (MFA)
Amazon S3: Bucket Policies
S3: Data Integrity
Amazon VPC: Security Groups and NACLs
AWS STS: Federation
Amazon Inspector
AWS KMS Essentials
AWS Certificate Manager
AWS Web Application Firewall (WAF)
AWS Trusted Advisor

Compute
Section 2

KMS - Key Management Service

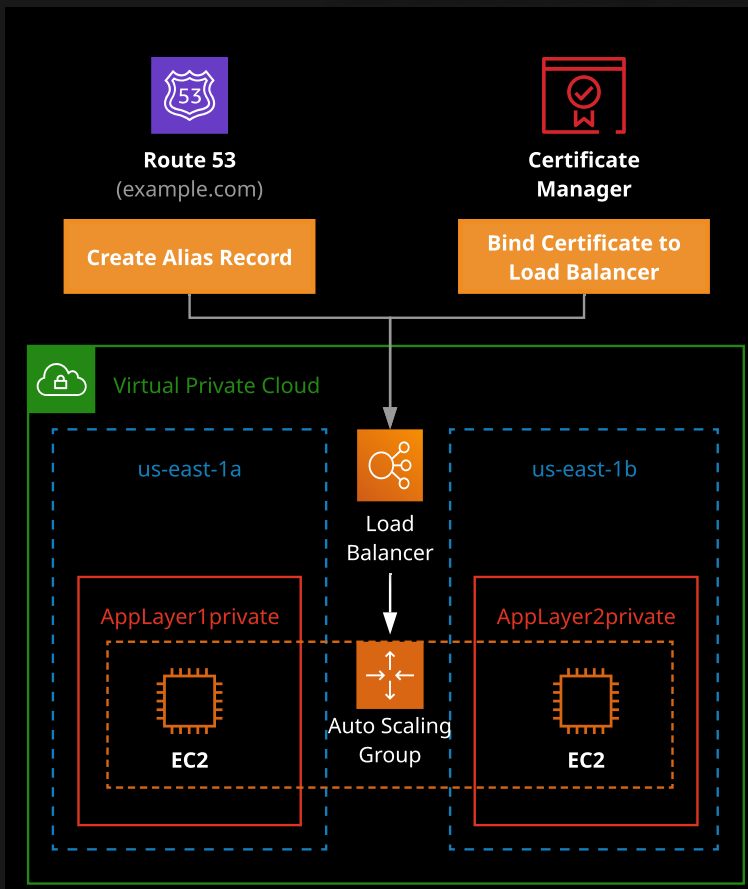
- Easy way to control access to your data using managed encryption
- Integrated with AWS services including EBS, S3, and RedShift to simplify encryption of your data
- Create, rotate, disable, enable, and define usage policies for master keys
- KMS keys are region-specific
- Data encrypted under a key becomes irretrievable if the key is lost
- Key usage is recorded in CloudTrail logs for audit purposes

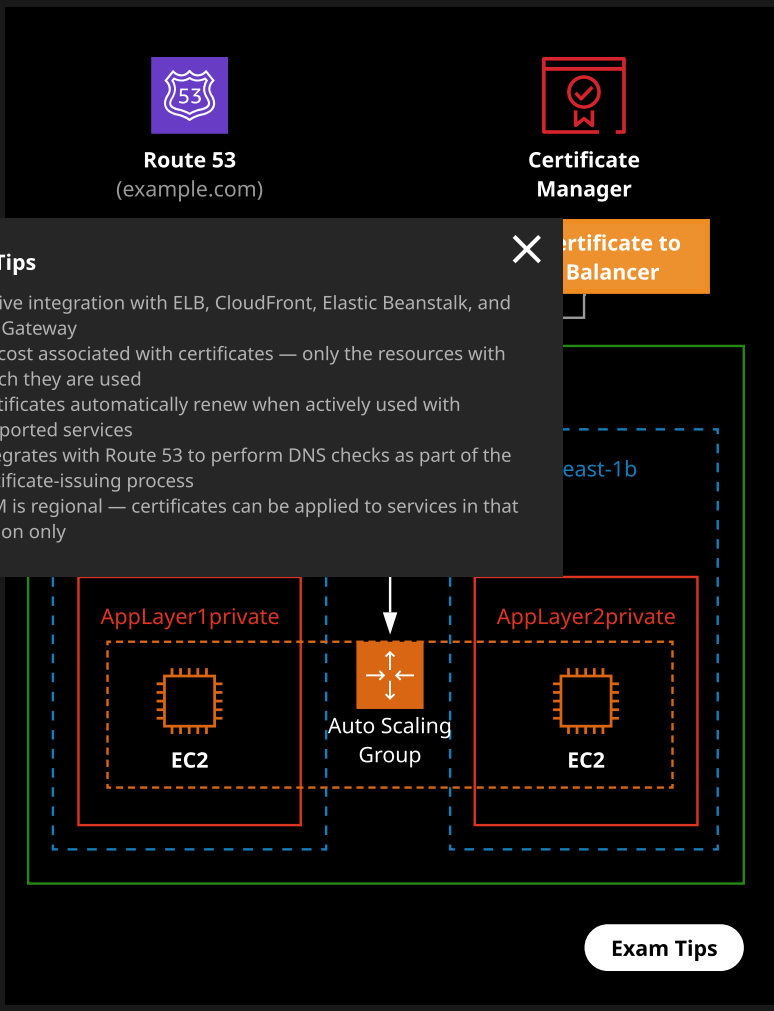
CloudHSM (Hardware Security Module)

- Dedicated hardware security modules under your exclusive control
- FIPS 140-2 **Level 3** compliance
- Designed to integrate with VPC
- Integrates with PKCS#11, Java JCE, and Microsoft CNG
- Can connect to CloudHSM from your on-premises datacenter using VPN or AWS Direct Connect

Security on AWS

AWS Certificate Manager (ACM)



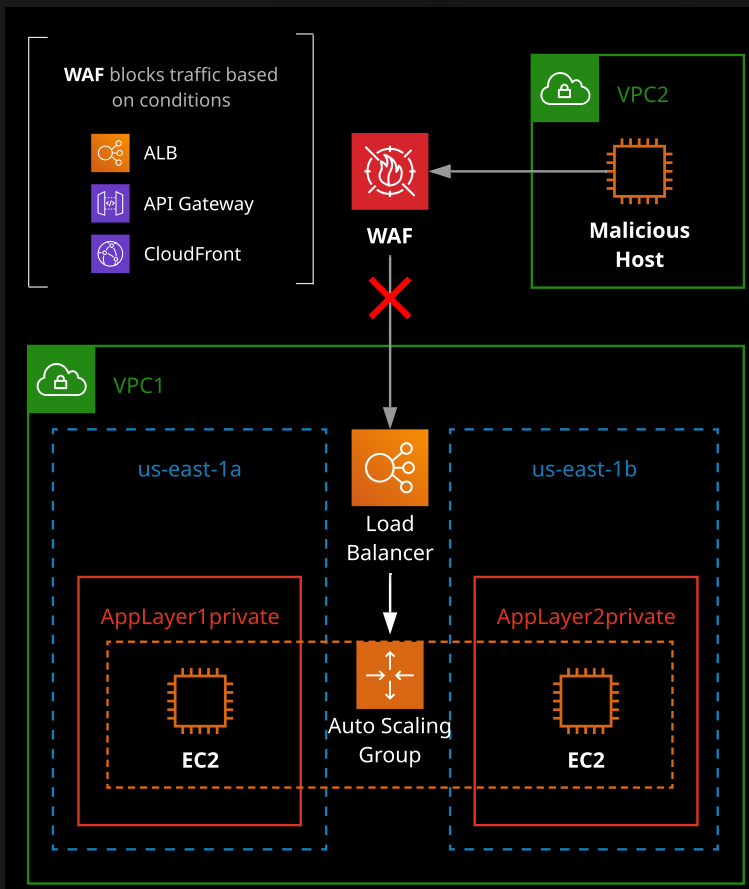


Exam Tips

- Native integration with ELB, CloudFront, Elastic Beanstalk, and API Gateway
- No cost associated with certificates — only the resources with which they are used
- Certificates automatically renew when actively used with supported services
- Integrates with Route 53 to perform DNS checks as part of the certificate-issuing process
- ACM is regional — certificates can be applied to services in that region only

Security on AWS

AWS Web Application Firewall (WAF)



Exam Tips

Security on AWS

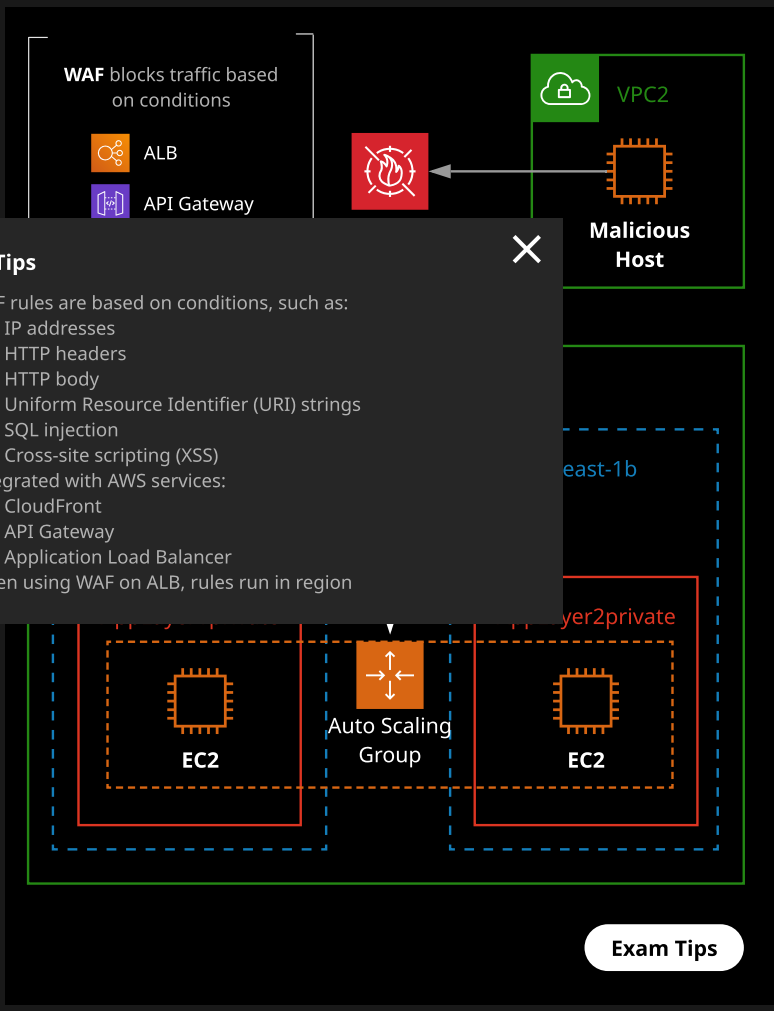
AWS Web Application Firewall (WAF)

Course Navigation

Security on AWS
Section 1

- Shared Responsibility Model
- IAM: Users and Groups
- IAM: Policies
- IAM: Roles
- IAM: Multi-Factor Authentication (M
- Amazon S3: Bucke Policies
- S3: Data Integrity
- Amazon VPC: Secu Groups and NACL
- AWS STS: Federati
- Amazon Inspector
- AWS KMS Essentials
- AWS Certificate Mgr.
- AWS Web Application Firewall (WAF)
- AWS Trusted Advisor

Compute
Section 2



Exam Tips

- WAF rules are based on conditions, such as:
  - IP addresses
  - HTTP headers
  - HTTP body
  - Uniform Resource Identifier (URI) strings
  - SQL injection
  - Cross-site scripting (XSS)
- Integrated with AWS services:
  - CloudFront
  - API Gateway
  - Application Load Balancer
- When using WAF on ALB, rules run in region

Exam Tips

Back to Main

Linux Academy

Course Navigation

Security on AWS
Section 1

Shared Responsibility Model
IAM: Users and Groups
IAM: Policies
IAM: Roles
IAM: Multi-Factor Authentication (MFA)
Amazon S3: Bucket Policies
S3: Data Integrity
Amazon VPC: Security Groups and NACLs
AWS STS: Federation
Amazon Inspector
AWS KMS Essentials
AWS Certificate Mgr.
AWS Web Application Firewall (WAF)
AWS Trusted Advisor

Compute
Section 2

AWS Trusted Advisor provides real-time guidance to help ensure adherence to AWS best practices.

- Allows an AWS customer to get reports on their environment, including:
  - Cost Optimization
  - Performance
  - Security
  - Fault Tolerance
  - Service Limits
- Available to all customers:
  - Access to seven core checks:
    - S3 Bucket Permissions
    - Security Groups - Specific Ports Unrestricted
    - IAM Use
    - MFA on Root Account
    - EBS Public Snapshots
    - RDS Public Snapshots
    - Service Limits
- Available to Business and Enterprise Support customers:
  - Access to the full set of checks
  - Notifications
    - Weekly updates
    - Create alerts and automate actions with CloudWatch
  - Programmatic access
    - Retrieve results from the AWS Support API

System Status Checks

- Monitor the systems on which your instances run
- Reasons for failure:
  - Loss of network connectivity
  - Loss of system power
  - Software issues on physical host
  - Hardware issues on the physical host that impact network reachability
  - Even though AWS will have to correct the original issue, we can resolve it ourselves
    - For **EBS-backed** instances:
      - Stop and start instance to obtain new hardware
    - For **instance store-backed** instances:
      - Terminate and replace (can't stop) the instance for new hardware
      - Can't recover data

Instance Status Checks

- Monitor the network and software configuration on an instance
- You must intervene to fix
- Reasons for failure:
  - Failed system status checks
  - Incorrect networking or startup configuration
  - Exhausted memory
  - Corrupted file system
  - Incompatible kernel
- Solutions:
  - Make instance configuration changes
  - Reboot the instance

Viewing Status Checks Using the AWS CLI

```
aws ec2 describe-instance-status [--instance-ids i-1234567890]
```

Show instances with `impaired` status:

```
aws ec2 describe-instance-status --filters \
    Name=instance-status.status,Values=impaired
```

AMI Virtualization Types

Hardware Virtual Machine (HVM) AMIs

- Executes the master boot record of the root storage device
- Virtual hardware set allows for running an OS as if it were run on bare metal; the OS doesn't know it's virtualized
- No modification needed
- Can use hardware extensions
  - Provides fast access to host hardware
  - Enhanced networking and GPU processing

Paravirtual (PV) AMIs

- Runs a special boot loader and then loads the kernel
- Can run on hardware that does not support virtualization
- No hardware extension support
- PV historically performed faster than HVM, but that is no longer the case
- PV has special drivers for networking and storage that used less overhead than an HVM instance trying to emulate the hardware. These drivers can now be run on HVM instances, making the performance of both types the same.

NOTE: AWS now recommends using HVM instances because the **performance** is the same as PV, and **enhanced networking** and **GPU processing** can be utilized when necessary.

Next

EC2 Instance Types

- **General Purpose**
  - Balanced compute, memory, and network resources
  - Instance types:
    - **A1**, **T2**, **T3**, **M4**, **M5**
- **Compute Optimized**
  - Compute-intensive workloads
  - Instance types:
    - **C4**, **C5**
- **Memory Optimized**
  - Memory-intensive workloads
  - Large-scale or in-memory applications
  - Data mining
  - Instance types:
    - **R4**, **R5**, **X1**, **X1e**, **High Memory**, **z1d**
- **Accelerated Computing**
  - Greater GPU capacities
  - Instance types:
    - **P2**, **P3**: General-purpose GPU compute
    - **G3**: Graphics-intensive workloads
    - **F1**: Customizable hardware acceleration
- **Storage Optimized**
  - High data throughput
  - Instance types:
    - **H1**: Up to 16 TB of local HDD storage
    - **D2**: Up to 64 TB of local HDD storage
    - **I3**: NVMe SSD-backed (high IOPS at low cost)
    - This category also includes "bare metal" instances
      - Direct access to physical processors and memory
      - Non-virtualized workloads

Instance Sizes

- Instance size determines the:
  - Number of virtual CPUs
  - Memory
  - Networking performance
  - Clock speed
- Larger instance sizes provide better network performance
  - Some instance types offer enhanced networking
  - Some types offer 10 Gb network connectivity
- Jumbo frames (larger than 1500 MTU) support depends on instance type

Back

Auto Scaling
- Distributes the load across multiple instances
- Uses metrics and rules to automate spinning up/terminating instances
- Auto Scaling can scale or shrink on a schedule:
  - One-time occurrence or recurring schedule
  - Can define a new minimum, maximum, and scaling size
  - Lets you scale out before you actually need capacity in order to avoid delays

Changing Instance Sizes
- Increases/decreases resources available to our application

When to choose one over the other? ... They both have pros and cons.

Example Scenarios

Challenges of Auto Scaling
- Auto Scaling is relatively complicated to set up:
  - Instances can be started and stopped at any time
  - Applications need to be designed to handle distributed work
  - Important data (sessions, images, etc.) needs to be stored in a central location
  - If one server terminates, the application should still function
- Delays in scaling:
  - Instances take time to initialize
  - Applications may require setup, which could take even more time

Challenges of Resizing Instances
- Compatibility: Instances must have the same virtualization type to resize
- EBS-backed instances need to be stopped before resizing
- Instance store-backed instances require migration
- Resizing isn't very flexible compared to Auto Scaling
- There usually has to be downtime and careful planning
- Resizing instances in Auto Scaling groups may need "suspending"

Auto Scaling

- Distributes the load across multiple instances
- Uses metrics and rules to automate spinning up/terminating instances
- Auto Scaling can scale or shrink on a schedule:
  - One-time occurrence or recurring schedule

r to avoid

ons.

k
n a central

n

e time

to resize

- There usually has to be downtime and careful planning
- Resizing instances in Auto Scaling groups may need "suspending"

Auto Scaling vs. Resizing Instances

Scenario 1: Our PHP application is growing in terms of demand and needs to be highly available. It should scale with demand and shrink back down during slower times. It should also be able to withstand an Availability Zone going down. For these reasons, we've implemented Auto Scaling.

Should we also **resize instances**?

- We may not want to launch a lot of smaller instances if we can launch fewer larger ones.
- We could launch specialized instances to meet the needs of our application if we need more of one type of resource (compute optimized, for example).

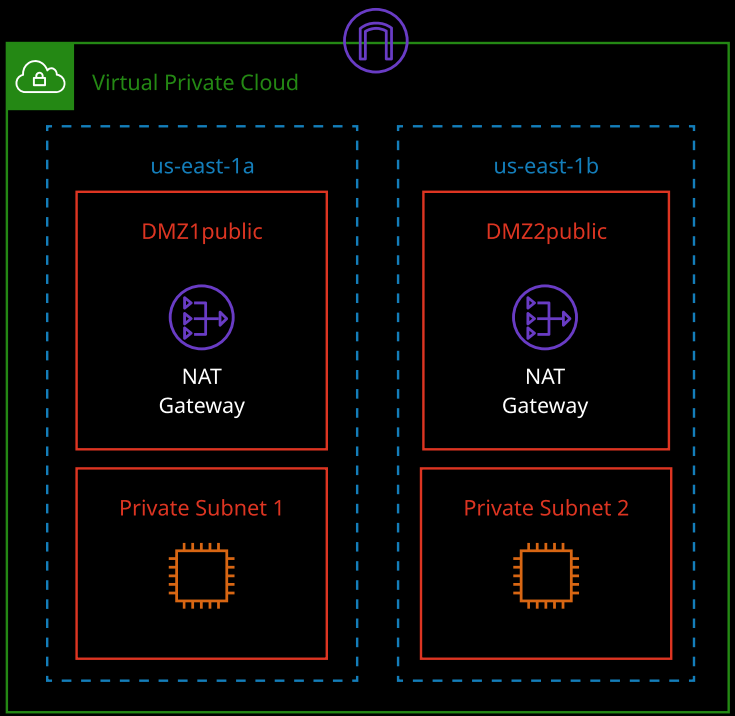
Scenario 2: We have an application that processes customer orders with the help of SQS. Orders are added to a queue, which is then polled by backend instances that process the orders. To meet capacity, we launch a certain number of instances. The issue is that sales change depending on the season, time of day, and day of the week.

Should we **Auto Scale**, **resize instances**, or **both**?

- Upgrading instance sizes to meet peaks in sales would leave us overpaying during slow periods.
- We can use Auto Scaling to check the queue length and adjust based off of that.
- Auto Scaling makes the most sense in this scenario.

NAT Gateways

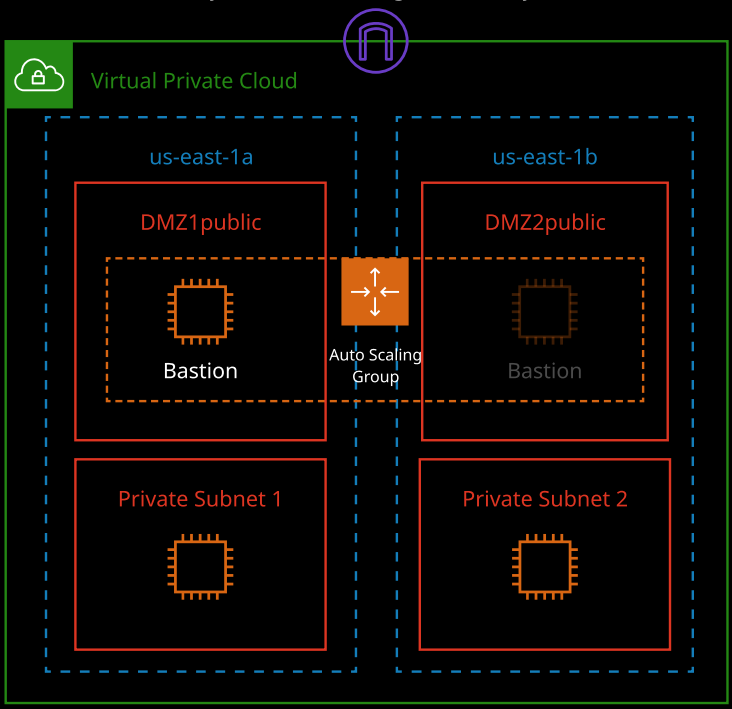
- Enable instances in a private subnet to access the internet for updates
- The instances in a private subnet are not accessible via the internet
- If updates/outside communication is business critical, consider using multiple NAT gateways



Back

Bastion Hosts

- "Gate" that protects our infrastructure but allows access for updates or other management
- Used to control remote access (e.g., via RDP or SSH)
- These should be hardened and secured very carefully and regularly
- Can have an Elastic IP address that never changes and can be whitelisted
- We can have standby bastion hosts for higher availability



Next

EC2: Reserved Instances

- Reserved Instances give us the ability to purchase instance capacity for a specific period of time (12 or 36 months).
- We can choose Standard Reserved Instances or Scheduled Reserved Instances.
  - Offer discounts
  - Reserve capacity

Example Scenarios

Reserved Instance Marketplace

- If requirements or needs change, we can sell Reserved Instances in the marketplace.
- Sellers can avoid wasting capacity and money.
- Buyers can get shorter terms.

Amazon RDS and ElastiCache

- Reserved capacity is also available for Amazon RDS instances and ElastiCache nodes.
- New generations of Reserved Cache Nodes only offer Heavy Utilization nodes, while older generations offer Heavy, Medium, and Light Utilization.

EC2: Reserved Instances

- Reserved Instances give us the ability to purchase instance capacity for a

ed

n the

ElastiCache

ation nodes,

Reserved Instance Scenarios

Scenario 1: A company is using large T2 instances but is expecting consistent growth and needs to upgrade to M4 instances towards the end of the year. M4 instances will put the company over budget, but they know that they'll be able to use that instance type for at least 3 years. What can they do?

Solution: They could purchase Reserved Instances.

Explanation: Reserved M4 instances purchased under a 3-year term could offer significant discounts. Even if the company needs to change instance sizes, as long as they are still M4 instances running non-licensed Linux platforms, they can change their Reserved Instances at no extra cost.

Scenario 2: We work for an e-commerce platform that loses $X per minute of downtime. We set up Auto Scaling for elasticity. One day, during our peak hour of sales, AWS returns the following error when Auto Scaling attempts to launch more instances: "InsufficientInstanceCapacity". This causes our instances to be overworked and miss requests. As a result, we lose a lot of sales. How can we avoid this in the future?

Solution: We could purchase Reserved Instances.

Explanation: When we purchase Reserved Instances, we're purchasing capacity. Even if we don't need it 100% of the time, it's there if we need it. That means we don't have to rely on AWS having enough On-Demand capacity. We could also purchase Scheduled Reserved Instances for peak hours.

Course Navigation

EC2: Initializing Volumes

- New EBS volumes operate at maximum performance as soon as they are available.
- For volumes restored from snapshots:
  - Maximum performance is not reached until all the blocks on the volume are read.
  - Must be initialized (reading all the blocks).
- Some utilities we can use:
  - **lsblk**
  - **dd**
  - **fio**

```
sudo dd if=/dev/xvdf of=/dev/null bs=1M

sudo fio --filename=/dev/xvdf \
    --rw=read \
    --bs=128k \
    --iodepth=32 \
    --ioengine=libaio \
    --direct=1 \
    --name=volume-initialize
```

EC2: Troubleshooting Autoscaling Issues

Common Issues:

- Attempting to use the wrong subnet
- Availability zone is no longer available or supported
- Security group does not exist
- Key pair associated does not exist
- Auto Scaling configuration is not working correctly
- Instance type specification is not supported in that Availability Zone
- Auto Scaling service is not enabled on the account
- Invalid EBS device mapping
- Attempting to attach EBS block device to an instance store AMI
- AMI issues
- Placement group attempting to use the wrong instance type
- “We currently do not have sufficient instance capacity in the AZ that you requested”
- Updating instance in Auto Scaling group with "suspended state"

Lightsail

- A VPS (Virtual Private Server) service
- Single instance
- Supported operating systems:
  - Amazon Linux
  - Debian
  - FreeBSD
  - OpenSUSE
  - Ubuntu
  - Windows Server
- One-click application deployments:
  - WordPress, Magento, Drupal, Joomla!, Redmine, Plesk
- Development stacks:
  - Node.js, GitLab, LAMP, MEAN, Nginx

Batch

- Runs batch computing workloads at scale
- Runs on containers
- Managed service:
  - Configures and manages the environment automatically
  - Batch provides the tools to control the scheduling and sequencing of jobs
- Components:
  - **Job**: Shell script, Linux executable, or Docker container image
  - **Job Definition**: Specifies how jobs are to be run
  - **Job Queue**: Jobs reside here while waiting for a compute environment
  - **Compute Environment**: Compute resources that run the jobs

S3 Cross-Region Replication

- Bucket-level configuration
- Enables automatic, asynchronous copying to a bucket in a different region
- Objects are replicated only once
- The following are retained by default:
  - Storage class
  - Object names
  - Owners
  - Permissions

Replication Configuration:

- Added to the source bucket
- Requires versioning enabled
- Contains:
  - Destination bucket
  - Objects identified for replication (can use key name prefixes, e.g., "folders")
  - Storage class for destination

What Is Not Replicated:

- Objects that existed before replication was activated
- Encrypted objects:
  - Server-side encryption using customer-provided keys (SSE-C)
  - Server-side encryption using KMS-managed keys (SSE-KMS) but can be explicitly enabled
- Objects that the owner does not have permissions to
- Lifecycle policies
- Objects that were replicated to the source bucket (can only be replicated once)

Data Storage

S3: Storage Classes

Every object stored in S3 has an associated storage class — also called a storage tier. Storage classes can be adjusted either manually or automatically using lifecycle policies. All storage classes have 99.999999999% (11 nines) durability.

Storage classes determine the cost of storage, the availability, durability, and latency for object retrieval. The current classes for S3 object storage are:

Standard
- Designed for general, all-purpose storage
- The default storage option
- Designed for 99.99% (four nines) availability
- 3+ AZ replication
- Most expensive storage class, but has no minimum object size and no retrieval fee

Intelligent-Tiering
- Moves objects across access tiers based on usage patterns
- Same performance as Standard

Standard Infrequent Access (Standard-IA)
- Designed for important objects, where access is infrequent, but rapid retrieval is a requirement
- Designed for 99.9% (three nines) availability
- 3+ AZ replication
- Cheaper than the Standard storage class
- 30-day minimum storage charge per object, 128 KB minimum storage charge, object retrieval fee

One Zone-IA
- Designed for non-critical, reproducible objects
- Designed for 99.5% availability
- 1 AZ replication (less resilient)
- Cheaper than the Standard or Standard-IA storage classes
- 30-day minimum storage charge per object, 128 KB minimum storage charge, object retrieval fee

Next

Back to Main

Glacier

- Designed for long-term archival storage (not to be used for hot backups)
- May take several minutes or hours for objects to be retrieved (several options available)
- Designed for 99.99% (four nines) availability
- 3+ AZ replication
- 90-day minimum charge per object, 40 KB minimum storage charge, object retrieval fee

Glacier Deep Archive

- Designed for long-term archival storage
- Ideal alternative to tape backups
- Cheaper than normal Glacier, but retrievals take longer
- Designed for 99.99% (four nines) availability
- 3+ AZ replication
- May take several hours for objects to be retrieved
- 180-day minimum charge per object, 40 KB minimum storage charge, object retrieval fee

Glacier Terminology

- Archive
  - A durably stored block of information
  - TAR and ZIP are common formats used to aggregate files
  - Total volume of data and number of archives is unlimited
  - Each archive can be up to 40 terabytes
  - Largest single upload is 4 gigabytes (use multipart upload >100 MB)
  - Archives can be uploaded and deleted, but not edited or overwritten
- Vault
  - A way to group archives together
  - Control access using vault-level access policies using IAM
  - SNS notifications available for when retrieval requests are ready for download
- Vault Lock
  - Lockable policy to enforce compliance controls on vaults
  - Vault lock policies are immutable

Back

Back to Main

Data Storage

AWS Storage Gateway

AWS Storage Gateway connects local data center software appliances to cloud-based storage, such as Amazon S3. We can use it for hybrid cloud backup, archiving and disaster recovery, tiered storage, application file storage, and data processing workflows.

File Gateway

- Comprises the S3 service and a virtual appliance
- Allows for storage and retrieval of files in S3 using standard file protocols (NFS and SMB)

Volume Gateway

- Gateway-Cached Volumes
  - Create storage volumes and mount them as iSCSI devices on the on-premises servers
  - The gateway will store the data written to this volume in Amazon S3 and will cache frequently accessed data on-premises in the storage device
- Gateway-Stored Volumes
  - Store all the data locally (on-premises) in storage volumes
  - Gateway will periodically take snapshots of the data as incremental backups and store them on Amazon S3

Tape Gateway

- A cloud virtual tape library that writes to Glacier
- Used for archiving data
- Can run as a VM on-premises or on an EC2 instance

Data Storage

AWS Snowball

Snowball is a petabyte-scale data transport solution that uses secure appliances to transfer large amounts of data into and out of the AWS cloud.

Snowball

- Suitcase-sized device used to transfer TB or PB of data into S3
- Connect to on-premises network with Snowball client
- 80 TB capacity
- 256-bit encryption using KMS
- Shipping and transfer times are typically one week
  - Typically used with >100 TB of data and bandwidth limitations
  - Example: 100 TB @ 1 Gbps = 100+ days to transfer. Done in ~1 week with two Snowballs.

Snowball Edge

- Larger capacity than Snowball
- Embedded computing capability using Lambda
- Used for IoT, media transcoding, and other use cases

Snowmobile

- Shipping container on a semi truck
- Up to 100 PB (1250 Snowballs)
- Multiple levels of logical and physical security
  - Encryption
  - Fire suppressions
  - Security personnel
  - GPS tracking
  - Video surveillance
  - Escort vehicle

Back to Main

Data Storage

Amazon EBS Essentials

Key Features

- Maximum volume size of **16 TiB**
- Volumes can only be mounted to **one** instance at a time
- **Multiple** volumes can be mounted to a single instance
- Raw, unformatted **block** storage
  - A file system must be created
  - RAID is supported if the OS supports it
- Created in an **Availability Zone** and must be used with instances in that zone
  - Automatically replicated in that zone
    - Protects against hardware failure, **not** high availability

Persistence

- Attached volumes are independent of the instance
  - Terminating the instance will not terminate the volume
- Root volumes, by default, terminate with the instance
  - This behavior is changed by setting `DeleteOnTermination` attribute to false

Snapshots

- Images or backups of EBS volumes
- Stored in an Amazon-managed S3 bucket (charged based on volume's total size)
- Exact copy of the original volume
  - Encryption included
- Incremental in nature, but full volume can be restored from any snapshot

Back to Main

Data Storage

EBS: Performance

EBS uses **IOPS** (input/output operations per second) as a performance measure, measured in KiB (kibibytes = 1024 bytes).

- **SSD:** Consistent performance with sequential and random operations
  - Good for frequent read/writes using small I/O sizes (IOPS)
  - IOPS measured with **256 KiB I/O size** (chunks of data)
    - One 1024 KiB operation = 4 IOPS
    - Four 64 KiB **sequential** operations = 1 IOPS
    - Four 64 KiB **random** operations = 4 IOPS

- **HDD:** Optimal performance with large and sequential operations
  - Good for large streaming workloads (throughput)
  - IOPS measured with **1024 KiB I/O size** (chunks of data)
    - One 1024 KiB operation = 1 IOPS
    - Eight 128 KiB **sequential** operations = 1 IOPS
    - Eight 128 KiB **random** operations = 8 IOPS

Burst Buckets

- Allows an EBS volume to "burst" above the baseline performance
  - Volumes earn **credits**
  - Credits are then spent whenever the volume needs more performance
  - There is a maximum number of credits
- Not available for Provisioned IOPS SSD (io1)
- Reported as a `BurstBalance` metric in CloudWatch

Join multiple gp2, io1, st1, or sc1 volumes together in a RAID 0 configuration (stripe set) to use the available bandwidth, improving throughput.

SSD Volumes HDD Volumes

Course Navigation

EBS uses **IOPS** (input/output operations per second) as a performance measure, measured in KiB (kibibytes = 1024 bytes).

SSD Volumes

- **General Purpose SSD Volumes (gp2)**
  - Volume size: 1 GiB to 16 TiB
  - Baseline performance: 100 to 16,000 IOPS
    - 3 IOPS per GiB of volume size
      - Minimum of 100 IOPS (below 33.3 GiB volume size)
  - Maximum throughput = 250 MiB/s
  - Burst Bucket
    - 5.4 million credits to start (max 3,000 IOPS for 30 minutes)
    - Credits earned at 3 IOPS per GiB of volume size
      - Maximum credits = 5.4 million
    - Volumes greater than 1,000 GiB never deplete (baseline = burst max)
- **Provisioned IOPS SSD volumes (io1)**
  - Volume size: 4 GiB to 16 TiB
  - Baseline performance: 100 to 64,000 IOPS
    - User/provisioner chooses a consistent IOPS rate
    - AWS SLA = 99.9 of the time within 10% of provisioned IOPS
    - Maximum ratio is 50:1
      - A 640 GiB volume size or greater can use maximum IOPS
    - AWS recommends a ratio larger than 2:1 with provisioned IOPS volumes
      - If 4,000 IOPS are needed, volume size should be less than 2,000 GiB
  - Maximum throughput = 1,000 MiB/s

Course Navigation

EBS uses **IOPS** (input/output operations per second) as a performance measure, measured in KiB (kibibytes = 1024 bytes).

HDD Volumes

- **Throughput Optimized HDD volumes (st1)**
  - Not supported as a boot device
  - Ideal for frequently accessed and throughput intensive workloads
  - Volume size: 500 GiB to 16 TiB
  - Maximum throughput = 500 MiB/s
  - Burst bucket
    - Credits gained at 40 MiB/s per TiB
    - Credit capacity = 1 TiB
    - Maximum burst = 500 MiB/s (volume sizes 2 TiB and larger)

- **Cold HDD volumes (sc1)**
  - Not supported as a boot device
  - Ideal for infrequently accessed data and lowest storage cost
  - Volume size: 500 GiB to 16 TiB
  - Maximum throughput = 250 MiB/s
  - Burst bucket
    - Credits gained at 12 MiB/s per TiB
    - Credit capacity = 1 TiB
    - Maximum burst = 250 MiB/s (volume sizes 2 TiB and larger)

- **Magnetic volumes**
  - Somewhat deprecated (Previous Generation volume)
  - Low-cost storage for small volume sizes
  - Volume Size: 1 GiB to 1 TiB
  - Burst capability to hundreds of IOPS

CloudWatch Metrics

- Five-minute period data available at no charge
- Provisioned IOPS SSD send one-minute period data automatically
- Included metrics measure disk management
  - `VolumeReadBytes`
  - `VolumeWriteBytes`
  - `VolumeReadOps`
  - `VolumeWriteOps`
  - `VolumeTotalReadTime`
  - `VolumeTotalWriteTime`
  - `VolumeIdleTime`
  - `VolumeQueueLength`
- Provisioned IOPS SSD (io1) volumes only:
  - `VolumeThroughputPercentage`
  - `VolumeConsumedReadWriteOps`
- General Purpose SSD (gp2), Throughput Optimized HDD (st1), and Cold HDD (sc1) volumes only:
  - `BurstBalance`

EBS Status Checks

- Tests run every five minutes
- Returns: OK, warning, impaired, insufficient-data
- User can change the result of the impaired response

Modify the size, IOPS, or type of an EBS volume

The Manual Method:

1. Modify the EBS volume.
2. Extend the partition to fill available space.
3. Expand the filesystem in the resized partition.

For Nitro-based instances (e.g., t3.micro):

```
lsblk
df -h
sudo file -s /dev/nvme?n*
sudo growpart /dev/nvme0n1 1
sudo xfs_growfs -d /
```

For T2 instances:

```
sudo file -s /dev/xvd*
sudo growpart /dev/xvda 1
sudo resize2fs /dev/xvda
```

The "Automated" Method:

- We can replace the launch configuration of an Auto Scaling group.
- In a true n-tier application that is decoupled, we should then be able to terminate instances one by one to recreate them using the new configuration.

Ensuring data durability upon EC2 instance termination

- By default, instance store and EBS **root** volumes are not backed up
- Will not persist upon termination

- Cannot stop instance store volumes, so termination is the only option
- This is why EBS volumes are recommended

- How do we save the data on a root volume?
  - Uncheck "Delete on Termination" in the console:
    - Also a CLI parameter with run-instances
  - Create a snapshot before deletion
  - We can create a separate volume and attach to the instance:
    - **Attached** volumes persist when the instance is terminated

Data Storage

Amazon EFS: Deployment and Provisioning

- Highly available, scalable file system:
  - Spans multiple Availability Zones
  - Throughput for parallel workloads:
    - Big Data, Analytics, Media Processing, Content Management, Web Serving
- Shared data store that can be mounted to multiple EC2 instances or on-premises servers:
  - For on-premises servers, use AWS Direct Connect or AWS VPN
- Linux-only; Windows is not supported
- Two performance modes:
  - **General Purpose**: Most file system needs
  - **Max I/O**: Cases where hundreds or more instances access the file system
    - Scales throughput and IOPS (slightly higher latencies)
- Bursting:
  - Burst to 100 MiB/s for any size file system
  - Larger than 1 TiB = bursting 100 MiB/s per TiB of data stored
  - Credit system: Earns credits at 50 MiB/s per TiB of data stored
- Security groups should be used to control NFS traffic
  - Use the EC2 security group as the source
- Supports encryption at rest and in transit
- Storage classes and lifecycle management:
  - Standard
  - Infrequent Access (IA)
  - Lifecycle management automatically moves files to IA not accessed for 30 days

Course Navigation

Data Storage
Section 3

S3: Cross-Region Replication
S3: Storage Classes
AWS Storage Gateway
AWS Snowball
Amazon EBS Essentials
EBS: Performance
EBS: Metrics
EBS: Resizing or Changing Root Volumes
EBS: Ensuring Data Durability
Amazon EFS: Deployment and Provisioning
EFS: Monitoring for Performance and Availability
RDS: Scaling for Performance

Networking
Section 4

Back to Main

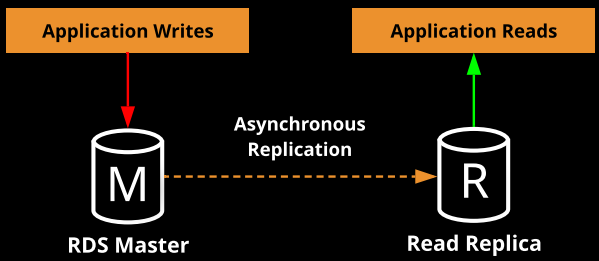
CloudWatch Metrics

- `BurstCreditBalance`
- `ClientConnections`
- `DataReadIOBytes`
- `DataWriteIOBytes`
- `MetadataIOBytes`
- `PercentIOLimit`
- `PermittedThroughput`
- `TotalIOBytes`

EFS metric data is sent to CloudWatch at one-minute intervals and retained for 15 months.

RDS: Scaling for Performance

- Read replicas can be used to offload work from the main database:
  - Writes go to the source instance.
  - Reads go to the read replica(s).
- Replication to Read Replicas is made asynchronously (not at the same time) .
- Data is written to the source instance and then replicated to the read replica(s).



Scenario: You need to pull data for analysis, but you don't want to degrade performance on your production database.
Solution: Create a read replica that's only used for this reason.

AWS RDS Read Replication vs. Multi-AZ Failover Deployments

- Read replicas are built primarily for performance and offloading work.
- Multi-AZ deployments are used for high availability and durability.
- Multi-AZ deployments give us synchronous replication instead of asynchronous.
- Multi-AZ deployments are only used to perform a failover; they are idle the rest of the time.
- Read replicas are used to serve legitimate traffic.
- It is often beneficial to use both of these as complements.

Course Navigation

Networking
Section 4

VPC Essentials
VPC Flow Logs
VPC Peering
AWS VPN
AWS Direct Connect
EC2: Elastic IP (EIP) and Elastic Network Interfaces (ENI)
ELB: Monitoring for Performance and Availability
Amazon ELB: High Availability
ELB: SSL Offloading
Network Bottlenecks
Amazon CloudFront

Databases
Section 5

What Is a Virtual Private Cloud?

- A VPC resembles private data centers or corporate networks
- Private and public subnets
- Scalable infrastructure
- Ability to extend corporate/home network to the cloud as if it were part of your network

VPC Implementation:

- Logically isolated from other networks on AWS
- VPCs can't span regions
- Size can range from /16 to a /28 netmask (65,536 to 16 IP addresses)
- Subnets can't span Availability Zones

Benefits of a VPC:

- Ability to launch instances into a subnet
- Ability to define custom IP address ranges inside of each subnet (private and public subnets)
- Ability to configure route tables between subnets
- Ability to configure internet gateways and attach them to subnets
- Ability to create a layered network of resources
- Extending our network with VPN/VPG controlled access
- Ability to use security groups and subnet network ACLs

Understanding the Default VPC:

- Size /16 CIDR block (172.31.0.0/16)
- Default subnet in each AZ using /20 subnet mask
- Internet gateway
- Main route table sending all IPv4 traffic for 0.0.0.0/0 to the internet gateway
- Default security group allowing all traffic
- Default network ACL (NACL) allowing all traffic
- Default DHCP option set

Example VPC
Next

Back to Main

Course Navigation

Networking
Section 4

VPC Essentials
VPC Flow Logs
VPC Peering
AWS VPN
AWS Direct Connect
EC2: Elastic IP (EIP) and Elastic Network Interfaces (ENI)
ELB: Monitoring for Performance and Availability
Amazon ELB: High Availability
ELB: SSL Offloading
Network Bottlenecks
Amazon CloudFront

Databases
Section 5

Back to Main

VPC Scenarios:

- VPC with public subnet only: Single-tier apps
- VPC with public and private subnets: Resources that don't need public internet access/layered apps
- VPC with public and private subnets and hardware-connected VPN: Extending to on-premises
- VPC with a private subnet only and hardware VPN access

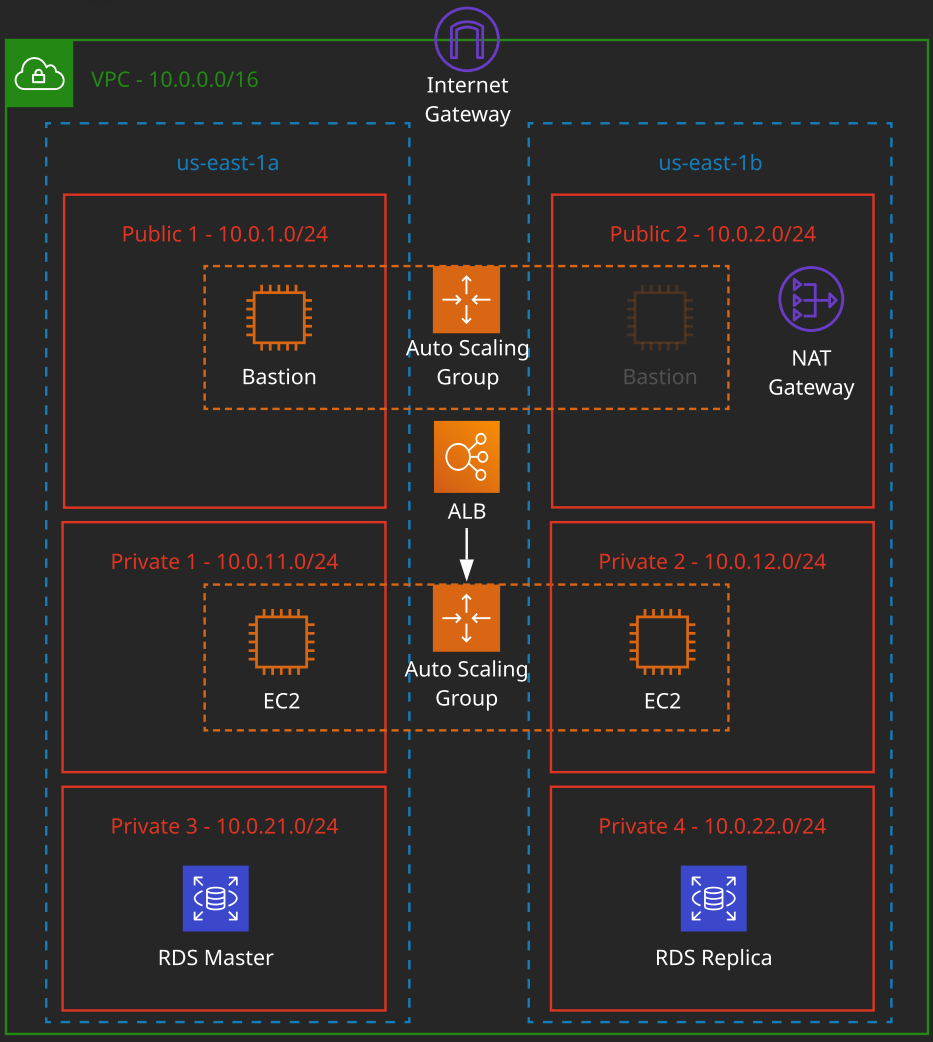
VPC IP Reservations:

AWS reserves the first four IP and the last IP addresses. In a 10.0.0.0/24, the following IPs are reserved:

- **10.0.0.0:** Network address
- **10.0.0.1:** Reserved by AWS for the Amazon VPC router
- **10.0.0.2:** Reserved by AWS. The IP address of the DNS server is always the base of the Amazon VPC network range; however, the base of each subnet range is also reserved.
- **10.0.0.3:** Reserved by AWS for future use
- **10.0.0.255:** Network broadcast address. AWS does not support broadcast in an Amazon VPC; therefore, they reserve this address.

Back
Example VPC

Example VPC


VPC - 10.0.0.0/16
Internet Gateway
us-east-1a
us-east-1b
Public 1 - 10.0.1.0/24
Public 2 - 10.0.2.0/24
Bastion
Auto Scaling Group
Bastion
NAT Gateway
ALB
Private 1 - 10.0.11.0/24
Private 2 - 10.0.12.0/24
EC2
Auto Scaling Group
EC2
Private 3 - 10.0.21.0/24
Private 4 - 10.0.22.0/24
RDS Master
RDS Replica

Course Navigation

Networking
Section 4

VPC Essentials
VPC Flow Logs
VPC Peering
AWS VPN
AWS Direct Connect
EC2: Elastic IP (EIP) and Elastic Network Interfaces (ENI)
ELB: Monitoring for Performance and Availability
Amazon ELB: High Availability
ELB: SSL Offloading
Network Bottlenecks
Amazon CloudFront

Databases
Section 5

VPC Flow Logs allows you to capture metadata about IP traffic going in and out of your network interfaces.

- Data can be stored in CloudWatch Logs or S3
  - Each network interface has a unique log stream
- Can be created for a VPC, subnet, or network interface
  - When choosing VPC, each network interface and subnet in that VPC is monitored
- Flow log records consist of fields describing the traffic for that network interface
- For security reasons, EC2 instances can't receive or *sniff* traffic destined for a different instance
- Delay of several minutes — flow logs do **not** capture real-time log streams
- Ability to create multiple flow logs per interface (e.g., accepted vs. rejected traffic)
- Launching new EC2 instances *after* creating flow logs will automatically create logs for each new network interface
- Create flow logs for network interfaces created by other AWS services:
  - Elastic Load Balancing, Amazon RDS, Amazon ElastiCache, Amazon Redshift, Amazon WorkSpaces

Flow Log Record Syntax:

- version
- account-id
- interface-id
- srcaddr
- dstaddr
- srcport
- dstport
- protocol
- packets
- bytes
- start
- end
- action
- log-status

What's NOT Logged:

- Amazon DNS server traffic
- Amazon Windows license activation
- Instance metadata to/from 169.254.169.254
- Amazon Time Sync to/from 169.254.169.123
- DHCP traffic
- Traffic to/from the default VPC router reserved IP address
- Traffic between an endpoint network interface and a Network Load Balancer network interface

Networking

VPC Peering

VPC peering allows you to set up **direct network routing** between different VPCs using **private IP addresses**.

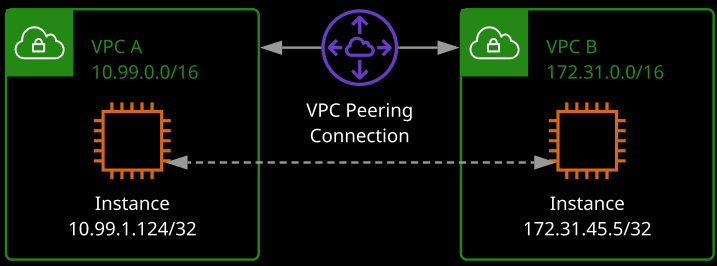
- Instances will communicate with each other as if they were on the same private network.
- VPC peering can occur between different AWS accounts as well as VPCs in other regions using Inter-Region VPC Peering.
- All inter-region traffic is encrypted.
- Traffic remains on the global AWS backbone.

Scenarios:

- Peering two VPCs: Company runs multiple AWS accounts and you need to link all the resources as if they were all under one private network
- Peering **to** a VPC: Multiple VPCs connect to a central VPC, but they can only communicate with the central VPC (file sharing, customer access, Active Directory) and not each other.

Limitations:

- Can't peer VPC with matching or overlapping CIDR blocks
- VPC peering connections are 1:1 between VPCs — transitive peering is **not** supported *(see Transit Gateway)*
- One peering connection between the same two VPCs
- Tags applied to the peering connection are only applied in the account and region in which you create them
- Security groups can't reference peer VPC security groups across regions
- IPv6 across regions is not supported
- DNS resolution for private hostnames must be enabled manually
  - If in different accounts, must be enabled in both accounts



Back to Main

Course Navigation

Networking
Section 4

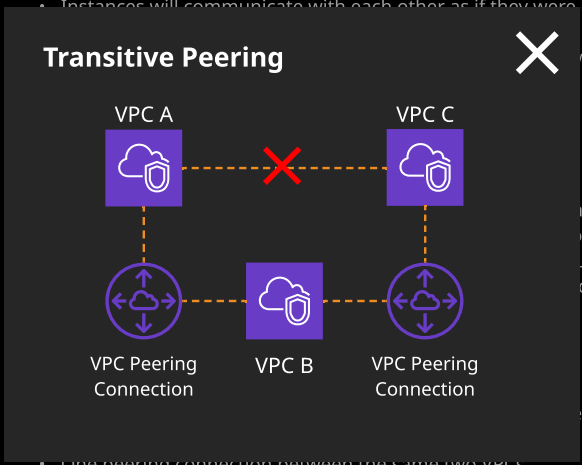
VPC Essentials
VPC Flow Logs
VPC Peering
AWS VPN
AWS Direct Connect
EC2: Elastic IP (EIP) and Elastic Network Interfaces (ENI)
ELB: Monitoring for Performance and Availability
Amazon ELB: High Availability
ELB: SSL Offloading
Network Bottlenecks
Amazon CloudFront

Databases
Section 5

Back to Main

VPC peering allows you to set up **direct network routing** between different VPCs using **private IP addresses**.

- Instances will communicate with each other as if they were on the same private

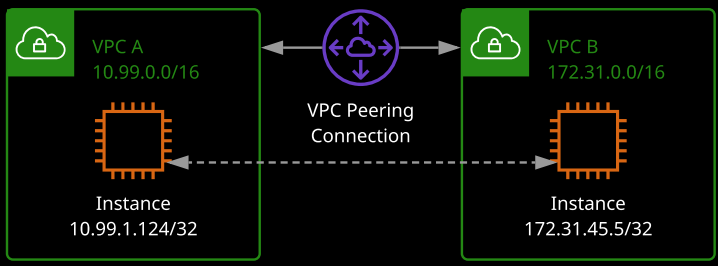


well as VPCs in other

d you need to link
rk
ut they can only
ccess, Active

peering is **not**

- One peering connection between the same two VPCs
- Tags applied to the peering connection are only applied in the account and region in which you create them
- Security groups can't reference peer VPC security groups across regions
- IPv6 across regions is not supported
- DNS resolution for private hostnames must be enabled manually
  - If in different accounts, must be enabled in both accounts



Networking

AWS VPN

Scenario:

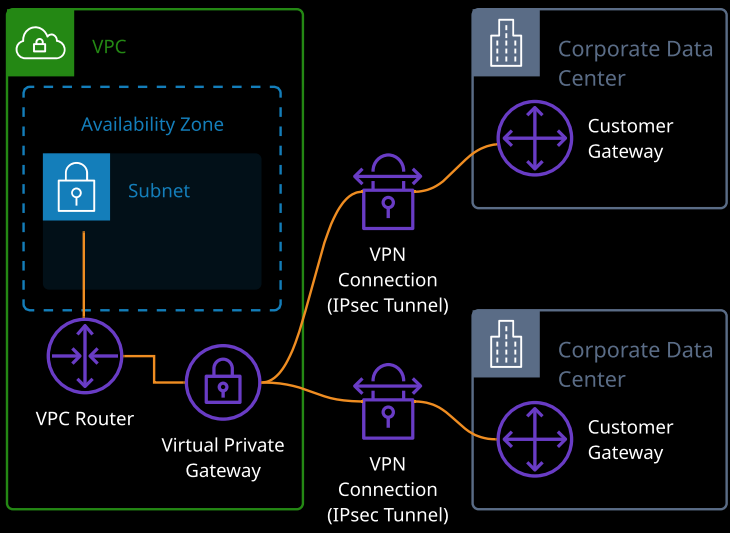
- Your organization requires secure communications
- Lesser need for dedicated throughput (e.g., AWS Direct Connect)
  - VPN transits public internet

Components:

- Customer gateway (initiates the VPN connection)
- Virtual private gateway
  - One per VPC - used with IPsec and AWS Direct Connect
- VPN connection (two IPsec tunnels)

Best Practice:

- Deploy VPN using standard AWS VPN components (VPN gateway, customer gateway, VPN connection)
- Can also use custom VPN solutions if required (software VPN on AWS Marketplace)
- Ensure VPC networking (subnets, security groups, NACLs) is secure



Networking

AWS Direct Connect

Dedicated link from your internal network to AWS:

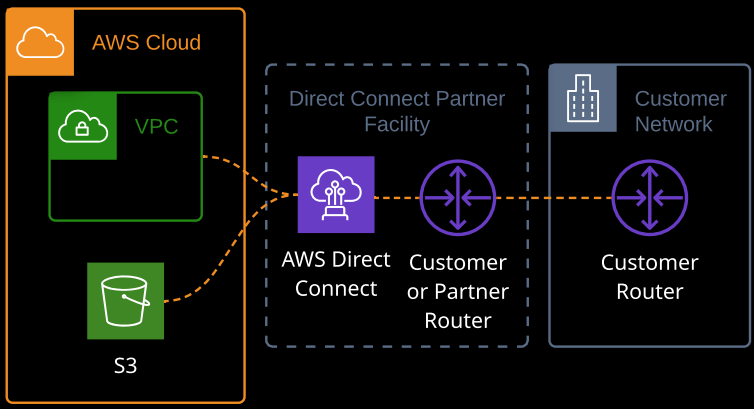
- Dedicated throughput
- Provide more consistent network performance
- Reduce bandwidth costs
- Private connection to AWS
- Elasticity and scaling - provision multiple 1 Gbps and 10 Gbps connections

For **dedicated connections**, DX requires **single-mode fiber:**

- 1 Gbps: 1000BASE-LX (1310nm)
- 10 Gbps: 10GBASE-LR (1310nm)

Best Practice:

- Using a private peered connection might not need extra security
- Check your organization's requirements
- VPC networking (subnets, security groups, NACLs)
- Avoid VPN hardware that can't support high data transfer rates (>4 Gbps)
- **Note**: Direct Connect (DX) is **not** highly available by default.
  - It is recommended to use multiple DX connections in different AWS regions.



Linux Academy

Course Navigation

Networking
Section 4

VPC Essentials
VPC Flow Logs
VPC Peering
AWS VPN
AWS Direct Connect
EC2: Elastic IP (EIP) and Elastic Network Interfaces (ENI)
ELB: Monitoring for Performance and Availability
Amazon ELB: High Availability
ELB: SSL Offloading
Network Bottlenecks
Amazon CloudFront

Databases
Section 5

Back to Main

EC2: Elastic IP (EIP) and Elastic Network Interfaces (ENI)

Elastic IPs (EIPs):

- A public IP address that can be "moved"
- Enables instances without a public IP to become accessible from the internet
- Good to know:
  - EIPs are region specific
  - IPv6 not currently supported
  - Two-step process to implement: allocation and association
  - Upon association, any previous public IP is released (DNS hostname changes as well)
  - Can be disassociated and reassociated with another instance
  - Two-step process to remove: disassociate and release
- You are charged for:
  - Elastic IPs not associated
  - More than one Elastic IP on an instance

Elastic Network Interfaces (ENIs):

- Virtual network card that can include:
  - Primary and secondary private IPv4 addresses
  - An Elastic IP address (IPv4)
  - A public IPv4 address
  - Public IPv6 addresses
  - Security groups
  - MAC address
  - Description
- Good to know:
  - Every instance in a VPC has a default network interface, called the **primary network interface** (eth0). You cannot detach a primary network interface from an instance.
  - When detaching and reattaching to instances, the attributes and traffic follow the interface.
  - You can modify attributes after creation (security groups and IP addresses).

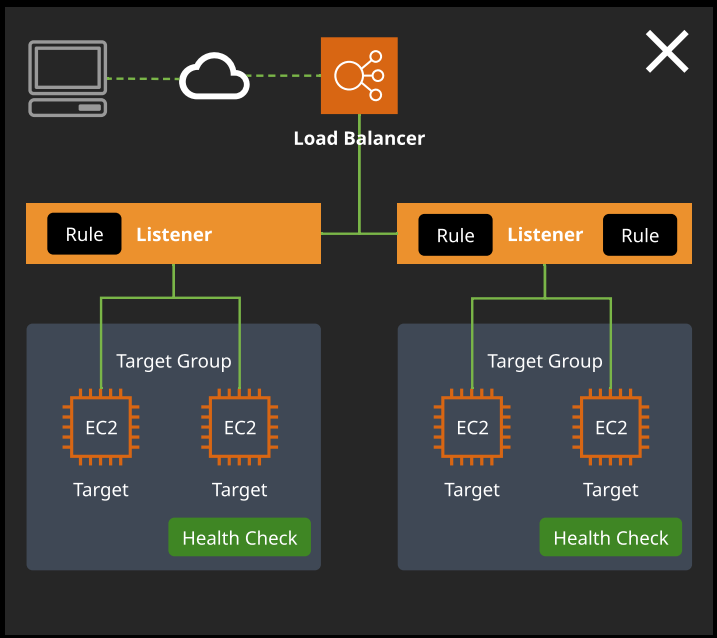
Course Navigation

Networking
Section 4

- **VPC Essentials**
- **VPC Flow Logs**
- **VPC Peering**
- **AWS VPN**
- **AWS Direct Connect**
- **EC2: Elastic IP (EIP) and Elastic Network Interfaces (ENI)**
- **ELB: Monitoring for Performance and Availability**
- Amazon ELB: High Availability
- ELB: SSL Offloading
- Network Bottlenecks
- Amazon CloudFront

Databases
Section 5

Back to Main



Application Load Balancer

Diagram

- Three components
  - **Load Balancer**
    - Receives client requests (HTTP, HTTPS)
  - **Listeners**
    - Reads the requests from clients
    - Compares the request to **rules**, then forwards to a target group
  - **Target Group**
    - Receives forwards from listeners
    - **Health checks** are configured per target group
    - Targets can be in multiple target groups

- Works at the Application layer (7)
- Content-based routing
  - **Path-based routing**: Forwards based on the URL in the request
    - */dev* and */prod* can route to different target groups
  - **Host-based routing**: Forwards based on the host field of the HTTP header
    - *dev.mysite.com* and *prod.mysite.com* can route to different target groups
- Routes to IP addresses, including outside the VPC (on-premises)
- Routes to microservices (allows dynamic port mapping)

- **Monitoring**
  - CloudWatch metrics
    - *ActiveConnectionCount*, *HealthyHostCount*, HTTP code totals, and more
  - Access logs: Sends detailed request information to S3
  - Request tracing: A header is added that includes a trace identifier for requests
  - CloudTrail logs: Records API activity

Next: Network Load Balancer

Back to Main

Network Load Balancer

- Same three components
  - **Load Balancer**
    - Receives client requests
  - **Listeners**
    - Reads the requests from clients
    - Compares the request to rules, then forwards to a target group
  - **Target Group**
    - Uses **TCP** protocol and port to route requests to targets (EC2, on-premises)
    - Health checks are configured per target group
    - Targets can be in multiple target groups

- Functions at the Transport layer (4)
- Millions of connections capability (no pre-warming needed)
- Each Availability Zone assigned gets a node created in it with a static IP (or EIP)
  - Reduces latency
- Register targets by:
  - Instance ID: Source addresses of clients are preserved
  - IP address: Source addresses of clients are the private IP of NLB node
- Client TCP connections have different source port and sequence numbers
  - Route traffic to different targets
- Change targets anytime

- **Monitoring**
  - CloudWatch metrics:
    - *ActiveFlowCount, HealthyHostCount, UnhealthyHostCount,* and more
  - VPC Flow Logs: Detailed log of traffic going to and from your NLB
  - CloudTrail logs: Records API activity

Course Navigation

Networking
Section 4

VPC Essentials
VPC Flow Logs
VPC Peering
AWS VPN
AWS Direct Connect
EC2: Elastic IP (EIP) and Elastic Network Interfaces (ENI)
ELB: Monitoring for Performance and Availability
Amazon ELB: High Availability
ELB: SSL Offloading
Network Bottlenecks
Amazon CloudFront

Databases
Section 5

Back to Main

Classic Load Balancer

- A simple, no-frills load balancer

- Supports EC2-Classic
- Supports HTTP, HTTPS, TCP, and SSL listeners

- Cross-zone load balancing
  - Enable to evenly distribute traffic to all registered instances
  - Recommended to keep roughly the same number in each AZ

- **Monitoring**
  - CloudWatch metrics:
    - *HealthyHostCount*, *RequestCount*, *Latency*, HTTP codes, and more
  - Access logs: Sends request information to S3
  - CloudTrail logs: Records API activity

Back

Course Navigation

Networking
Section 4

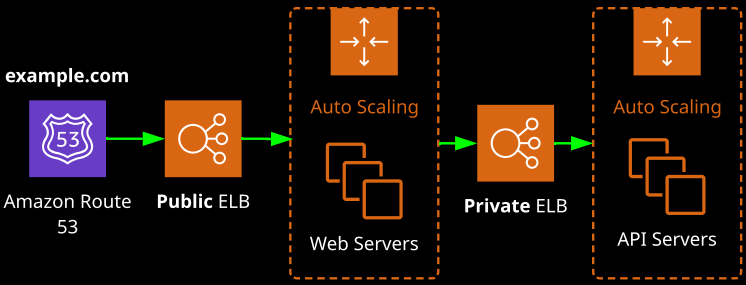
VPC Essentials
VPC Flow Logs
VPC Peering
AWS VPN
AWS Direct Connect
EC2: Elastic IP (EIP) and Elastic Network Interfaces (ENI)
ELB: Monitoring for Performance and Availability
Amazon ELB: High Availability
ELB: SSL Offloading
Network Bottlenecks
Amazon CloudFront

Databases
Section 5

Back to Main

Elastic Load Balancer

- We can have both external and internal load balancers.
- External load balancers are public facing:
  - Often used to distribute load between web servers
  - Provide a public DNS hostname
- Internal load balancers are not customer facing:
  - Often used to distribute load between private back-end servers
  - Provide an internal DNS hostname



Sticky Sessions

- Maintains a user's session state by ensuring they are routed to the same target
- Uses cookies to identify sessions. Clients must support them.
- Enabled on the target group (Application and Network Load Balancers)
- Enabled on the Classic Load Balancer itself after creation

Course Navigation

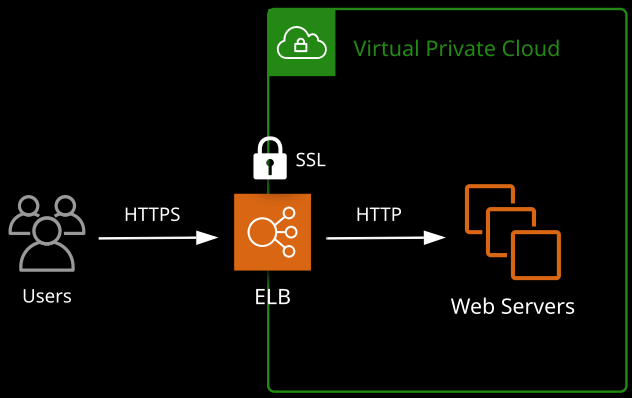
Networking
Section 4

VPC Essentials
VPC Flow Logs
VPC Peering
AWS VPN
AWS Direct Connect
EC2: Elastic IP (EIP) and Elastic Network Interfaces (ENI)
ELB: Monitoring for Performance and Availability
Amazon ELB: High Availability
ELB: SSL Offloading
Network Bottlenecks
Amazon CloudFront

Databases
Section 5

Back to Main

- In a highly available web application, we use load balancers to distribute traffic.
- We can also use their elasticity and scalability in the HTTPS/SSL process.
- Encryption and decryption require processing:
  - We can save processing on our instances by transferring the SSL process to the load balancer.
- There is no need for every EC2 instance to need a certificate and process encryption and decryption:
  - Application performance should increase.
- Certificate Manager also integrates for certificate generation and management.



Course Navigation

Networking
Section 4

VPC Essentials
VPC Flow Logs
VPC Peering
AWS VPN
AWS Direct Connect
EC2: Elastic IP (EIP) and Elastic Network Interfaces (ENI)
Amazon ELB: High Availability
ELB: SSL Offloading
ELB: Monitoring for Performance and Availability
Network Bottlenecks
Amazon CloudFront

Databases
Section 5

Back to Main

Potential Networking Issues

- One of the primary network bottlenecks comes from EC2 instances
- Potential causes for bottlenecks:
  - Instances are in different Availability Zones, regions, or continents
  - EC2 instance sizes (larger instances generally have better bandwidth performance)
  - Not using enhanced networking features
  - We can check network performance with iperf3
- VPCs can use VPC peering to create a reliable connection:
  - No single point of failure for communication or bandwidth bottlenecks
  - Peer VPCs between regions to avoid traffic transiting the public internet

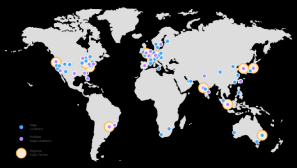
Bandwidth Limitations on Your VPN to Your AWS VPC

- Using a VPN to access our AWS VPC from our on-premises network means we have to communicate over the open internet:
  - Bandwidth, latency, consistency, and reliability issues
- Use **AWS Direct Connect**

Networking

Amazon CloudFront

AWS Global Content Delivery Network (CDN):
- Low latency
- High transfer speeds from the origin



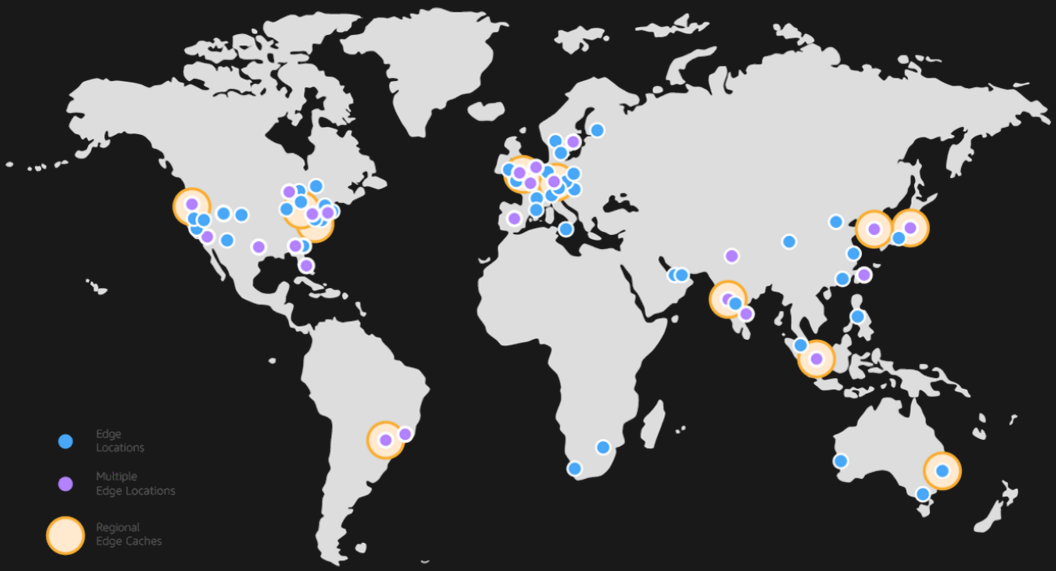
Click map to enlarge

Components:
- Origin:
  - The original version of your content
  - Can be an S3 bucket or a web server
- Distribution:
  - Points edge locations and regional caches back to the origin
  - Configuration of logging, availability, and limitations
- Edge Locations:
  - The location of your cached objects, located all over the globe
  - Current total is 169 in 30 countries
- Regional Edge Caches:
  - Location of cached objects that are not as frequently accessed
  - Current total is 11 in 30 countries

Process:
- When an edge location receives a request, it checks to see if it is cached locally
- If so, the content is delivered
- If not, the edge location can query the regional edge cache or the origin
- When receiving these requested objects, the edge location immediately starts to forward to the end user

Making Changes to Content:
- You can simply delete from the origin and wait for content at the edge locations to reach the expiration period
- You can invalidate content to have it removed before the expiration, but it does cost more

CloudFront Edge Locations

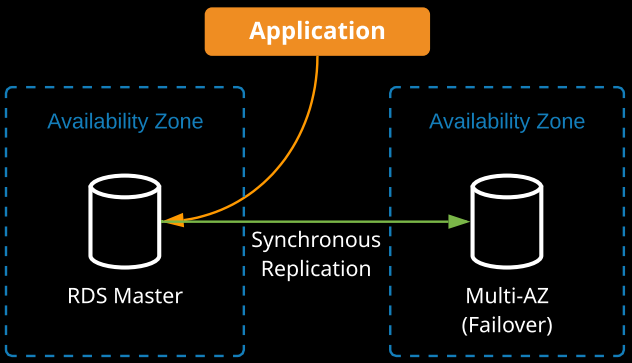


https://aws.amazon.com/cloudfront/features/

Back

Multi-AZ

- Provisions and maintains a standby replica in a different AZ
- The primary synchronously replicates to the standby instance for redundancy
- Can reduce downtime in the event of a failure on the primary



- The feature can be turned on from the console or API
- Amazon automatically handles replication
- Replication can cause higher write latency:
  - Using Provisioned IOPS is recommended

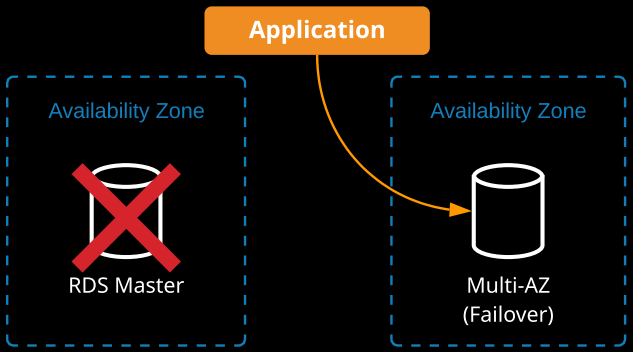
- Maintenance
  - AWS will perform the following steps:
    - Perform maintenance on the standby
    - Promote the standby
    - Perform maintenance on the old primary, now the standby

Next

Amazon RDS: Understanding Multi-AZ Deployments

What Can Trigger a Failover?

- Loss of availability in the primary Availability Zone
- Loss of network connectivity to the primary instance
- Resource failure with the underlying virtualized resources
- Storage failure on the primary database
- The DB instance's server type is changed
- Maintenance



How Do Failovers Work?

- The process is automated by AWS:
  1. Amazon detects an issue and starts the failover process.
  2. DNS records are modified to point to the standby instance.
  3. The application re-establishes any existing DB connections.
- The application requires no changes since the DB endpoint is the same.

How Do We Know When a Failover Happens?

- Use RDS events to notify via email or SMS.
- Use the API or console to manually check events.
- Use the API or console to check the state of the Multi-AZ deployment.

Back

Back to Main

RDS: Monitoring for Performance and Availability

- Managed database web service
  - AWS manages patching, backups, detecting failures, and recovery
- Supports these engines:
  - MySQL, MariaDB, PostgreSQL, Oracle, Microsoft SQL Server, and Amazon Aurora

- **Instance Classes**
  - General Purpose (M4, M5)
  - Memory Optimized (R4, R5, X1e, X1)
  - Burstable Performance (T2, T3)

- **Storage Type**
  - General Purpose (SSD)
    - 3 IOPS per GiB, burst to 3,000 IOPS
  - Provisioned IOPS (SSD)
    - 1,000 to 80,000 IOPS (depending on the engine)

- **Monitoring**
  - CloudWatch metrics:
    - *Swap Usage*: Increase = low or no available RAM
    - *ReadIOPS/WriteIOPS*: Use this to determine storage type changes
    - *ReadLatency/WriteLatency*: Higher latency = more IOPS needed
    - *ReadThroughPut/WriteThroughput*: Average bytes per second
  - RDS events:
    - A record of instance, snapshot, security group, and parameter group events
  - Enhanced monitoring:
    - Real-time metrics for the OS of the DB instance
    - Gets metrics from an agent on the instance

Back to Main

- Managed in-memory data store and cache
- Memcached or Redis engines
- Typically used to offload database reads to improve web application performance

- **Node Type**
  - Families affect CPU
    - General Purpose (T2, M4, M5)
    - Memory Optimized (R4, R5)
  - Sizes affect memory and network

- **Monitoring**
  - CloudWatch metrics
    - *CPU Utilization*
      - When threshold is exceeded
        - Increase the node family
        - Add more read replicas (Redis) or nodes (Memcached)
    - *Evictions*
      - Older items are removed to free up memory for new items
      - Frequent evictions will decrease performance
        - Increase the node size (memory)
    - *CurrConnections*
      - The application may not be releasing connections
    - *Swap Usage*
      - Affects performance if increased
      - Should be close to 0
      - Increase node size

Core Components

- **High-Level Architecture**
  - Relational (SQL) vs. NoSQL
  - Partitions and data distribution
- **Tables, Items, Attributes, and Streams**
- **Data Types**
  - Scalar (number, string, binary, boolean, null)
  - Document (list, map)
  - Set
- **Read and Write Consistency**
  - Eventually consistent reads
  - Strongly consistent reads
- **Provisioned Throughput**
  - Read capacity units
  - Write capacity units
  - Reserved capacity
  - On-demand mode
- **Secondary Indexes**
  - Global
  - Local
- **On-Demand Backups and Point-in-Time Recovery**
- **VPC Endpoints**
- **When *Not* to Use DynamoDB**

Next

Back to Main

Relational (SQL) vs. NoSQL

In a relational database, a record is often normalized and stored in separate tables, and relationships are defined by primary and foreign key constraints.

ISBN	Title
0441172717	Dune
0553418025	The Martian

Books

ISBN	AuthorID
0441172717	1
0553418025	2

Author-ISBN

ID	Name
1	Frank Herbert
2	Andy Weir

Authors

In a NoSQL database like DynamoDB, a book record is usually stored as a single JSON document.

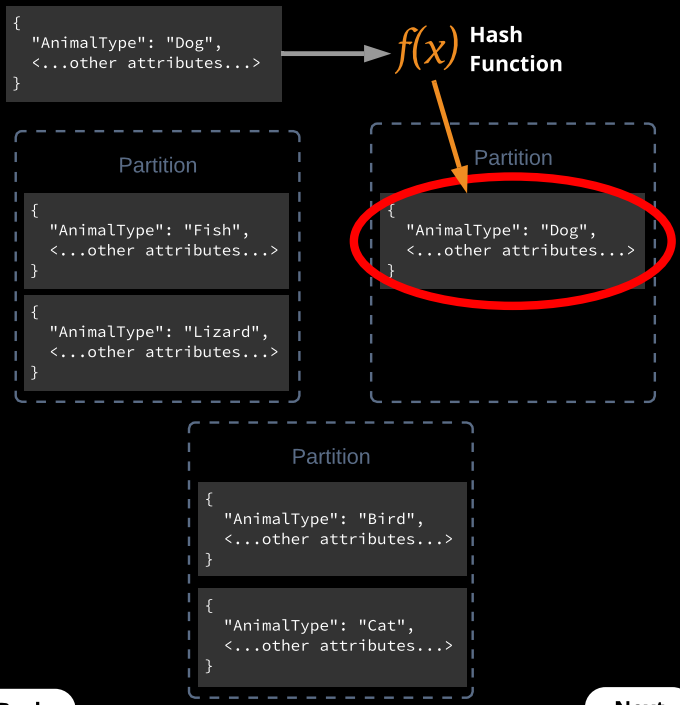
```
[
  {
    "ISBN": "0441172717",
    "Title": "Dune",
    "Author": "Frank Herbert"
  },
  {
    "ISBN": "0553418025",
    "Title": "The Martian",
    "Author": "Andy Weir"
  }
]
```

Key-value databases like DynamoDB are highly partitionable and allow horizontal scaling at scales that other types of databases cannot achieve. Use cases such as web, mobile, gaming, ad tech, and IoT lend themselves particularly well to the key-value data model. DynamoDB is designed to provide consistent single-digit millisecond latency for any scale of workloads.

Partitions and Data Distribution

DynamoDB stores data in partitions. A partition is an allocation of storage for a table, backed by solid-state drives (SSDs) and automatically replicated across multiple Availability Zones within an AWS Region.

To get the most out of DynamoDB throughput, create tables where the partition key has a large number of distinct values. Applications should request values fairly uniformly and as randomly as possible.

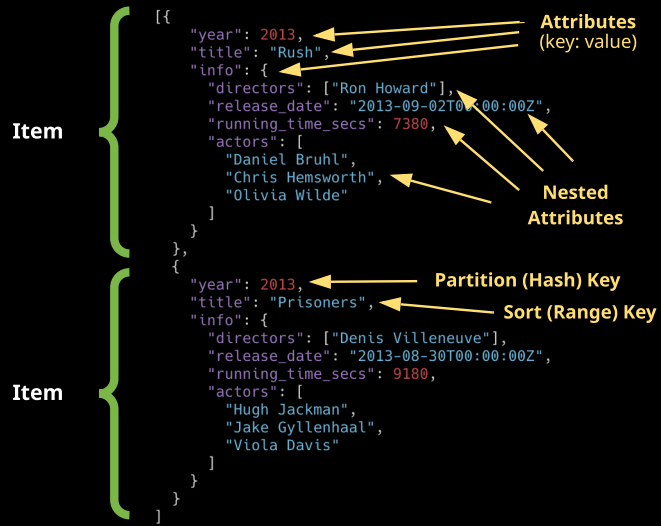


Tables, Items, Attributes, and Streams

Table: Collection of data. DynamoDB tables must contain a name, primary key, and the required read and write throughput values.

Item: A table may contain multiple items. An item is a unique group of attributes. Items are similar to rows or records in a traditional relational database. Items are limited to 400 KB.

Attribute: Fundamental data element. Similar to fields or columns in an RDBMS.



Stream: Ordered flow of information about table changes. For every create, update, or delete of an item, a **stream record** is created that contains those changes. Streams can be used to copy data from one table to another within a single region.

Data Types

Scalar: Exactly one value — number, string, binary, boolean, and null. Applications must encode binary values in base64-encoded format before sending them to DynamoDB.

Document: Complex structure with nested attributes (e.g.. JSON) — list and map.

Document Types

List: Ordered collection of values

```
FavoriteThings: ["Cookies", "Coffee", 3.14159]
```

Map: Unordered collection of name-value pairs (similar to JSON)

```
{
    Day: "Monday",
    UnreadEmails: 42,
    ItemsOnMyDesk: [
        "Coffee Cup",
        "Telephone",
        {
            Pens: { Quantity : 3},
            Pencils: { Quantity : 2},
            Erasers: { Quantity : 1}
        }
    ]
}
```

Set: Multiple scalar values of the same type — string set, number set, binary set.

```
["Black", "Green", "Red"]
[42.2, -19, 7.5, 3.14]
["U3Vubnk=", "UmFpbnk=", "U25vd3k="]
```

Read and Write Consistency

DynamoDB replicates data across multiple Availability Zones behind the scenes.

When an application writes data to a DynamoDB table and receives an HTTP 200 response (OK), all copies of the data are updated. The data will **eventually** be consistent across all storage locations. This is usually within one second or less.

DynamoDB supports eventually consistent and **strongly** consistent reads.

Eventually Consistent Reads

When you read data from a DynamoDB table, the response might not reflect the results of a recently completed write operation. The response might include some stale data. If you repeat your read request after a short time, the response should return the latest data. DynamoDB uses eventually consistent reads by default.

Strongly Consistent Reads

When you request a strongly consistent read, DynamoDB returns a response with the most up-to-date data, reflecting the updates from all prior write operations that were successful. A strongly consistent read might not be available if there is a network delay or outage.

Provisioned Throughput

Maximum amount of capacity that an application can consume from a table or index. Throttled requests: `ProvisionedThroughputExceededException`

Read Capacity Units

One read request unit represents one strongly consistent read request per second, or two eventually consistent read requests, for an item up to 4 KB in size. Transactional read requests require 2 read request units for items up to 4 KB.

For an 8 KB item size:

- 2 read request units for one strongly consistent read
- 1 read request unit for an eventually consistent read
- 4 read request units for a transactional read

Write Capacity Units

One write request unit represents one write per second for an item up to 1 KB in size. Transactional write requests require 2 write request units for items up to 1 KB.

For a 2 KB item size:

- 2 write request units
- 4 transactional write request units

Provisioned Throughput Scenario

On-Demand Mode

DynamoDB instantly accommodates your workloads as they ramp up or down to any previously reached traffic level. This is charged per request.

Reserved Capacity

Pay a one-time upfront fee and commit to a minimum usage level over a period of time, saving money compared to on-demand settings.

Back
Next

Scenario: You create a table with six RCUs and six WCUs. With these settings, your application could do the following:

- Perform strongly consistent reads of up to 24 KB per second (4 KB × 6 read capacity units).
- Perform eventually consistent reads of up to 48 KB per second (twice as much read throughput).
- Perform transactional read requests of up to 12 KB per second.
- Write up to 6 KB per second (1 KB × 6 write capacity units).
- Perform transactional write requests of up to 3 KB per second.

Secondary Indexes

Data structure that contains a subset of attributes from a table, along with an alternate key to support query operations. You can retrieve data from the index using a query, just like with a table. A table can have multiple secondary indexes.

Global Secondary Index: Index with a partition key and a sort key that can be different from those on the base table. The primary key of a GSI can be either simple (partition key) or composite (partition key and sort key).

Local Secondary Index: Index that has the same partition key as the base table, but a different sort key. The primary key of a LSI must be composite (partition and sort key).

Partition Key — Sort Key

username	email	order_id	total
mark	mark@linuxacademy.com	223	4096.64
terry	tcox@linuxacademy.com	224	1024.16

Order Table

Partition Key — Sort Key

username	email	order_id	total
mark	mark@linuxacademy.com	223	4096.64
terry	tcox@linuxacademy.com	224	1024.16

Global Secondary Index (GSI)

Partition Key — Sort Key

username	email	order_id	total
mark	mark@linuxacademy.com	223	4096.64
terry	tcox@linuxacademy.com	224	1024.16

Local Secondary Index (LSI)

On-Demand Backup and Restore

Create full backups of your tables at any time in either the AWS Management Console or with a single API call.

- Backup and restore actions execute with zero impact on table performance or availability.
- Backups are consistent within seconds and retained until deleted.
- Backup and restore operates within the same region as the source table.

Point-in-Time Recovery

Helps protect your DynamoDB tables from accidental writes or deletes. You can restore your data to any point in time in the last **35 days**.

- DynamoDB maintains **incremental** backups of your data.
- Point-in-time recovery is not enabled by default.
- The latest restorable timestamp is typically five minutes in the past.

After restoring a table, you must manually set up the following on the restored table:

- Auto scaling policies
- AWS Identity and Access Management (IAM) policies
- Amazon CloudWatch metrics and alarms
- Tags
- Stream settings
- Time to Live (TTL) settings
- Point-in-time recovery settings

VPC Endpoints

Use VPC endpoints to configure network traffic to be limited to the AWS cloud and avoid using the public internet.

- No need for an internet gateway or NAT gateway (keeps VPCs isolated from the public internet).
- No need to create and maintain firewalls to secure the VPC.
- Restrict access to DynamoDB through VPC endpoints using IAM policies.

VPC Endpoint Policy

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AccessFromSpecificEndpoint",
      "Action": "dynamodb:*",
      "Effect": "Deny",
      "Resource": "arn:aws:dynamodb:region:123456789012:mytable/*",
      "Condition": {
        "StringNotEquals": {
          "aws:sourceVpce": "vpce-11aa22bb33cc"
        }
      }
    }
  ]
}
```

When *Not* to Use DynamoDB

Before deciding to use DynamoDB, you should be able to answer "yes" to most of the following evaluation questions:

- Can you organize your data in hierarchies or a structure in one or two tables?
- Are data encryption and protection important?
- Are traditional backups impractical or cost-prohibitive because of table update rate or overall data size?
- Does your database workload vary significantly by time of day, or is it driven by a high growth rate or high-traffic events?
- Does your application or service consistently require response time in single milliseconds, regardless of loading and without tuning effort?
- Do you need to provide services in a scalable, replicated, or global configuration?
- Does your application need to store data in the high-terabyte size range?
- Are you willing to invest in a short but possibly steep NoSQL learning curve for your developers?

Some unsuitable workloads for DynamoDB include:

- Services that require ad hoc query access.
- Online analytical processing (OLAP), business intelligence (BI), or data warehouse implementations. Consider Amazon Redshift instead.
- Binary large object (BLOB) storage. DynamoDB can store binary items up to 400 KB, but DynamoDB is not generally suited to storing documents or images. The best architectural practice here is to store pointers to Amazon S3 objects in a DynamoDB table.

Redshift is a managed data warehousing solution, which can scale to petabytes or more.

Use Redshift to integrate with SQL; Business Intelligence (BI); and Extract, Transform, Load (ETL) tools to generate reports.

Redshift Spectrum allows you to perform SQL queries against exabytes of unstructured data in S3. It scales compute capacity based on the data being retrieved.

Redshift Architecture

Example: Amazon Payments

A Redshift cluster is a set of nodes that consists of a **leader** node and one or more **compute** nodes. The type and number of compute nodes needed depends on the size of the data, the number of queries executed, and the required query execution performance.

Leader node: Receives queries from client applications, parses the queries, and develops execution plans, which are an ordered set of steps to process these queries. The leader node then coordinates the parallel execution of these plans with the compute nodes, aggregates the intermediate results from these nodes, and finally returns the results back to the client applications. You can only have one leader node.

Compute nodes: Execute the steps specified in the execution plans and transmit data among themselves to serve these queries. The intermediate results are sent back to the leader node for aggregation before being sent back to the client applications.

Redshift vs. RDS

Both enable you to run traditional RDBMSs. RDS is typically used for OLTP and reporting. Redshift is appropriate for massively large data sets. Redshift provides excellent scale-out options and can be used to prevent interference with an OLTP workload.

Exam Tips

Course Navigation

Databases
Section 5

- **Amazon RDS: Understanding Multi-AZ Deployments**
- **RDS: Monitoring for Performance and Availability**
- **Amazon ElastiCache**
- **Amazon DynamoDB Concepts**
- **Amazon Redshift**

Amazon Aurora

Provisioning, Deployment, and Management
Section 6

Application Integration and Automation
Section 7

Monitoring and Metrics
Section 8

Redshift is a managed data warehousing solution, which can scale to petabytes or more.

Exam Tips

When to use which product?

RDS
- OLTP
- Read replicas across regions
- Snapshots in S3
- Lives inside VPC
- Security is DB user-based

DynamoDB
- Millisecond read latency
- Fully managed
- No backups required (PITR)
- Security is IAM-based

Redshift
- OLAP
- Accessed via SQL
- Massive amounts of data
- Complex queries across multiple data sources
- Lives inside VPC
- Security is DB user-based
- Best for structured data (e.g., CSV files)

Athena
- Apache Hive Query Language (HQL)
- Single data source
- Queries generally faster than Redshift
- Security is IAM-based
- Better for ad hoc querying

Elastic MapReduce (EMR)
- Based on Apache Hadoop
- Best for unstructured data

workload.

Exam Tips

Back to Main

Course Navigation

Databases
Section 5

- **Amazon RDS: Understanding Multi-AZ Deployments**
- **RDS: Monitoring for Performance and Availability**
- **Amazon ElastiCache**
- **Amazon DynamoDB Concepts**
- **Amazon Redshift**

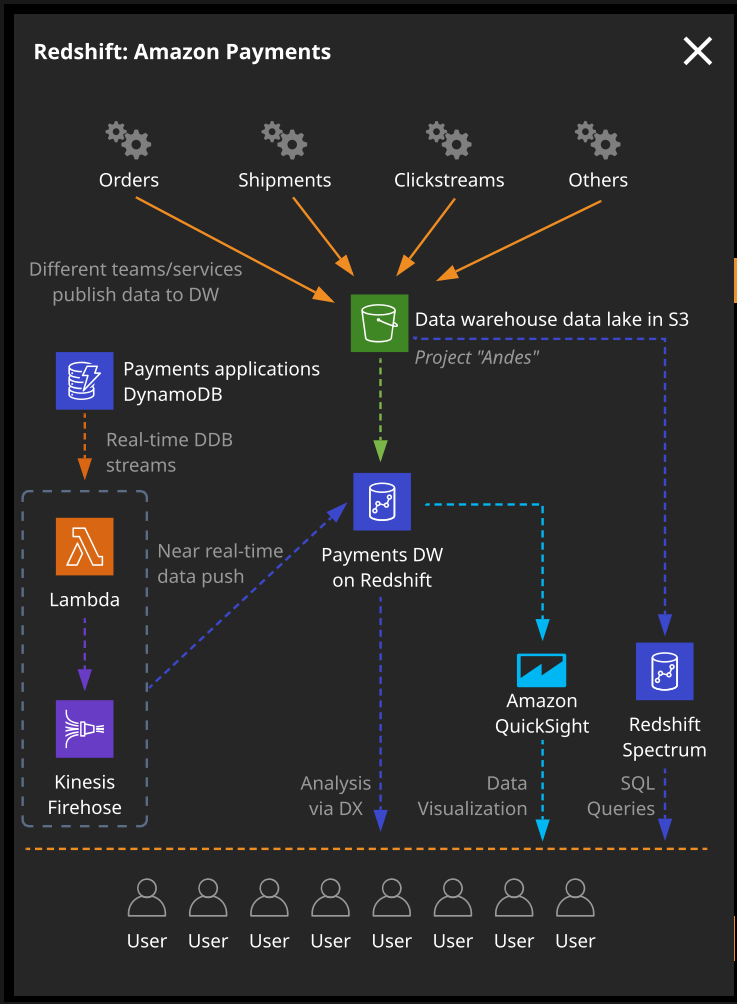
Amazon Aurora

Provisioning, Deployment, and Management
Section 6

Application Integration and Automation
Section 7

Monitoring and Metrics
Section 8

Back to Main



Databases

Amazon Aurora

Fully managed, highly available, relational database engine

Features:

- MySQL and PostgreSQL compatible
- Up to 5x faster than MySQL and 3x faster than PostgreSQL without changing your applications
- Fully managed, performing routine tasks:
  - Provisioning
  - Patching
  - Backup and recovery
  - Failure detection and repair
- Migration tools to convert RDS MySQL databases to Aurora
- Greater than 99.99% availability

Aurora Architecture

Distributed, fault-tolerant, self-healing storage system that scales up to 64 TB per DB instance.

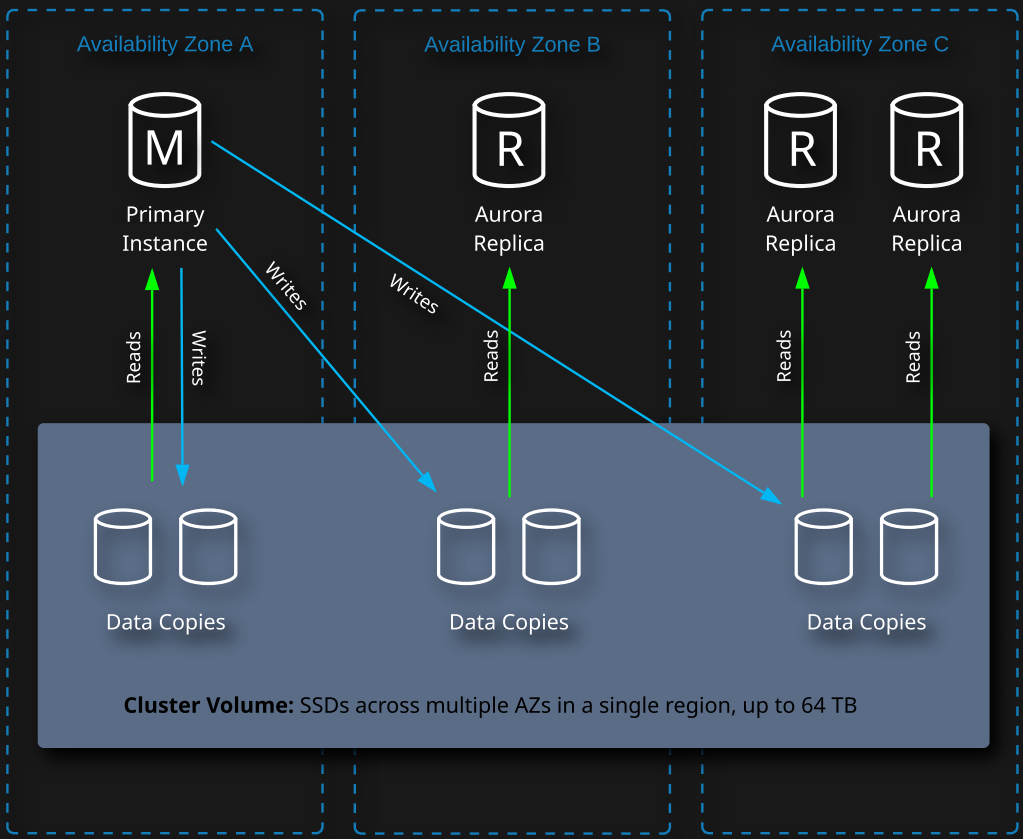
- Scales in 10 GB increments
- Replicates six copies of your data across three AZs
  - Single virtual volume using SSDs
  - Lose **two** copies of data without affecting **writes**
  - Lose **three** copies of data without affecting **reads**
- Continuously backs up to S3
- Restore from a DB snapshot or use point-in-time restore (PITR)
- **Global Database** can span multiple regions
- **Parallel Query** for MySQL distributes I/O across the storage layer for performance
- Security using IAM Database Authentication and VPC Security Groups
- Encrypt data at rest (on by default) with KMS or in transit with SSL
- Read replicas (encrypted if primary is)
  - Increase performance
  - Failover targets (can be promoted to primary)

Aurora Serverless

On-demand auto-scaling configuration for Aurora. No instances to manage. Charged on a per-second basis.

Back to Main

Amazon Aurora DB Cluster



Scenario: Aurora is near 100% CPU utilization. If writes, then scale up (increase instance size). If reads, then scale out (increase number of replicas).

Provisioning, Deployment, and Management

Course Navigation

AWS Elastic Beanstalk

Deploy and manage applications without worrying about the infrastructure.

Elastic Beanstalk handles capacity provisioning, health monitoring, Auto Scaling, load balancing, and updates for you automatically.

Highly available by default using two AZs and a load balancer.

Language and Platform Support

- Java
- .NET
- Node.js
- PHP
- Ruby
- Python
- Go
- Docker
- Apache
- IIS
- Nginx
- Tomcat

Example Web Server Environment

When to Use Elastic Beanstalk

- To quickly provision an AWS environment that requires little to no management
- The application fits within the parameters of the Beanstalk service
- Can deploy from repositories or from uploaded code files
- Easily update applications by uploading new code files or requesting a pull from a repository

Deployment Options

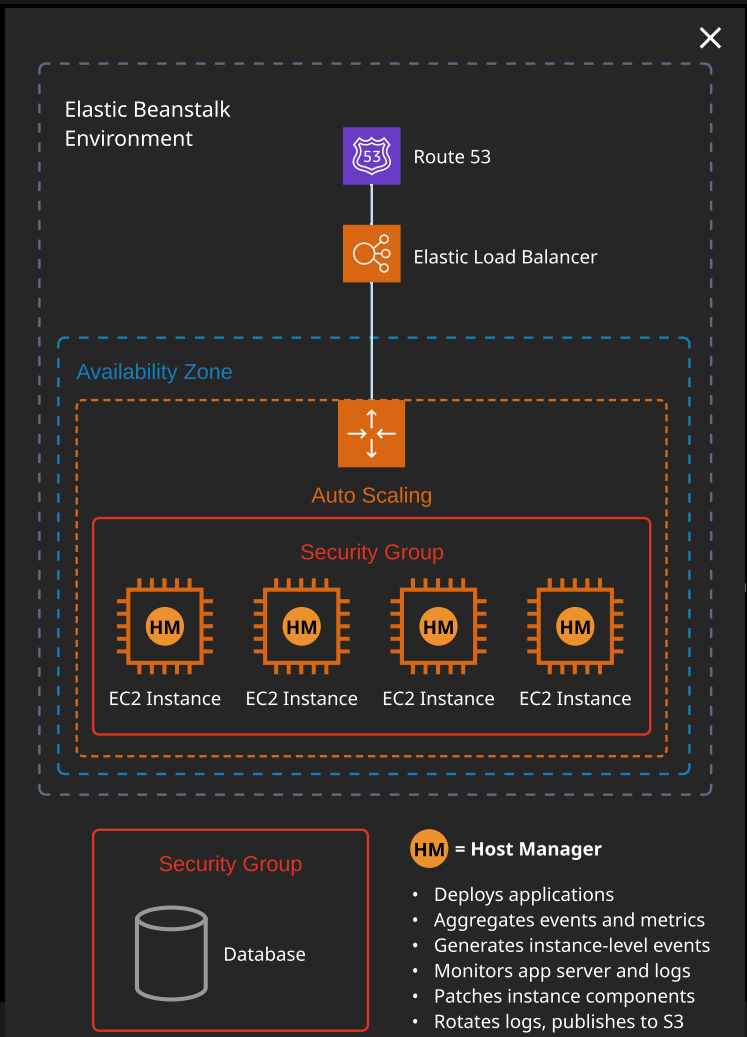
All at Once: Deploy the new version to all instances simultaneously. All instances in your environment are out of service for a short time while the deployment occurs.

Blue/Green: Deploy the new version to a separate environment, and then swap CNAMEs of the two environments to redirect traffic to the new version instantly.

Rolling: Beanstalk splits the environment's EC2 instances into batches and deploys the new version of the application to one batch at a time.

Back to Main

Provisioning, Deployment, and Management



Provisioning, Deployment, and Management

Amazon Elastic Container Service (ECS)

Amazon ECS is a highly scalable container management service. ECS makes it easy to run, stop, and manage Docker containers on EC2 instances, as well as deploy "serverless" containers with AWS Fargate.

Schedule the placement of containers on your clusters, based on resource needs.

Deploy microservices, as well as batch and extract, transform, load (ETL) workloads.

Components

- Container:
  - Virtualization method allowing you to run applications in isolated processes
  - Contains all the downloaded software, code, runtime, system tools, and libraries
  - Packaged as read-only templates called Docker images
- Dockerfile:
  - Text file that specifies all the components needed in the container:
    - The instructions for what will be placed inside a container
- Container Registry:
  - A repository where container/Docker images are stored and accessed
  - A container registry can be:
    - Amazon Elastic Container Registry (ECR)
    - A third-party repository like Docker Hub
    - Self-hosted registry
- Task Definition:
  - JSON-formatted text file that contains the "blueprint" for your application:
    - Container image
    - Container registry
    - Ports that should be open on the instance
    - Data volumes
- Service:
  - Defines how to run and maintain a specified number of instances together
  - Optional load balancing
- Cluster:
  - Group of tasks or services on multiple EC2 or Fargate instances
- Fargate:
  - A "serverless" launch type that eliminates the need for explicit infrastructure. Think AWS Lambda for containers.

Back to Main

Provisioning, Deployment, and Management

AWS Systems Manager

Systems Manager helps you automatically collect software inventory, apply OS patches, create system images, and configure Windows and Linux operating systems.

Systems Manager can be used for both EC2 instances and for compute instances located in your own data centers. These capabilities help you define and track system configurations, prevent drift, and maintain software compliance of your EC2 and on-premises configurations.

Systems Manager Automation simplifies common instance and system maintenance and deployment tasks. For example, you can use Automation as part of your change management process to keep your AMIs up to date with the latest application build. You can also create a backup of a database and upload it nightly to S3.

With Automation, you can avoid deploying scripts and scheduling logic directly to the instance. Instead, you can run maintenance activities through Systems Manager Run Command and AWS Lambda steps orchestrated by the Automation service.

Systems Manager Inventory provides visibility into your EC2 and on-premises environments. You can use Inventory to collect metadata from your managed instances. You can store this metadata in an S3 bucket, and then use built-in tools to query the data and quickly determine which instances are running the software and configurations required by your software policy, as well as which instances need to be updated.

You can configure Inventory on all your managed instances via a one-click procedure. You can also configure and view inventory data from multiple AWS regions and accounts.

Patch Manager automates the process of patching managed instances with both security-related and other types of updates. You can use Patch Manager to apply patches for both operating systems and applications.

Run Command automates tasks across resources (e.g., software package installs).

Parameter Store provides storage and management of your secrets and configuration data such as passwords, database strings

Back to Main

Provisioning, Deployment, and Management

AWS OpsWorks

- Flexible way to create and manage resources for our applications.
- Create a stack of resources and manage resources in layers.
- Use OpsWorks to automate, monitor, and maintain deployments.
- OpsWorks provides abstraction from the underlying infrastructure, while still giving plenty of control, with more customization than Elastic Beanstalk.
- It uses Chef, an open-source tool that automates infrastructure by turning it into code.

Anatomy

- Stacks:
  - Represent a set of resources we want to manage as a group
  - Can build a stack for a development, staging, or production environment
- Layers:
  - Used to represent and configure components of a stack
  - We can use built-in layers and customize those or create completely custom layers
  - Recipes are added to layers
- Instances:
  - Must be associated with at least one layer
  - Can run as: 24/7, load-based, time-based
- Apps:
  - Apps are deployed to the application layer through a source code repository like Git, Subversion, or S3
  - We can deploy an app against a layer and have OpsWorks execute recipes to prepare instances for the application

Recipes

- Created using the Ruby language and based on the Chef deployment software
- Custom recipes can customize different layers in an application
- Recipes are run at certain predefined events within a stack
  - Setup: Occurs on a new instance after first boot
  - Configure: Occurs on all stack instances when they enter or leave the inline state
  - Deploy: Occurs when we deploy an app
  - Undeploy: Happens when we delete an app from a set of application instances
  - Shutdown: Happens when we shut down an instance (but before it is actually stopped)

Provisioning, Deployment, and Management

Any event that has a negative impact on your company's business continuity or finances could be termed a disaster.

Recovery time objective (RTO): This represents the time it takes after a disruption to restore a business process to its service level, as defined by the operational level agreement (OLA). For example, if a disaster occurs at 12:00 PM (noon) and the RTO is eight hours, the DR process should restore the business process to the acceptable service level by 8:00 PM.

Recovery point objective (RPO): This is the acceptable amount of data loss measured in time. For example, if a disaster occurs at 12:00 PM (noon) and the RPO is one hour, the system should recover all data that was in the system before 11:00 AM. Data loss will span only one hour, between 11:00 AM and 12:00 PM (noon).

Backup and Restore Method

Pilot Light Method

Warm Standby Method

Multi-Site Solution Method

Failing Back

Once you have restored your primary site to a working state, you will need to restore your normal service, which is often referred to as a "failback."

Backup and restore failback is a reverse process: freeze data changes, take a backup, restore to primary site, redirect traffic, and unfreeze changes.

All other methods require reversing the data replication back to the primary site, freezing changes to the DR site, redirecting traffic, and unfreezing changes.

Provisioning, Deployment, and Management

Any event that has a negative impact on your company's business continuity or finances could be termed a disaster.

Recovery time objective (RTO): This represents the time it takes after a disruption to restore a business process to its service level, as defined by the operational level agreement (OLA). For example, if a disaster occurs at 12:00 PM (noon) and the RTO is eight hours, the DR process should restore the business process to the acceptable service level by 8:00 PM.

Recovery point objective (RPO): This is the acceptable amount of data loss measured in time. For example, if a disaster occurs at 12:00 PM (noon) and the RPO is one hour, the system should recover all data that was in the system before 11:0 PM (noo

Multi-Site Solution Method

- Fastest possible system restore
- One-to-one copy of all infrastructure in another AZ or region
- **Active-active** configuration
- Most expensive DR plan
- Best RTO and RPO, as no downtime and no data loss are expected
- Uses data replication (either synchronous or asynchronous)
- Can perform weighted DNS routing with Route 53
- Uses Auto Scaling and instance resizing to increase capacity in a disaster scenario

Failing Back

Once you have restored your primary site to a working state, you will need to restore your normal service, which is often referred to as a "failback."

Backup and restore failback is a reverse process: freeze data changes, take a backup, restore to primary site, redirect traffic, and unfreeze changes.

All other methods require reversing the data replication back to the primary site, freezing changes to the DR site, redirecting traffic, and unfreezing changes.

Any event that has a negative impact on your company's business continuity or finances could be termed a disaster.

Recovery time objective (RTO): This represents the time it takes after a disruption to restore a business process to its service level, as defined by the operational level agreement (OLA). For example, if a disaster occurs at 12:00 PM (noon) and the RTO is eight hours, the DR process should restore the business process to the acceptable service level by 8:00 PM.

Recovery point objective (RPO): This is the acceptable amount of data loss measured in time. For example, if a disaster occurs at 12:00 PM (noon) and the RPO is one hour, the system should recover all data that was in the system before 11:0 PM (noo

Pilot Light Method

- Quicker than backup and restore method
- Slower than warm standby method
- **Most critical core components** of your system are always running and kept up to date
- More costly than backup and restore
- Rapidly provision a full production environment around the critical core
- Typically includes your database servers, configured for replication
- Restoring the other components includes using EBS snapshots and EC2 AMIs that you should be generating regularly

Fail

Once you have restored your primary site to a working state, you will need to restore your normal service, which is often referred to as a "failback."

Backup and restore failback is a reverse process: freeze data changes, take a backup, restore to primary site, redirect traffic, and unfreeze changes.

All other methods require reversing the data replication back to the primary site, freezing changes to the DR site, redirecting traffic, and unfreezing changes.

Provisioning, Deployment, and Management

Any event that has a negative impact on your company's business continuity or finances could be termed a disaster.

Recovery time objective (RTO): This represents the time it takes after a disruption to restore a business process to its service level, as defined by the operational level agreement (OLA). For example, if a disaster occurs at 12:00 PM (noon) and the RTO is eight hours, the DR process should restore the business process to the acceptable service level by 8:00 PM.

Recovery point objective (RPO): This is the acceptable amount of data loss measured in time. For example, if a disaster occurs at 12:00 PM (noon) and the RPO is one hour, the system should recover all data that was in the system before 11:0 (noo

Warm Standby Method

- Faster than pilot light method, but slower than having a multi-site solution
- Scaled-down version of a fully functional environment is always running
- More costly than pilot light
- Resize instances after failover
- Like pilot light, uses database replication

Method Solution Method

Failing Back

Once you have restored your primary site to a working state, you will need to restore your normal service, which is often referred to as a "failback."

Backup and restore failback is a reverse process: freeze data changes, take a backup, restore to primary site, redirect traffic, and unfreeze changes.

All other methods require reversing the data replication back to the primary site, freezing changes to the DR site, redirecting traffic, and unfreezing changes.

Provisioning, Deployment, and Management

Any event that has a negative impact on your company's business continuity or finances could be termed a disaster.

Recovery time objective (RTO): This represents the time it takes after a disruption to restore a business process to its service level, as defined by the operational level agreement (OLA). For example, if a disaster occurs at 12:00 PM (noon) and the RTO is eight hours, the DR process should restore the business process to the acceptable service level by 8:00 PM.

Recovery point objective (RPO): This is the acceptable amount of data loss measured in time. For example, if a disaster occurs at 12:00 PM (noon) and the RPO is one hour, the system should recover all data that was in the system before 11:00 AM. Data loss will exist between 11:00 AM and 12:00 PM (noon).

Backup and Restore Method

- Slowest restoration time after an event
- Requires frequent snapshots of data (e.g., EBS volumes, RDS databases) and storing them in a secure location (e.g., S3)
- Storage Gateway enables snapshots of on-premises data to be copied to S3
- Gateway VTL (virtual tape library) can replace magnetic tape backup
- Used in conjunction with other DR methods since it is critical to always have a working backup of your system
- Ensure proper encryption and data access policies

Failing Back

Once you have restored your primary site to a working state, you will need to restore your normal service, which is often referred to as a "failback."

Backup and restore failback is a reverse process: freeze data changes, take a backup, restore to primary site, redirect traffic, and unfreeze changes.

All other methods require reversing the data replication back to the primary site, freezing changes to the DR site, redirecting traffic, and unfreezing changes.

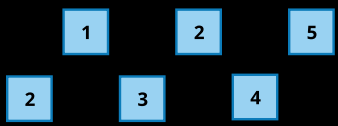
Application Integration and Automation

Simple Queue Service (SQS)

- Fully managed message queuing service
  - Highly scalable to billions of messages per day
  - Message durability is high
- Uses a polling model for message exchange
  - Applications must poll the queue
  - Separates the sending and receiving
  - Application components do not have to be available together

Queue Types

- **Standard**
  - Unlimited throughput
  - Messages are delivered at least once
    - Can be more than once
  - Best effort ordering
    - Nearly sequential



- **FIFO**
  - High throughput
  - Messages are sent exactly once
  - First-In, First-Out delivery
    - Order is strictly preserved

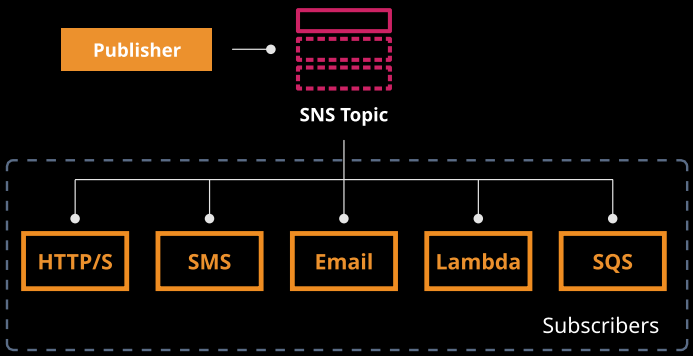
Next

Simple Notification Service (SNS)

A fully managed service that sends messages to subscription endpoints.

Components

- **Publisher:** The client user or service that sends the message to SNS
- **Topic:** The channel for "routing" the message/communication
- **Subscription:** The client user or service who receives the message



Important Features

- Uses a "push" model for message delivery.
- Applications can send one message instead of many.
- Subscriptions can change without changing the application.
- SNS can also send mobile push notifications.

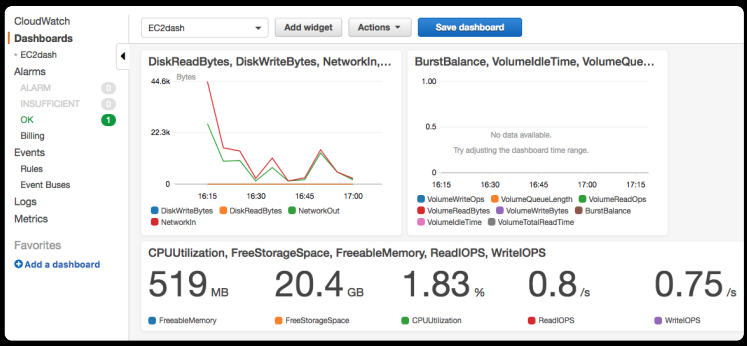
Back

Application Integration and Automation

- Lambda is a "serverless" computing platform.
- Serverless means you can run code without provisioning or managing servers:
  - If you want to run code, you don't have to spin up an EC2 instance and install software — you can just create a Lambda function, drop your code in it, and execute it.
- Lambda scales the required compute power automatically with your code.
- You pay only for the compute time you consume (to the 100 ms).
- By default, it is highly available, fault tolerant, scalable, elastic, and cost-efficient.
  - Regional service running across multiple AZs
- Lambda integrates with many other AWS services.
- Current supported languages include:
  - .NET Core
  - Go
  - Java
  - Node.js
  - Python
  - Ruby
  - Custom
- Security is integrated with IAM using **execution roles**.
  - Govern which AWS resources the function has access to
- When should you use Lambda over EC2? Generally, you want to use Lambda when you want to run code that is in response to events, such as:
  - Changes to S3 buckets
  - Messages in SQS queues
  - Updates to a DynamoDB table
  - CloudWatch alarms
  - Custom events generated by your applications or devices

CloudWatch is:

- Real-time monitoring of your applications and resources on AWS
- Repository of metrics (AWS provided and custom)
  - **Metrics** are data points related to resources and applications
  - Examples:
    - CPU utilization on EC2
    - Number of connections to a database in RDS
  - AWS services that provide free metrics:
    - EC2, EBS, RDS
  - Retention periods:
    - 1-minute metrics are available for 15 days (detailed monitoring)
    - 5-minute metrics are available for 63 days
    - 1-hour metrics are available for 455 days
- Metrics can be graphed and compared
  - For example: The CPU utilization for many instances can be layered on the graph together
- These graphs can then be configured as widgets and added to **dashboards**
  - A user is fully able to customize a dashboard to show the metrics they want to monitor
  - A user can also configure multiple dashboards



Monitoring and Metrics

CloudWatch Alarms

What Else Can We Do with Metric Data?

- CloudWatch **Alarms**
  - Watches a single metric over a specified time period and, based on the value of that metric relative to a threshold over time, performs one or more specified actions
  - Three states of alarm:
    - **OK:** The threshold is in the normal range
    - **ALARM:** The threshold has been exceeded
    - **INSUFFICIENT:** There is not enough data to evaluate the state
-
  - The main components of an alarm configuration:
    - **Metric:** The data we are measuring
    - **Thresholds:** The point at which we want some type of notification
    - **Period:** The defined amount of time before notification
    - **Action:** Has two parts:
      - Change the state
      - Send a notification

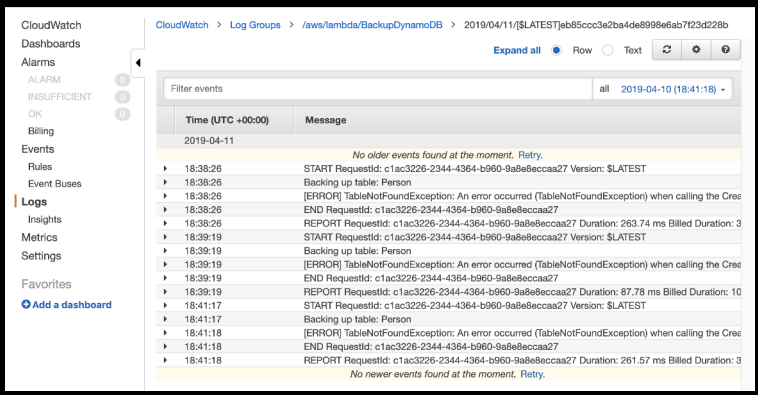
Where Do These Notifications Go?

- Services that can receive CloudWatch alarm notifications
  - **Simple Notification Service (SNS)**
    - A topic gets triggered by CloudWatch
      - Subscribers to that topic are notified
        - HTML, email, SQS, application, Lambda, SMS
  - **Auto Scaling**
    - CloudWatch alarms trigger the scale up/down scenarios
  - **EC2**
    - CloudWatch alarms trigger EC2 actions like:
      - Recover, stop, terminate, or reboot
    - "Per-instance" metric is required

Monitoring and Metrics

CloudWatch Logs

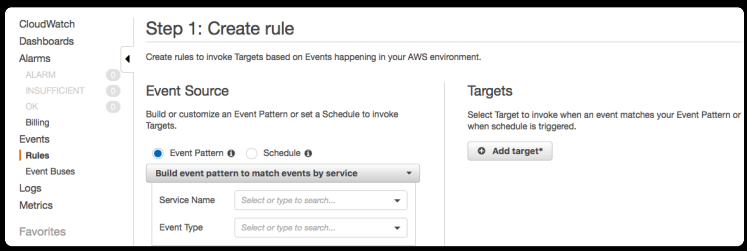
- Use **CloudWatch Logs** to monitor, store, and access your log files from:
  - **EC2:** Applications can be configured to send logs
    - Exceptions, rate of errors, etc.
    - Requires the install of CloudWatch Logs agent
  - **On-Premises Servers:** With agent installed
  - **CloudTrail:** Get logs from API activity in your account
  - **Route 53:** Log information from DNS queries
- **Components**
  - **Log Events:** Record of activity recorded by the monitored resource
  - **Log Streams:** Sequence of log events from the same source/application
  - **Log Groups:** A collection of log streams with same access control, monitoring, and retention settings
  - **Metric Filters:** Assigned to log groups, it extracts data from the groups' log streams and converts that data into a metric data point
  - **Retention Settings:** Period of time logs are kept. Assigned to log groups, but applies to all the streams in a group.
- **CloudWatch Logs Insights** lets you interactively query and analyze your CloudWatch Logs data.



Monitoring and Metrics

CloudWatch Events

- CloudWatch Events are similar to alarms. Instead of configuring thresholds and alarming on metrics, CloudWatch Events are matching event patterns and use targets to react.
- Near real-time
- Consists of three parts:
  - **Event Source:** An operational **change** in a service or can be **scheduled**
  - **Rules:** Route matching events to targets
  - **Targets:** The services that will react to the event
    - There can be more than one
    - Some of the services that can be targets:
      - EC2, Lambda functions, ECS tasks
      - Kinesis Data Streams and Firehose
      - Systems Manager Run Command and Automation
      - CodeBuild project, CodePipeline
      - SNS and SQS
- Examples:
  - Sharing an EBS snapshot triggers an SNS topic and a Lambda function
  - Scheduling an EBS snapshot

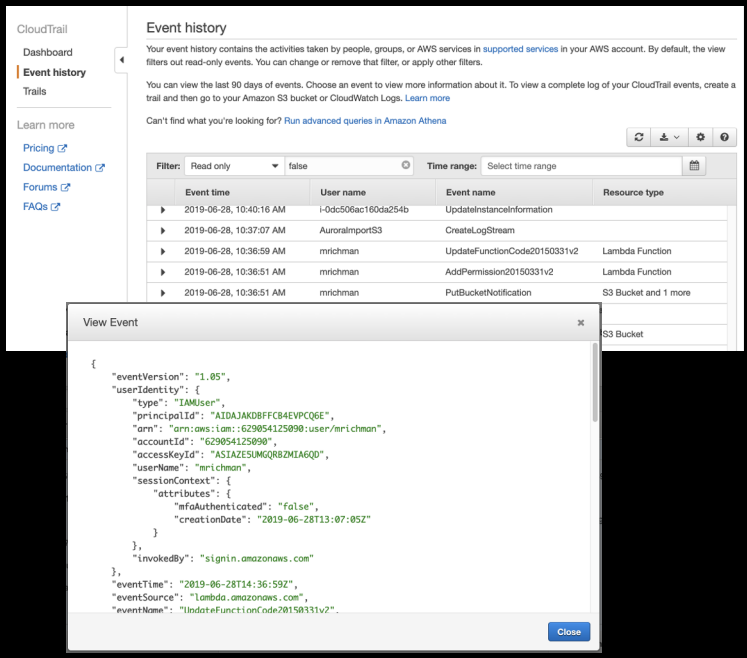


Back to Main

Monitoring and Metrics

AWS CloudTrail

- Allows for auditing of your AWS environment
- Logs every API call to your resources (Console, AWS CLI, SDK)
- Every interaction with AWS is an API call
- Can provide information for many types of activities:
  - Governance
  - Compliance
  - Operational auditing
  - Risk auditing
- **Trail:** Configuration allowing for event logging from all regions to a single S3 bucket:
  - Can be analyzed by your preferred tool from there
- Can also report to CloudWatch Logs for near real-time monitoring and alerting



- A detailed view of the configuration of AWS resources (EC2, EBS, security group, VPC, etc.)
- A complete list of supported services is available in AWS documentation

With **AWS Config**, you can:

- Evaluate resource configurations for desired settings
- Get a snapshot of the current configurations associated with your account
- Retrieve configurations of resources in your account
- Retrieve past configurations
- Receive notifications for creations, deletions, and modifications
- View relationships between resources (e.g., members of a security group)

Uses of AWS Config

- Administering resources
  - Receive a notification when a resource violates configuration rules
- Auditing and compliance
  - Records of configurations are sometimes needed for audits
- Configuration management and troubleshooting
  - Configuration changes on one resource might affect others
  - Can help find these issues quickly and restore last known good configuration
- Security analysis
  - Allows for records of IAM policies
  - For example, what permissions a user had at the time of an issue
    - Allows for records of security group configurations

Service Health Dashboard

Provides access to current status and historical data about all AWS services. If there is a problem with a service, you can expand the appropriate line in the details section to get more information.

https://status.aws.amazon.com/

You can subscribe to the RSS feed for any service.

There is a **Contact Us** link if you experience any real-time operational issues.

Status history shows outage issue details on a daily basis.

Personal Health Dashboard

Provides alerts and remediation guidance when AWS is experiencing issues that may impact customers.

Shows a personalized view of the performance and availability of the AWS services underlying your provisioned AWS resources.

https://phd.aws.amazon.com

Management, Governance, and Cost Controls

AWS Billing and Organizations

- Pay your AWS bill, monitor usage, and set budgets

- **Organizations**
  - Centrally manage billing; control access, compliance, and security; and share resources across multiple AWS accounts
  - Group accounts together under a master or *payer* account
    - Initially low limit on number of linked accounts
    - Contact AWS Support to increase the limit
  - Volume discounts (data tiers) and shared resources (Reserved Instances)

- **Cost and Usage Report**
  - Creates detailed reports and stores them in S3
    - Takes up to 24 hours on first load
    - Updates happen at least once a day

- **Budgets**
  - Uses Cost Explorer data to show budget status
  - Sort of a prediction of what the costs will be
  - Billing alarms (CloudWatch)
    - Set a billing alarm so a notification goes out when $X is spent

- **CloudWatch Metrics**
  - Two dimensions:
    - By service
    - Total estimated charge
  - The following metrics are available in CloudWatch for billing:
    - *EstimatedCharges*
    - *ServiceName*
    - *LinkedAccount*
      - Consolidated billing only
    - *Currency*
      - Change this in the My Account settings

Management, Governance, and Cost Controls

AWS Cost Explorer

- A graphing tool for costs
  - Displays data from the last 13 months
  - Displays projected data for the next three months
- Helps reveal patterns and identify areas that can help control costs
- Uses different resource attributes for filtering the graphs
  - Filters include:
    - *Service*
    - *Linked Account*
    - *Region*
    - *Availability Zone*
    - *Instance Type*
    - *Usage type*
    - *Usage type group*
    - *Tag*

Management, Governance, and Cost Controls

- **EC2 Reserved Instances**
  - Save costs by purchasing Reserved Instances (1- to 3-year reservation)
  - Pay all, in part, or nothing up front (more savings the more you pay up front)
  - Some instances can be sold for a fee on the Marketplace
- **Spot Instances**
  - "Name your price" purchasing option (save up to 90%)
  - Can be interrupted, so plan accordingly (new pause feature for C5 and M5)
- **Low Utilization**
  - Set CloudWatch alarms to terminate underutilized instances
    - Example: 5% CPU utilization for 50 minutes
  - Find the right balance between availability and cost
    - Example: How much does 1 minute of downtime cost vs. the cost of eliminating that downtime?
- **Unused Load Balancers**
  - Remove them
- **EBS Volumes**
  - EBS volumes cost money, even when not in use
  - Delete unused volumes; take a snapshot if you want to keep the data
    - Snapshot storage is cheaper
  - Provisioned IOPS cost more
    - Make sure you're not provisioning more than necessary
  - Downsize volumes that aren't anywhere near full capacity
- **Elastic IP Addresses**
  - EIPs cost money when not in use — associate them
  - Having more than one EIP associated to an instance costs money
  - EIPs on stopped instances cost an hourly fee
- **Idle Amazon RDS DB Instances**
  - Snapshot unused DB instances, and delete them (0 connections over time)