Implementing Monitoring, Elasticity, and High Availability

AWS Academy Cloud Architecting

Introduction

Implementing Monitoring, Elasticity, and High Availability

Module objectives

This module prepares you to do the following:

- Examine how reactive architectures use Amazon CloudWatch and Amazon EventBridge to monitor metrics and facilitate notification events.
- Use Amazon EC2 Auto Scaling within an architecture to promote elasticity and create a highly available environment.
- Determine how to scale your database resources.
- Identify a load balancing strategy to create a highly available environment.
- Use Amazon Route 53 for DNS failover.
- Use the AWS Well-Architected Framework principles when designing highly available systems.

Module overview

Presentation sections

- Monitoring your resources
- Scaling your compute resources
- Scaling your databases
- Using load balancers to create highly available environments
- Using Route 53 to create highly available environments
- Applying AWS Well-Architected Framework principles to highly available systems

Demos

- Creating Scaling Policies for Amazon EC2 Auto Scaling
- Creating a Highly Available Application
- Amazon Route 53: Simple Routing
- Amazon Route 53: Failover Routing
- Amazon Route 53: Geolocation Routing

Knowledge checks

- 10-question knowledge check
- Sample exam question

Hands-on labs in this module

Guided lab

- Creating a Highly Available Environment

Challenge (Café) lab

- Creating a Scalable and Highly Available Environment for the Café

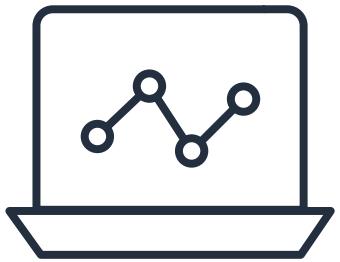
As a cloud architect building architectures that perform as needed:

- I need to design workload monitoring across distributed components to provide a complete picture of workload performance.
- I need to understand how to automate the scaling of cloud compute and database resources to ensure high availability for my workloads.
- I need to design workload failover mechanisms by using load balancers and network routing to ensure high availability in case of failures.

Monitoring your resources

Implementing Monitoring, Elasticity, and High Availability

Monitoring distributed application components



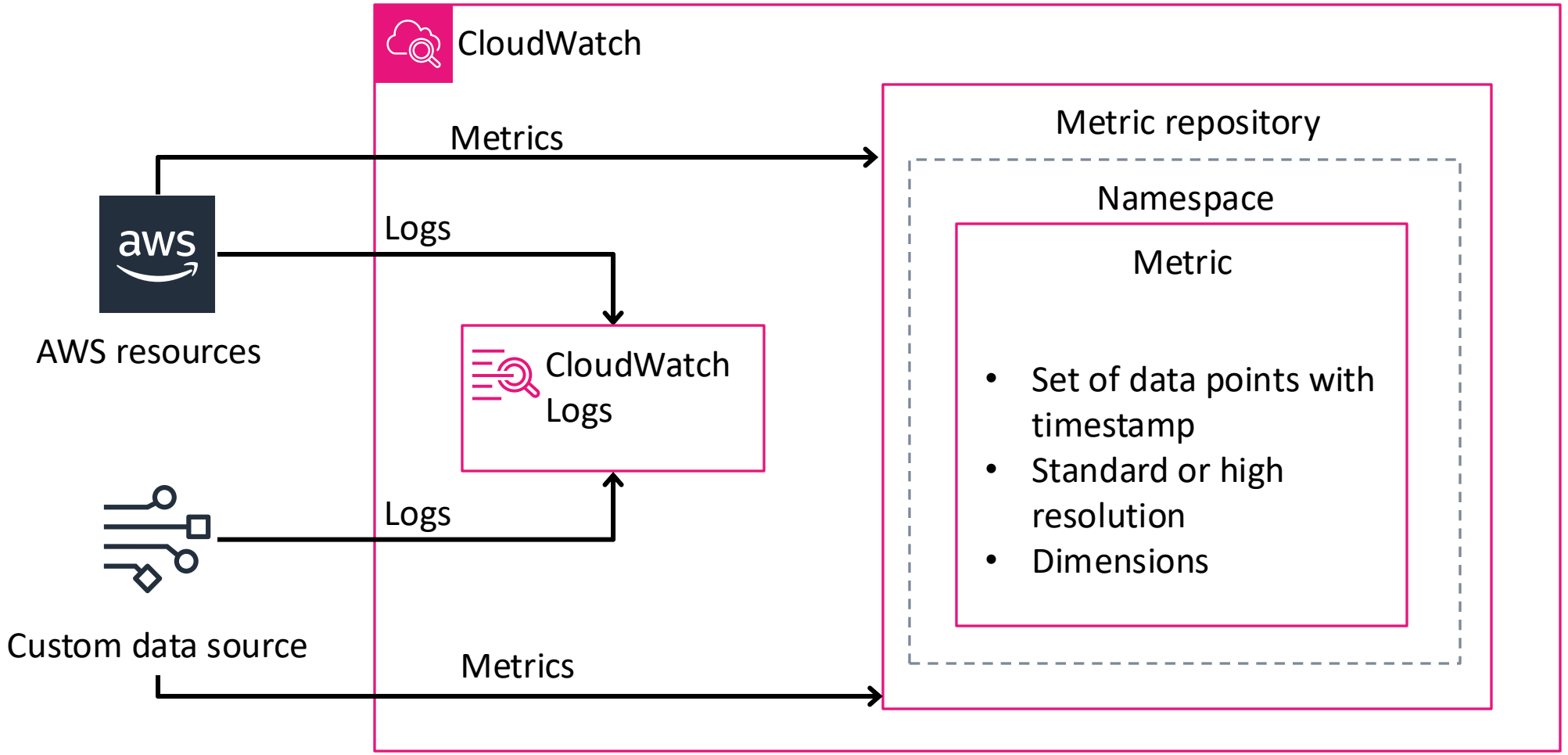
Operational health	Resource utilization	Application performance

CloudWatch

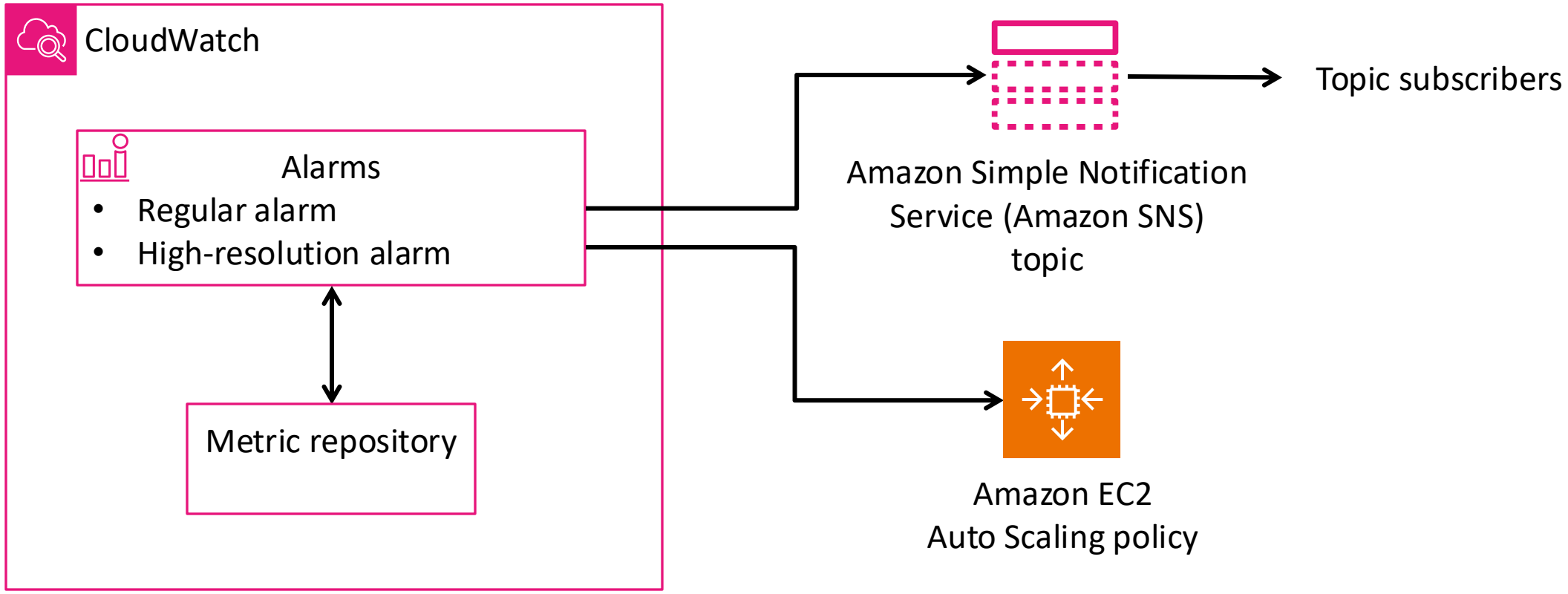
CloudWatch

- Collects and tracks metrics for AWS services across Regions in a metric repository
- Collects logs by using Amazon CloudWatch Logs
- Supports built-in AWS service metrics or custom metrics
- Calculates statistics from metrics and and displays graphs of the metric statistics
- Provides alarms for responsive event-driven architectures
- Provides notifications to make changes to monitored resources

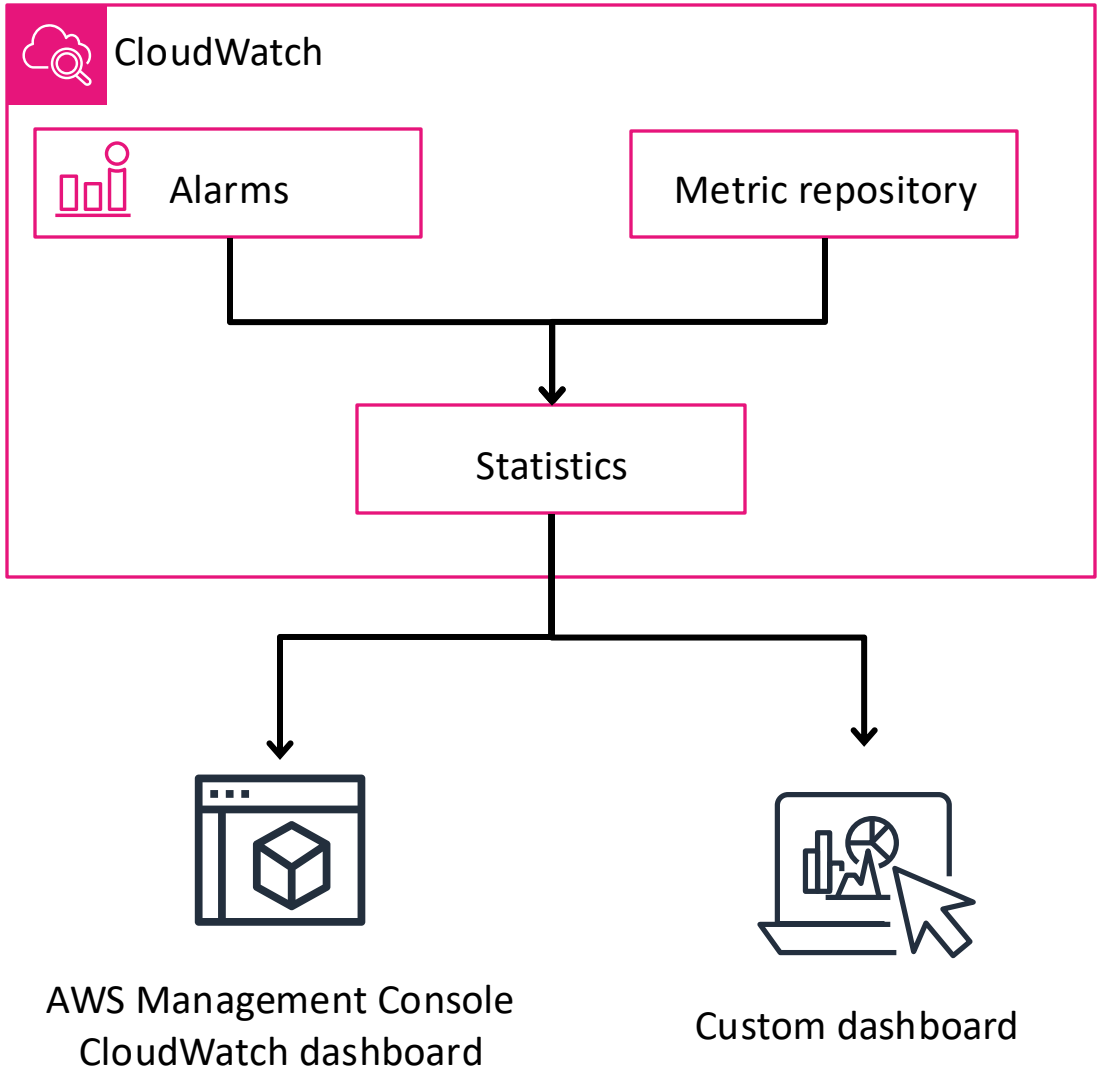
CloudWatch metric and log collection



CloudWatch alarms



CloudWatch dashboards

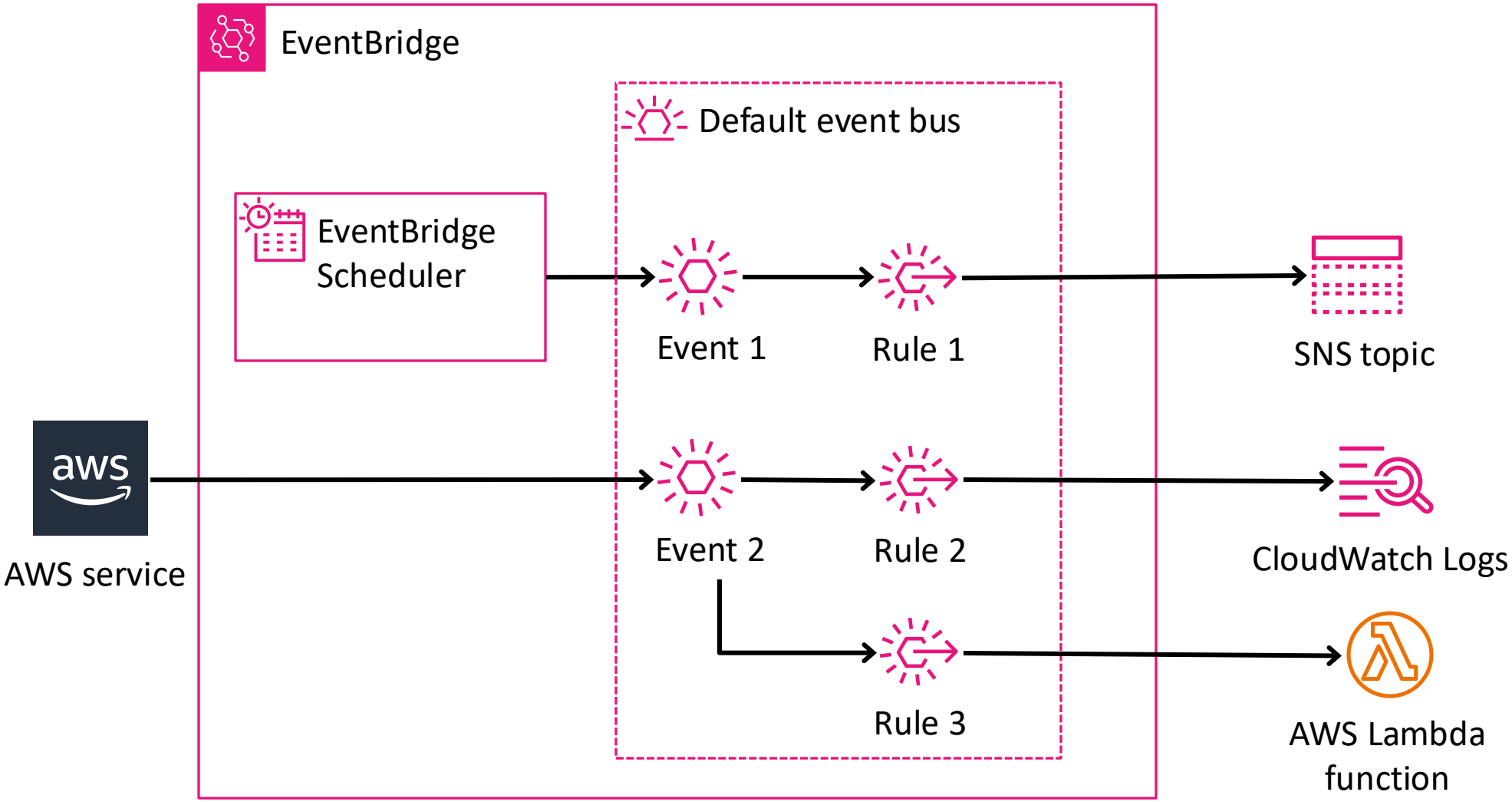


EventBridge

EventBridge

- Processes events with an event bus or a pipe
  - Pipes are point-to-point integrations between one source and one target.
  - Event buses are routers that receive events and optionally delivers them to one or multiple targets.
- Makes routing decisions with configurable rules
  - Rules run based on matching an event pattern or on a schedule with Amazon EventBridge Scheduler.
  - Targets can be in another AWS account, in another Region, or in both.

EventBridge event bus



Monitoring your resource costs

AWS Cost Explorer

AWS Budgets

AWS Cost and Usage Report

Key takeaways: Monitoring your resources

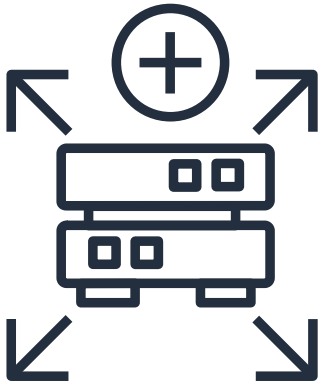
- CloudWatch alarms can send notifications to Amazon EC2 Auto Scaling and SNS topics.
- CloudWatch collects logs and metrics from AWS services across Regions.
- You can use CloudWatch dashboards to visualize metrics and alarms.
- EventBridge processes and routes event with an event bus or a pipe.
- AWS Cost Explorer, AWS Budgets, and AWS Cost and Usage Report can help you understand and manage the cost of your AWS infrastructure.

Scaling your compute resources

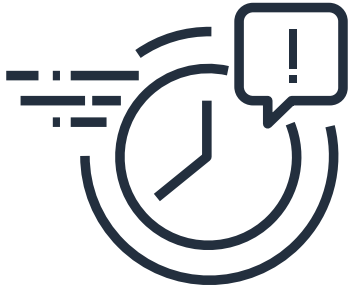
Implementing Monitoring, Elasticity, and High Availability

The need for reactive architectures

Applications need to handle large amounts of data with sub-second response times while having close to 100 percent uptime.



Elastic



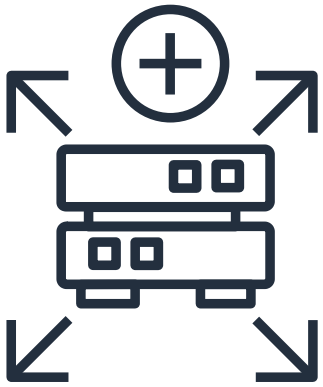
Responsive

Resilient

Message-driven

Achieve elasticity with scaling

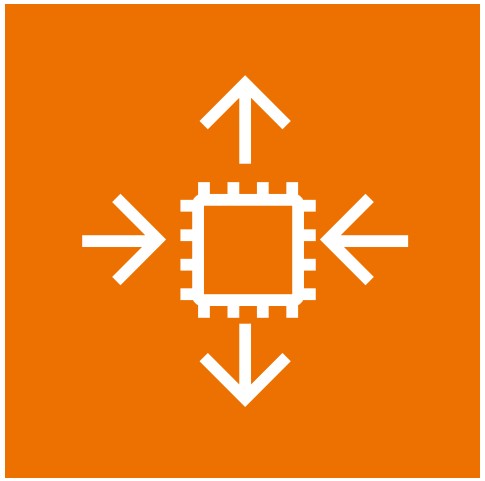
Elasticity is the ability of infrastructure to expand and contract as capacity requirements change.



Scaling is the ability to increase or decrease the compute capacity of your application.

- Vertical scaling replaces a resource with a different-sized resource.
- Horizontal scaling adds or removes resources available to the application.

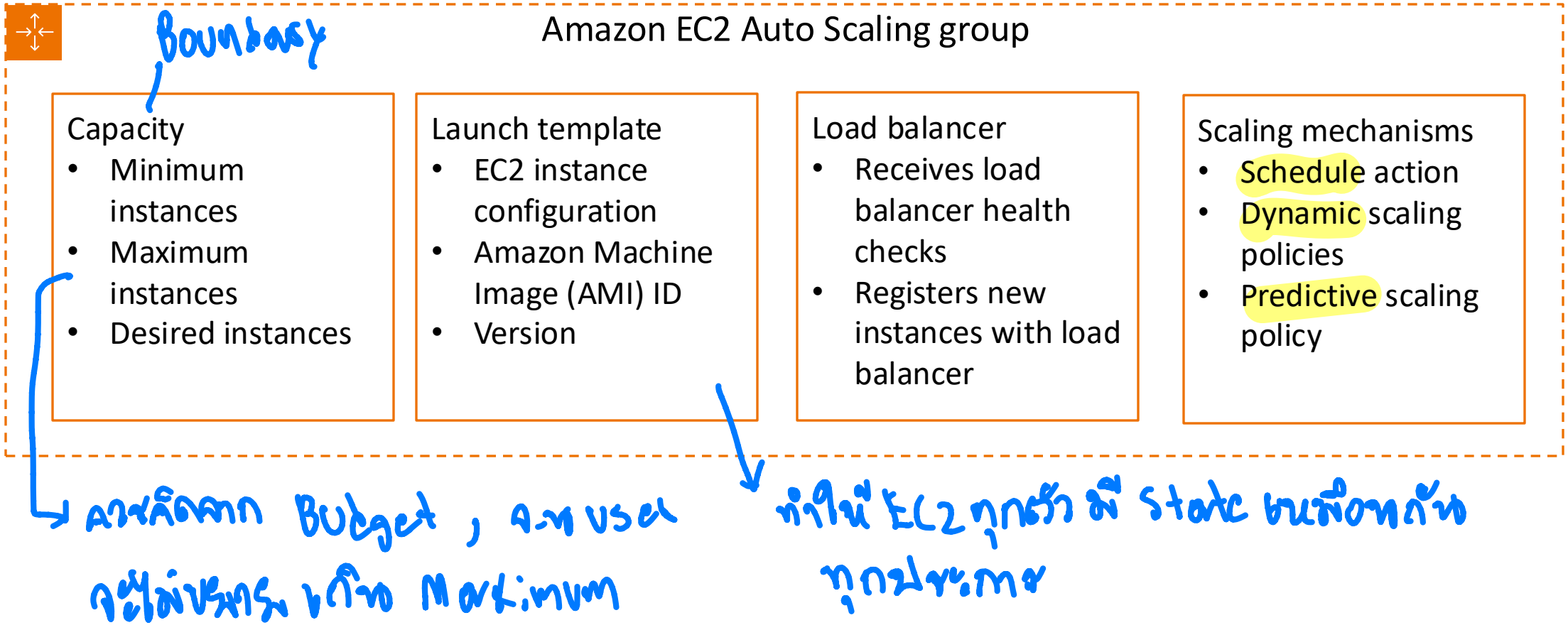
Amazon EC2 Auto Scaling



Amazon EC2 Auto Scaling

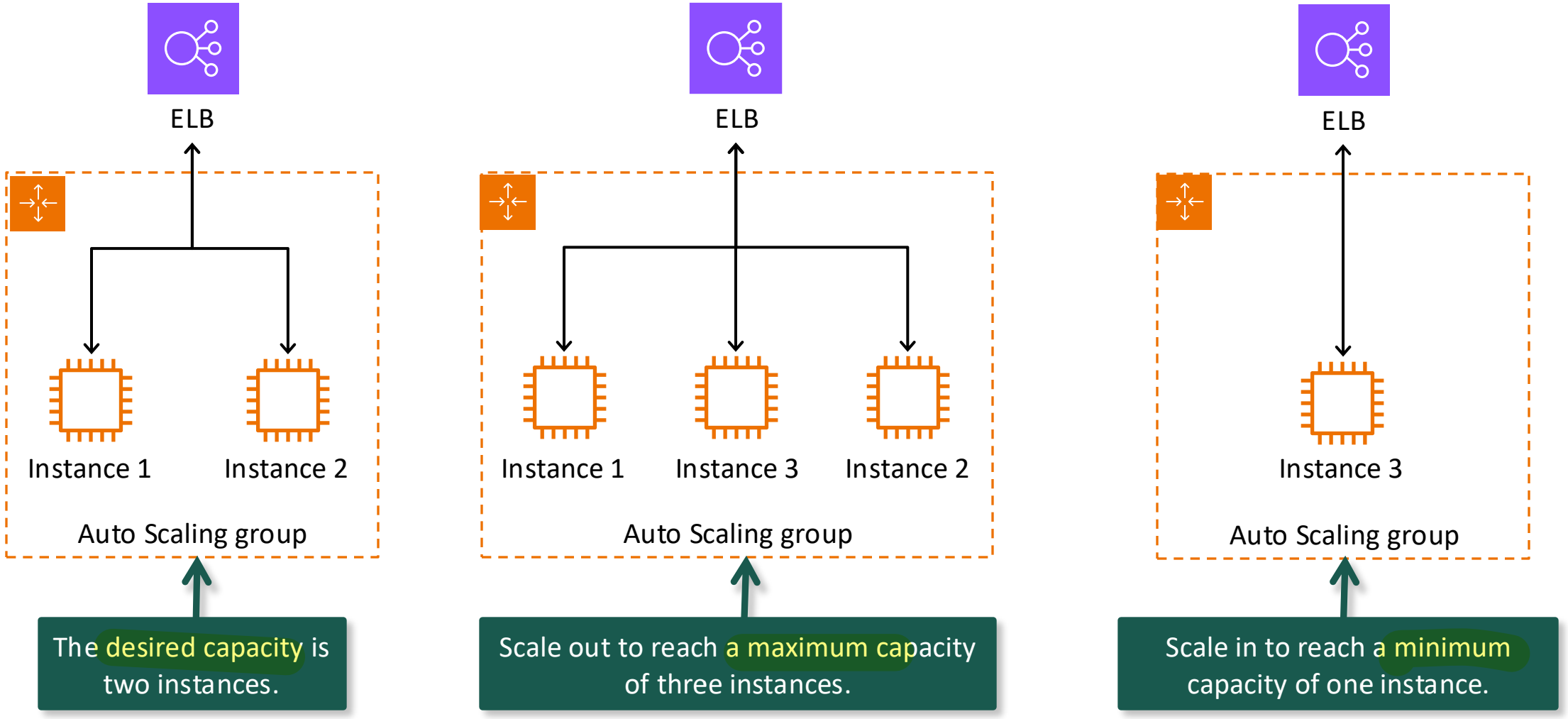
- Manages a logical collection of Amazon Elastic Compute Cloud (Amazon EC2) instances called an Amazon EC2 Auto Scaling group across Availability Zones
- Launches or retires EC2 instances configured by launch templates
- Resizes based on events from scaling policies, load balancer health check notifications, or schedule actions
- Integrates with Elastic Load Balancing (ELB) to send new instances registrations and receive health notifications
- Balances the number of instances across Availability Zones
- Is available free of charge

Amazon EC2 Auto Scaling group components

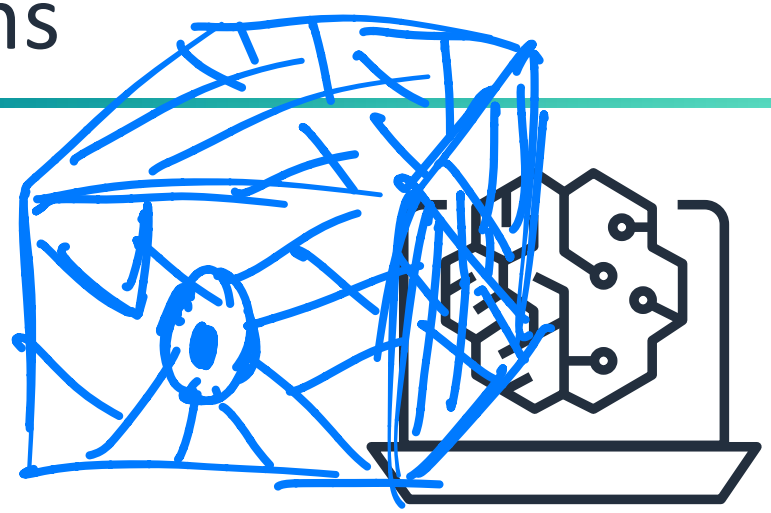


Horizontal scaling with Amazon EC2 Auto Scaling groups

Group desired capacity = 2, maximum capacity = 3, minimum capacity = 1 → Boundary



Amazon EC2 Auto Scaling mechanisms

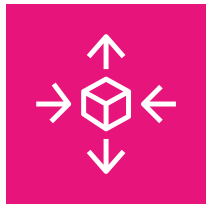


Scheduled actions	Dynamic policies	Predictive policy
• Scale based on a date and time • Are for predictable workloads	• Scale based on tracked metrics • Are for moderately spiky workloads	• Scales based on previous traffic patterns with machine learning • Is for workload traffic that can be predicted with a pattern

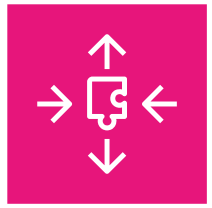
Scaling with a step scaling policy

Metric value	Less than 30%	30–39%	40–60%	61–70%	71–100%
Percentage capacity change	-30%	-10%	No change	+10%	+30%

More AWS scaling options



→ ซึ่ง fully managed service



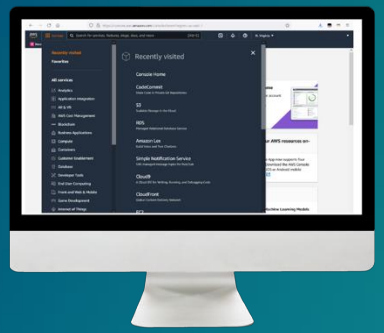
AWS Auto Scaling

- Use a scaling plan to configure auto scaling for multiple resources.
- Scale multiple AWS services:
  - Amazon Aurora
  - Amazon EC2 Auto Scaling
  - Amazon Elastic Container Service (Amazon ECS)
  - Amazon DynamoDB

AWS Application Auto Scaling

- Scale multiple resources with target tracking, step scaling, or scheduled scaling.
- Scale multiple AWS services:
  - AWS Auto Scaling services
  - AWS Lambda functions
  - Amazon SageMaker
  - Amazon ElastiCache for Redis

Demo: Creating Scaling Policies for Amazon EC2 Auto Scaling



- This demonstration uses Amazon EC2 Auto Scaling.
- In this demonstration, you will see how to do the following:
  - Create and launch an EC2 instance.
  - Create an Auto Scaling group.
  - Create a scaling policy.

Key takeaways: Scaling your compute resources

- With Amazon EC2 Auto Scaling, you can create a group to manage a logical collection of EC2 instances, called an Amazon EC2 Auto Scaling group.
- A group has capacity settings that specify the minimum, maximum, and desired number of instances required to run an application.
- Group size can be scaled in and out with schedule actions, dynamic policies, and predictive policies.
- To scale more services than EC2 instances, use AWS Auto Scaling or Application Auto Scaling.

Scaling your databases

Implementing Monitoring, Elasticity, and High Availability

Scaling AWS databases

Aurora

Scale a cluster

- Vertical scaling with instance types
- Horizontal scaling with read replicas
- Service managed storage scaling

Amazon Aurora Serverless

Scale a cluster

- Horizontal compute scaling with Aurora Compute Units (ACUs)
- Service managed storage scaling

Amazon RDS

Scale a database

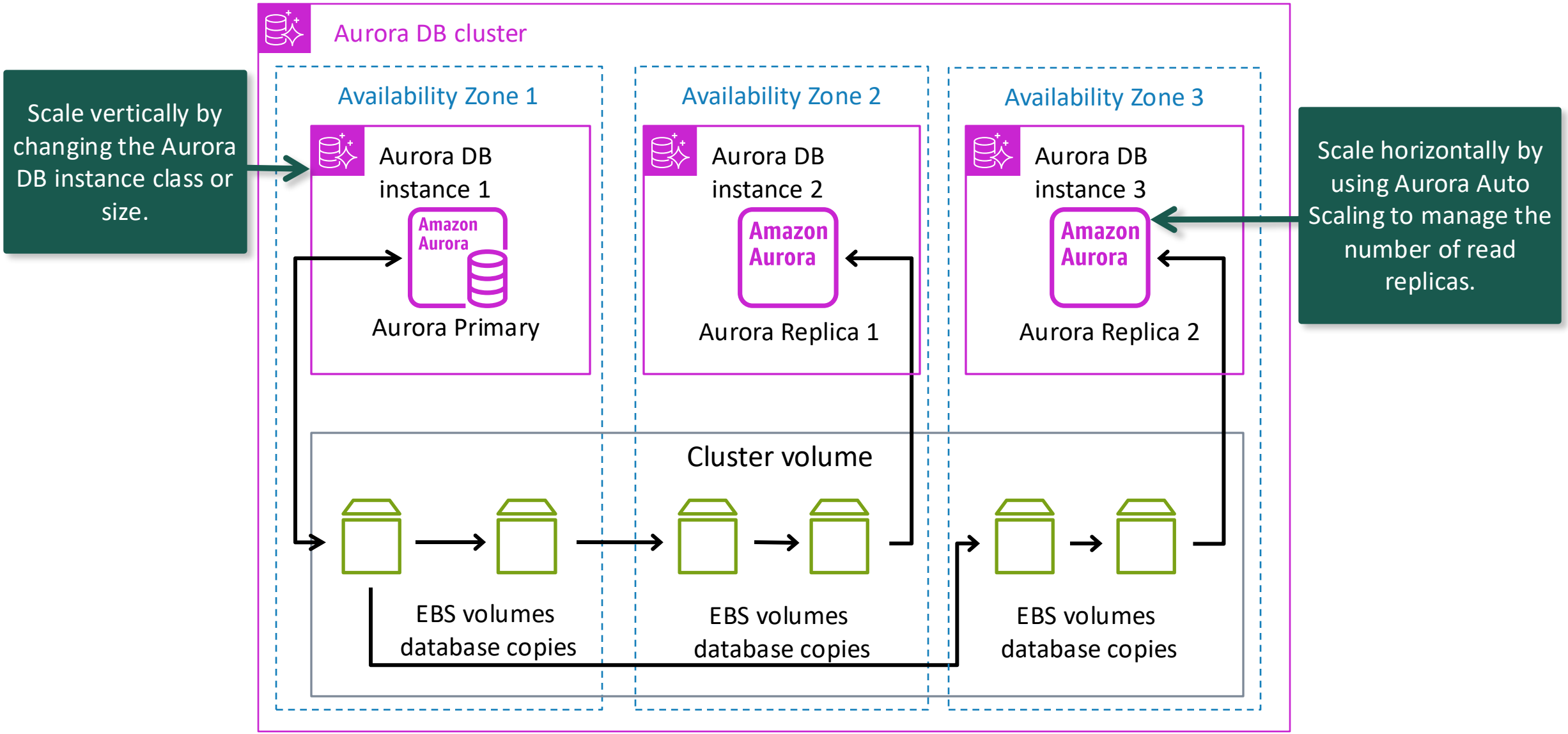
- Vertical scaling with instance types and storage volume size and type
- Horizontal scaling with read replicas

DynamoDB

Scale a table

- Horizontal scaling to scale read capacity units (RCUs) and write capacity units (WCUs) separately
- Service managed storage scaling

Scaling an Aurora cluster

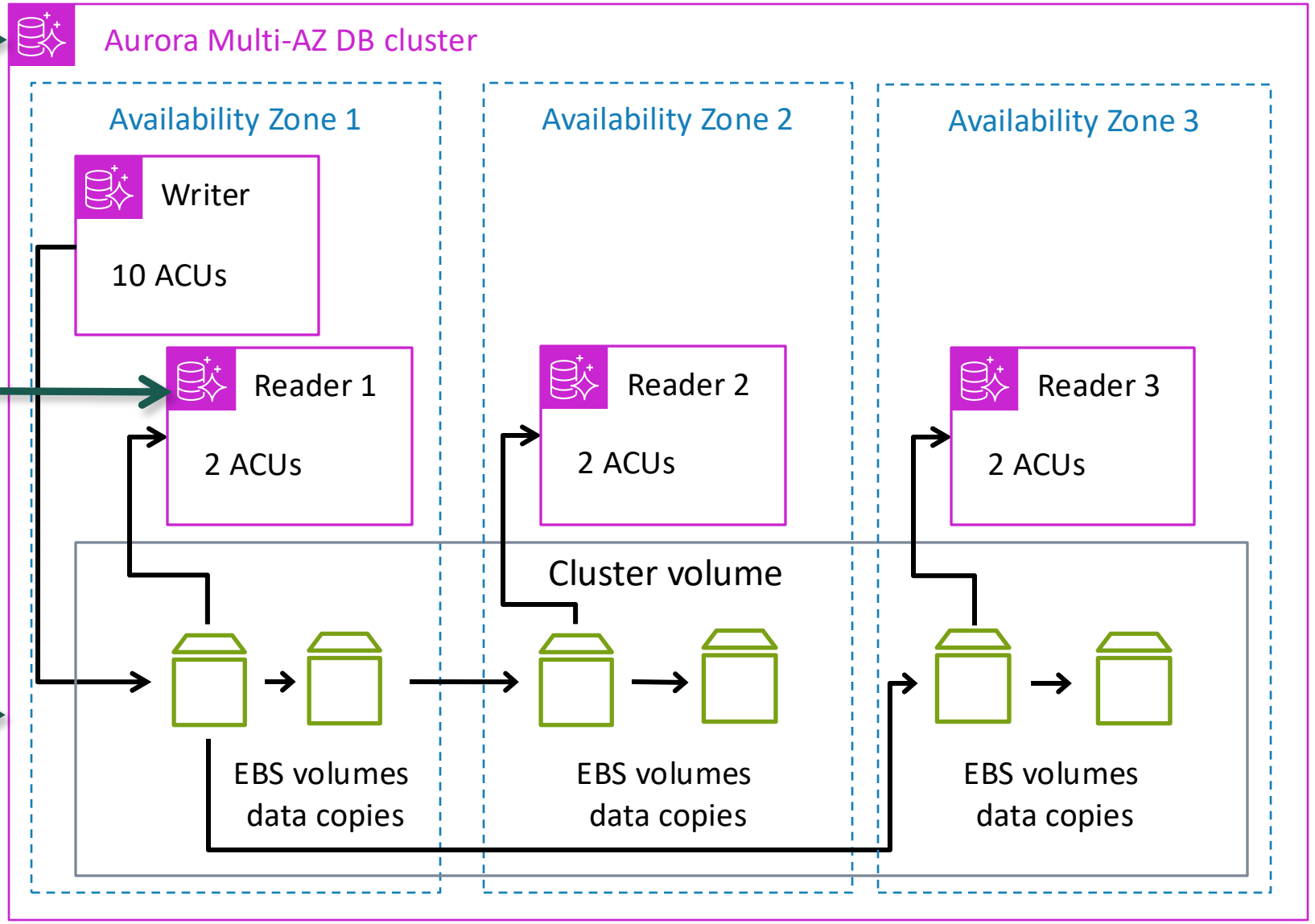


Scaling an Aurora Serverless cluster

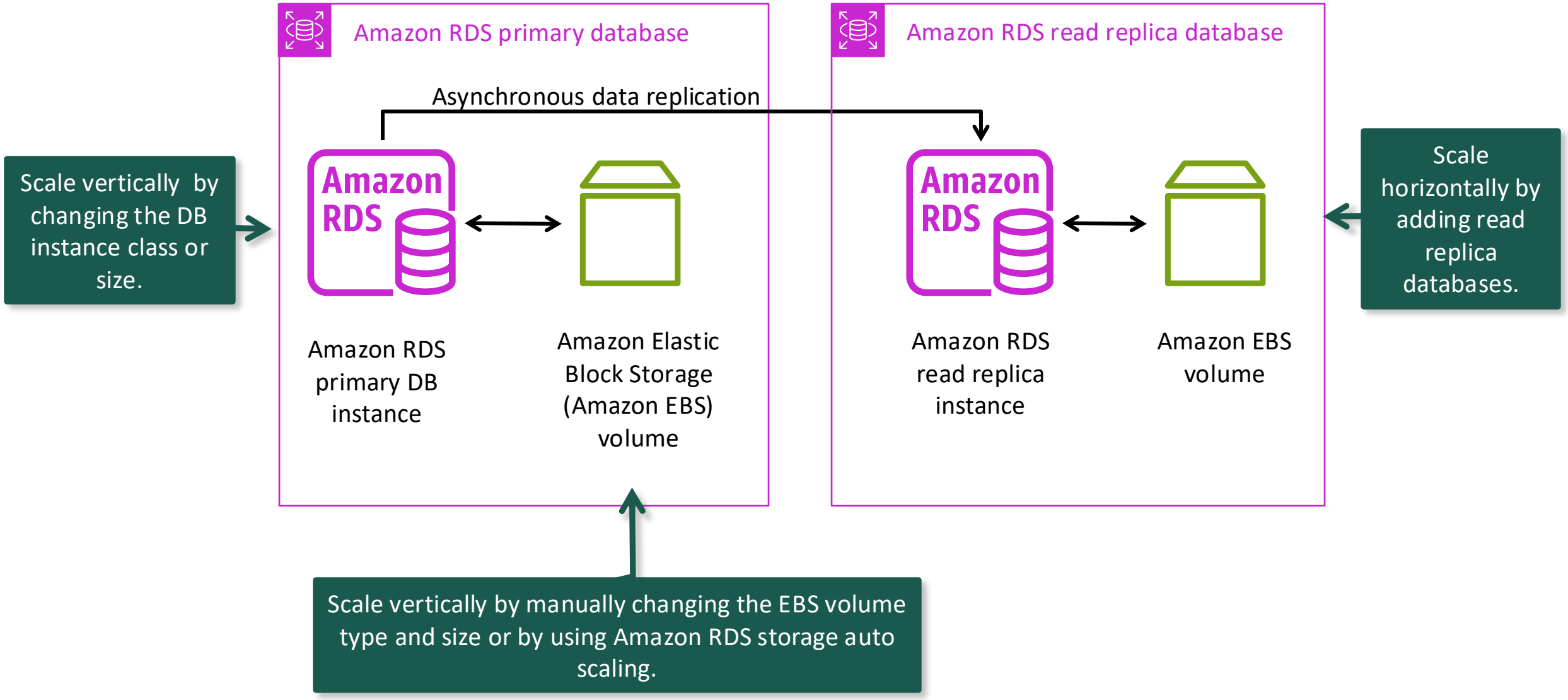
Implement Aurora Automated horizontal scaling by specifying the minimum and maximum cluster Aurora Capacity Units (ACUs).

Each Reader and Writer scales between a minimum and maximum cluster ACUs.

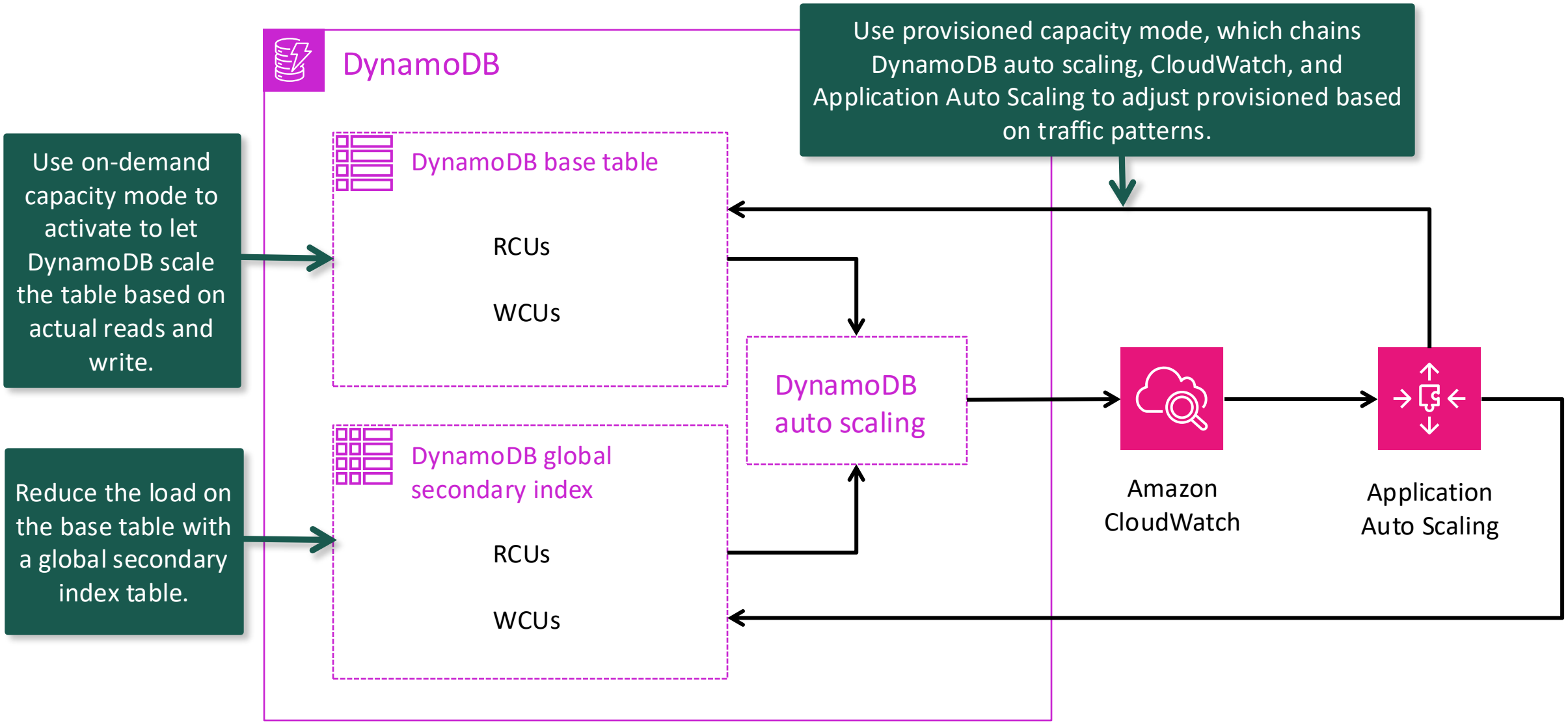
Start and stop a cluster for intermittent workloads.



Scaling an Amazon RDS database



Scaling a DynamoDB table



Comparing AWS database scaling mechanisms

Scaling	Aurora	Aurora Serverless	Amazon RDS	DynamoDB
Scope	Cluster	Cluster	Database	Table
Vertical scaling	Instance class or size	Cluster ACU throughput limits with each Reader and Writer automatically scaled within limits	• Instance class or size • Amazon RDS storage auto scaling	N/A
Horizontal scaling	Aurora Auto Scaling for read replicas	• Multi-AZ replicas • Reader replicas	Read replica databases	• On-demand mode • Provisioned mode with Application Auto Scaling • Global secondary indexes
Managed by AWS service	Storage scaling	Storage scaling	N/A	Storage scaling

Key takeaways: Scaling your databases

- With Aurora, you can choose the database instance class size and number of Aurora Replicas.
- Aurora Serverless scales resources automatically based on the minimum and maximum capacity specifications.
- You can manually vertically scale compute capacity for your RDS DB instance.
- You can use read replicas to horizontally scale your RDS DB instance.
- DynamoDB On-Demand offers a pricing model based on actual table reads and writes.
- DynamoDB auto scaling uses Application Auto Scaling to dynamically adjust provisioned throughput capacity.

Using load balancers to create highly available environments

Implementing Monitoring, Elasticity, and High Availability

Highly available systems

Percentage of uptime	Maximum downtime per year	Equivalent downtime per day
90%	36.5 days	2.4 hours
99%	3.65 days	14 minutes
99.9%	8.76 hours	86 seconds
99.99%	52.6 minutes	8.6 seconds
99.999%	5.25 minutes	0.86 seconds

ELB

เป็น fully managed

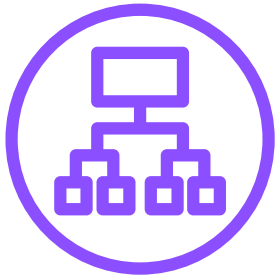
ELB

- Region service → อยู่ใน VPC (เลือกได้ ว่าจะอยู่ subnet ไหนบ้าง AZ)
- Distributes traffic across multiple targets in one or more Availability Zone
- Can receive public or private traffic
- Monitors the health of registered targets with health checks

↓ Health check ไม่ผ่าน = ไม่ส่งให้

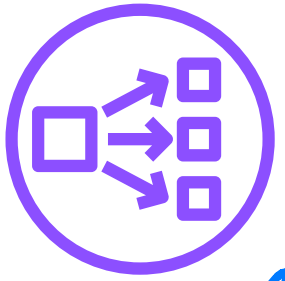
- Routes traffic to only healthy targets
- Scales based on incoming traffic

Types of AWS load balancers



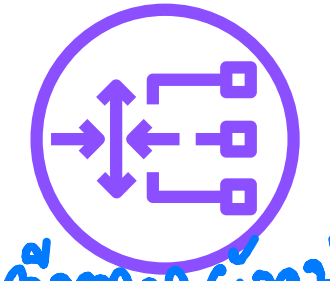
Application Load Balancer

- Is used for HTTP and HTTPS traffic
- Operates at OSI layer 7, the application layer
- Is used for application architectures



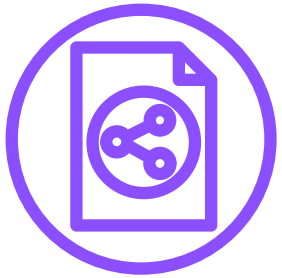
Network Load Balancer

- Is used for TLS offloading, UDP, and static IP addresses
- Operates at OSI layer 4, the transport layer
- Is used for millions of requests per second at ultra-low latency



Gateway Load Balancer

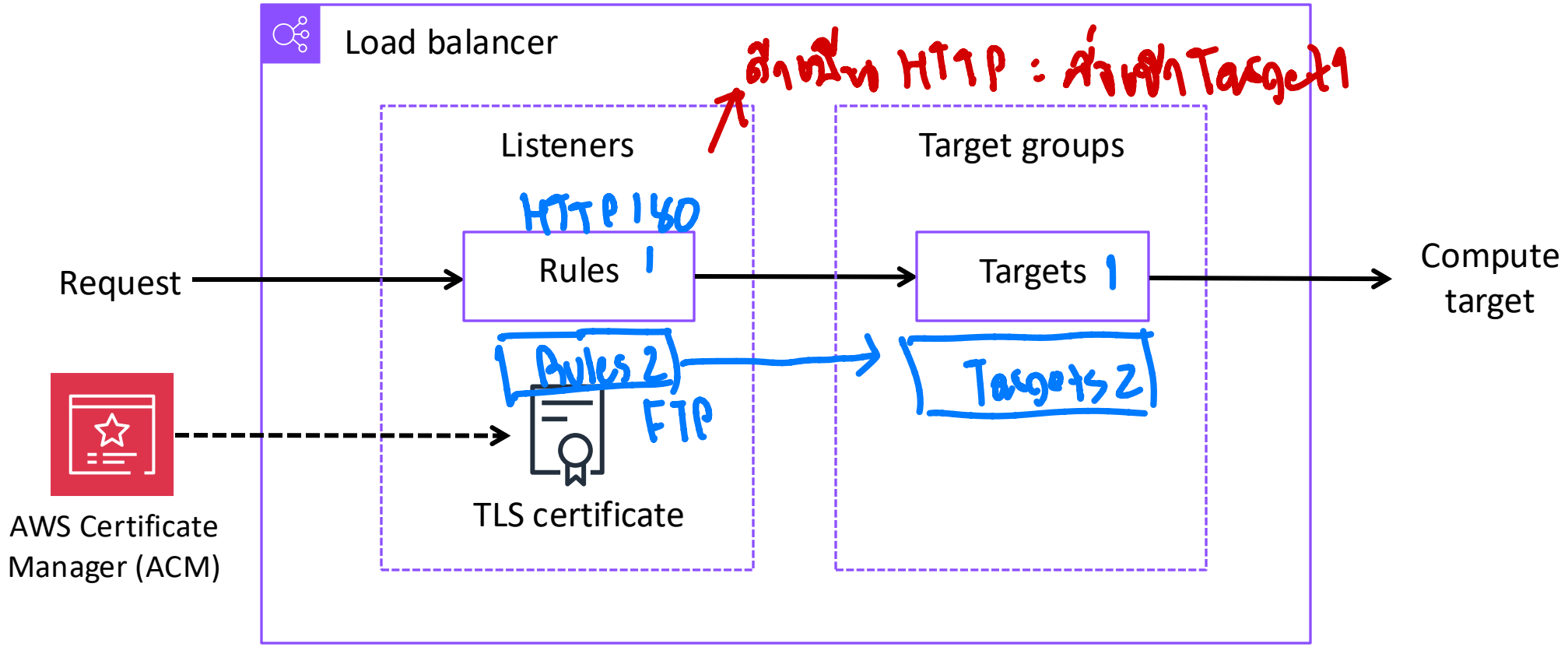
- Is used for third-party virtual appliance fleet using GENEVE protocol
- Operates at OSI layer 3, the network layer
- Is used to improve security, compliance, and policy controls



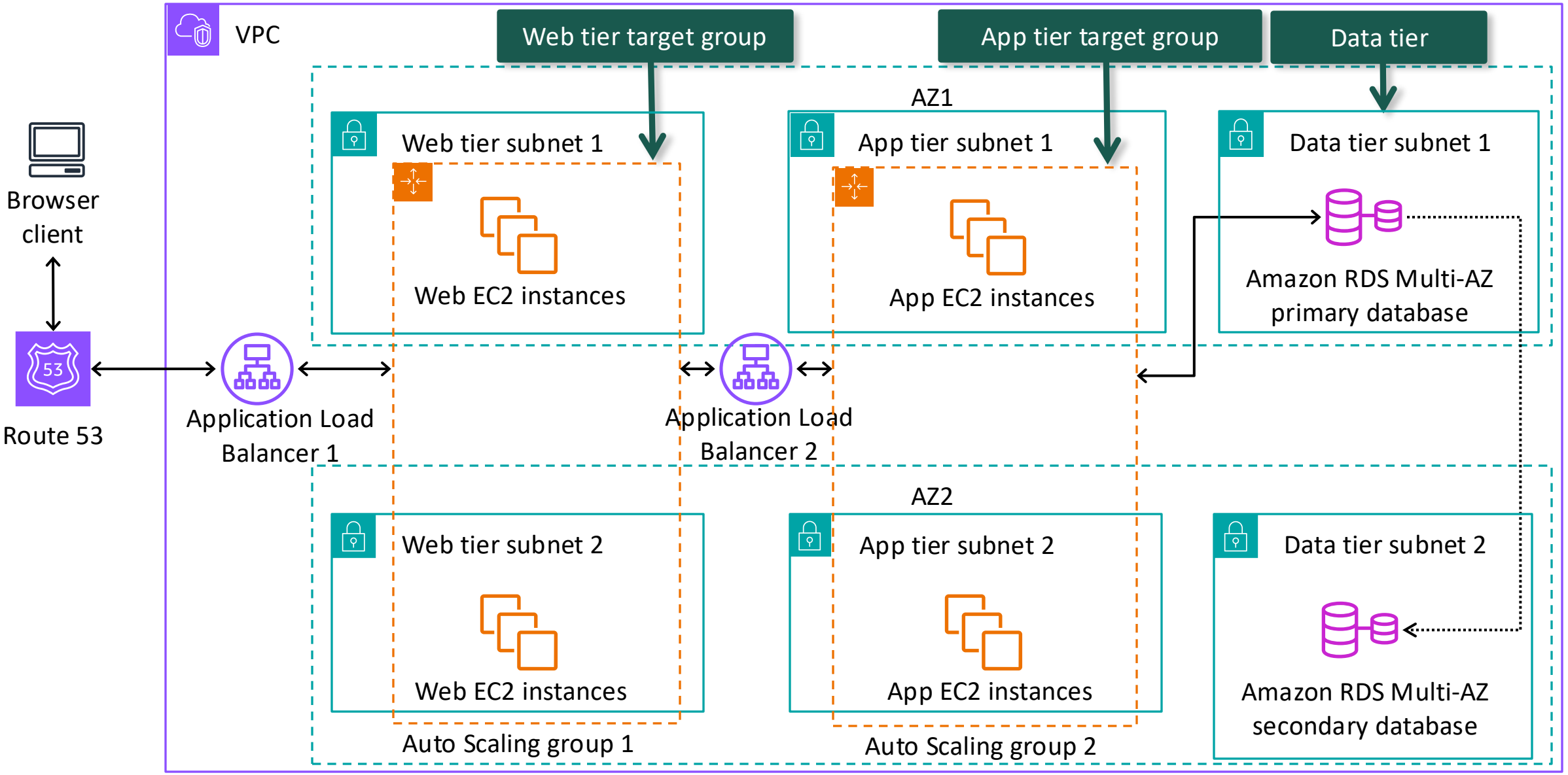
Classic Load Balancer

- Is used for previous generation EC2-Classic networks
- Operates at OSI layers 3 and 7, the transport and application layers
- Is used if upgrading to other load balancers is not feasible

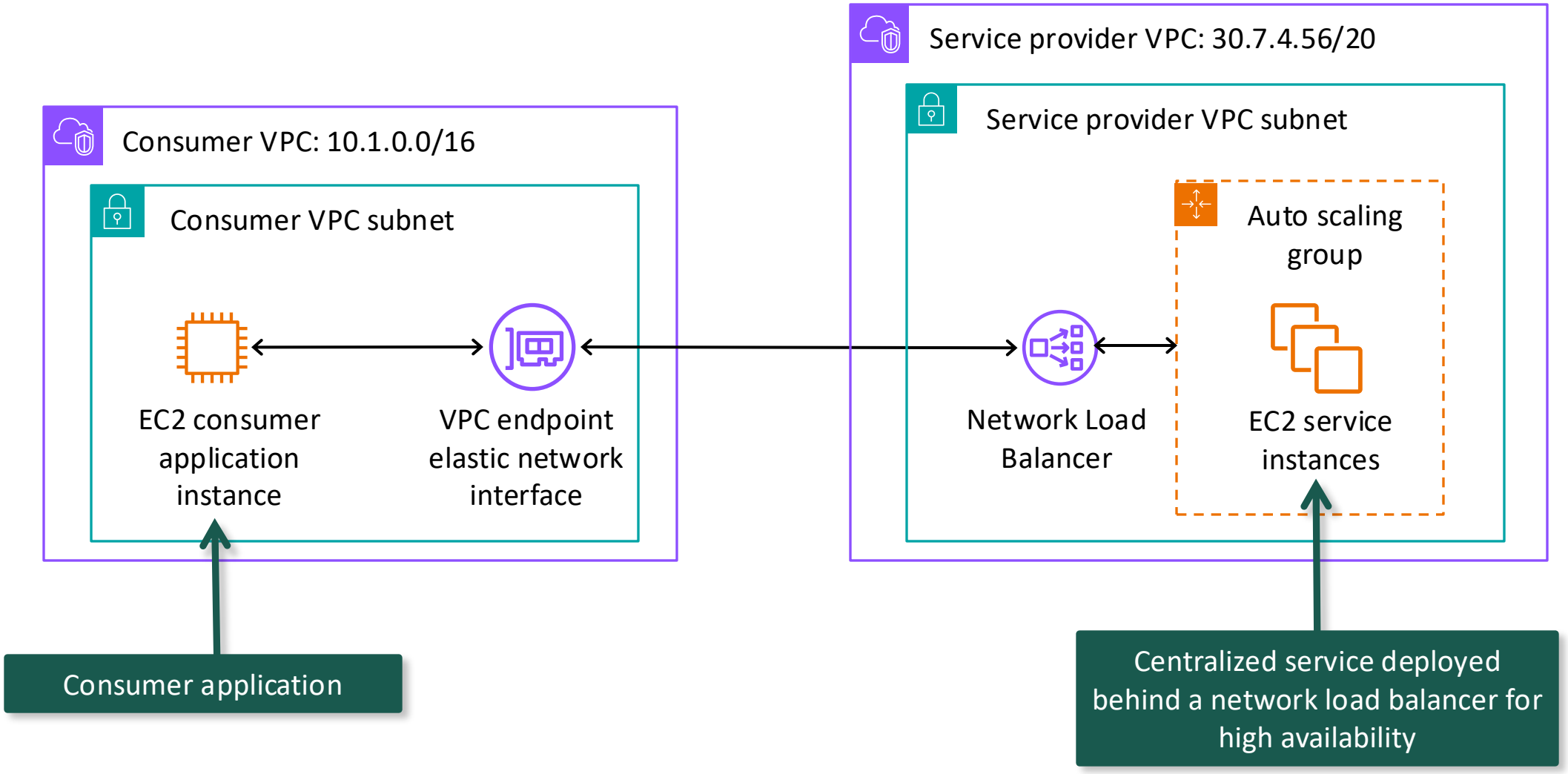
Load balancer components



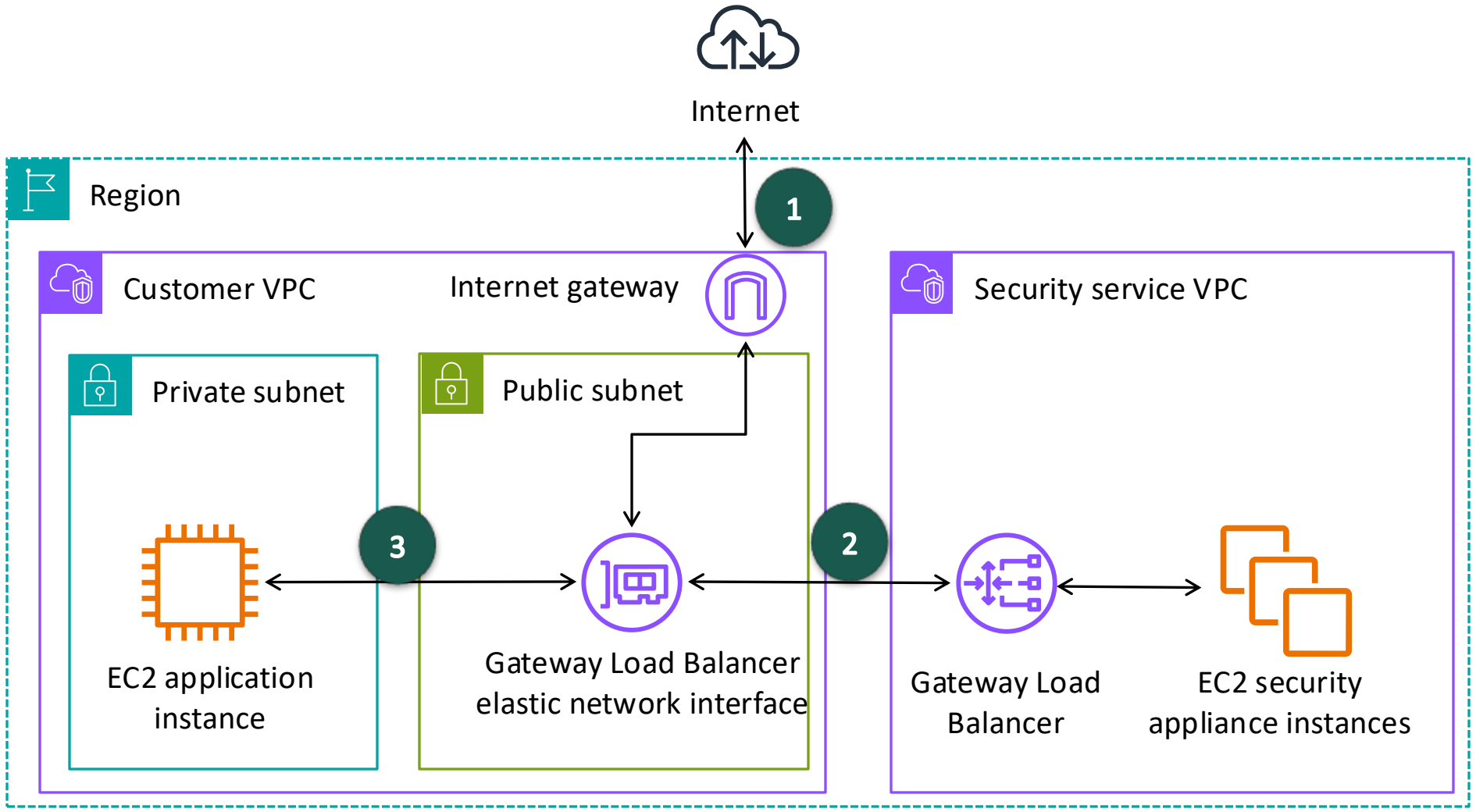
High availability by using an Application Load Balancer and Amazon RDS Multi-AZ



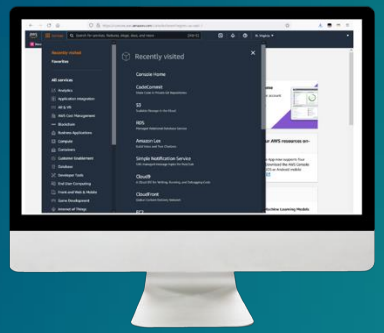
High availability by using a Network Load Balancer



Security service by using a Gateway Load Balancer



Demo: Creating a Highly Available Application



- This demonstration uses Amazon EC2.
- In this demonstration, you will see how to do the following:
  - Create a highly available application with Elastic Load Balancing and Auto Scaling groups.
  - Create an Application Load Balancer.

Key takeaways: **Using load balancers to create highly available environments**

- ELB distributes traffic across multiple targets in one or more Availability Zones and monitors the health of registered targets with health checks.
- An Application Load Balancer is used for application architectures and operates at the OSI model application layer (layer 7).
- A Network Load Balancer is used for millions of concurrent, ultra-low latency requests and operates at the OSI model transport layer (layer 4).
- A Gateway Load Balancer is used to improve security, compliance, and policy controls and operates at the OSI model network layer (layer 3).

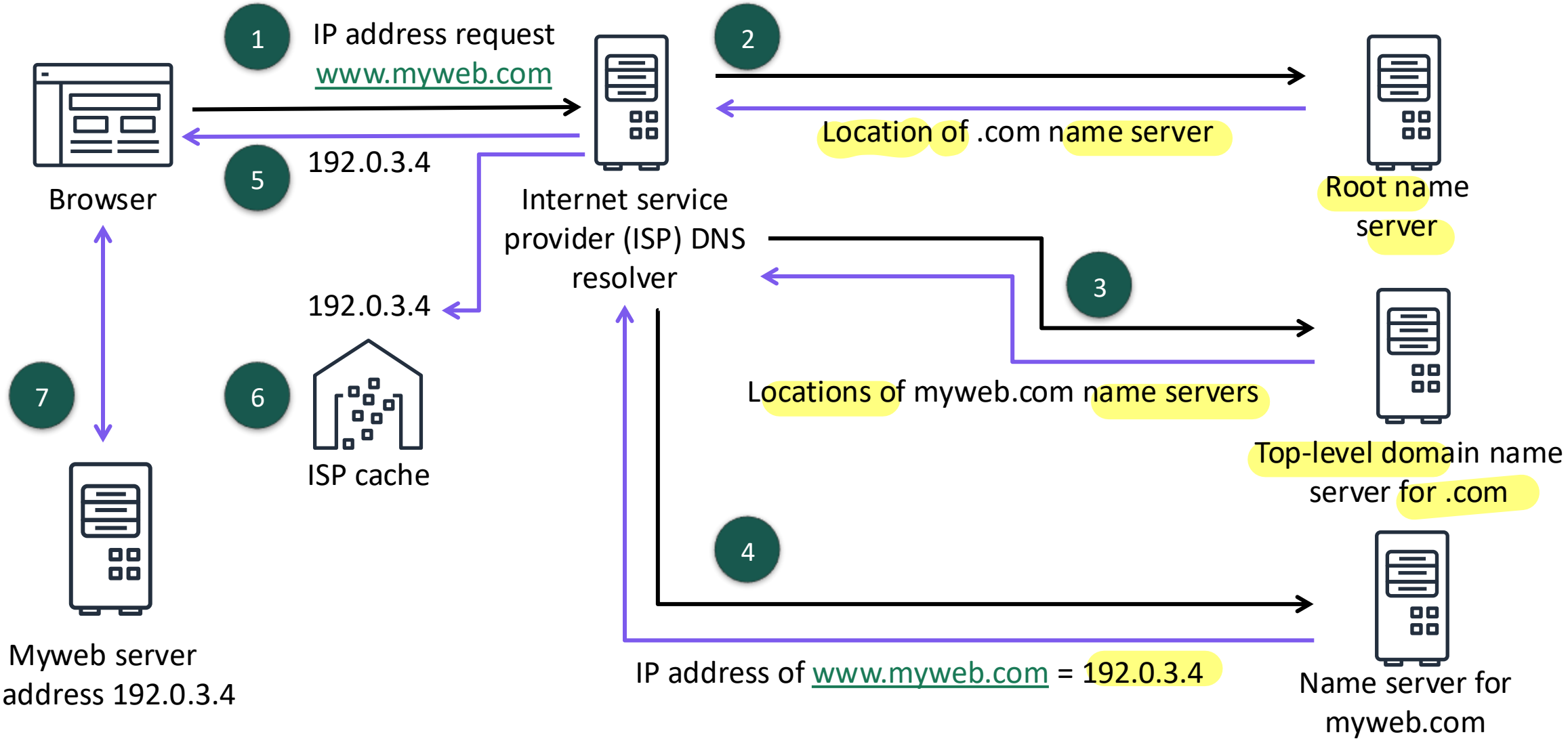
DNS (Fully Managed service)

Using Route 53 to create highly available environments

Implementing Monitoring, Elasticity, and High Availability

DNS lookups

หน้าที่ Resolve DN → IP

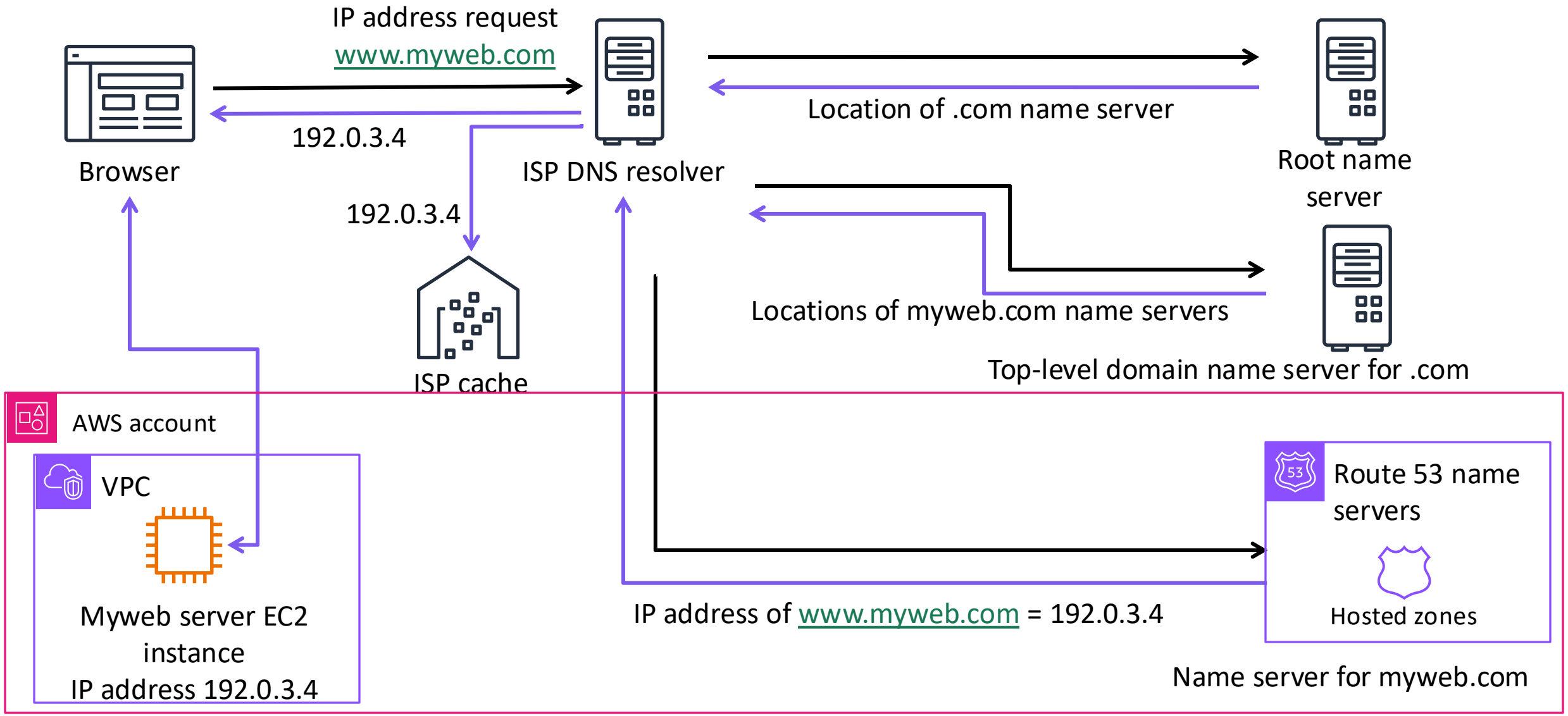


Route 53

Route 53

- Is a DNS web service that manages domain name registrations and provides hosted zones.
- Provides authoritative name servers for DNS resolution
- Performs DNS routing to route traffic to healthy endpoints
- Performs health checks against IP addresses or domains to manage Availability Zone or Regional failovers
- Can monitor CloudWatch alarms
- Supports multiple routing options

DNS lookups with Route 53

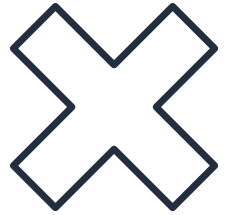


Route 53 routing policies

Simple

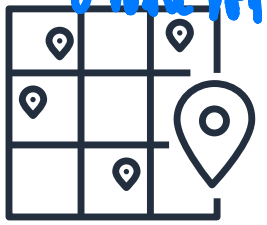
Weighted

Latency



Failover

Geoproximity

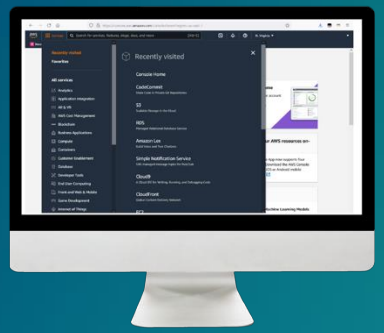


Geolocation

Multivalue answer

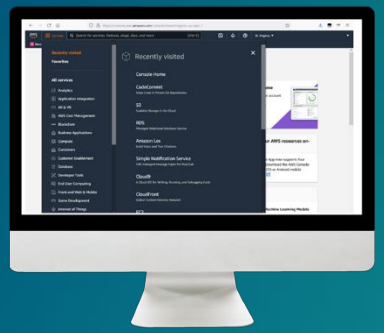
IP-based

Demo: Amazon Route 53: Simple Routing



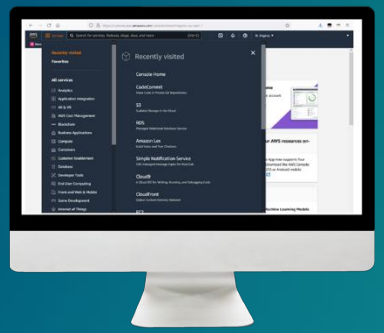
- This demonstration uses Amazon Route 53.
- In this demonstration, you will see how to do the following:
  - Configure Route 53 to evenly distribute HTTP request between two EC2 instances.
  - Define a Route 53 record set.

Demo: Amazon Route 53: Failover Routing



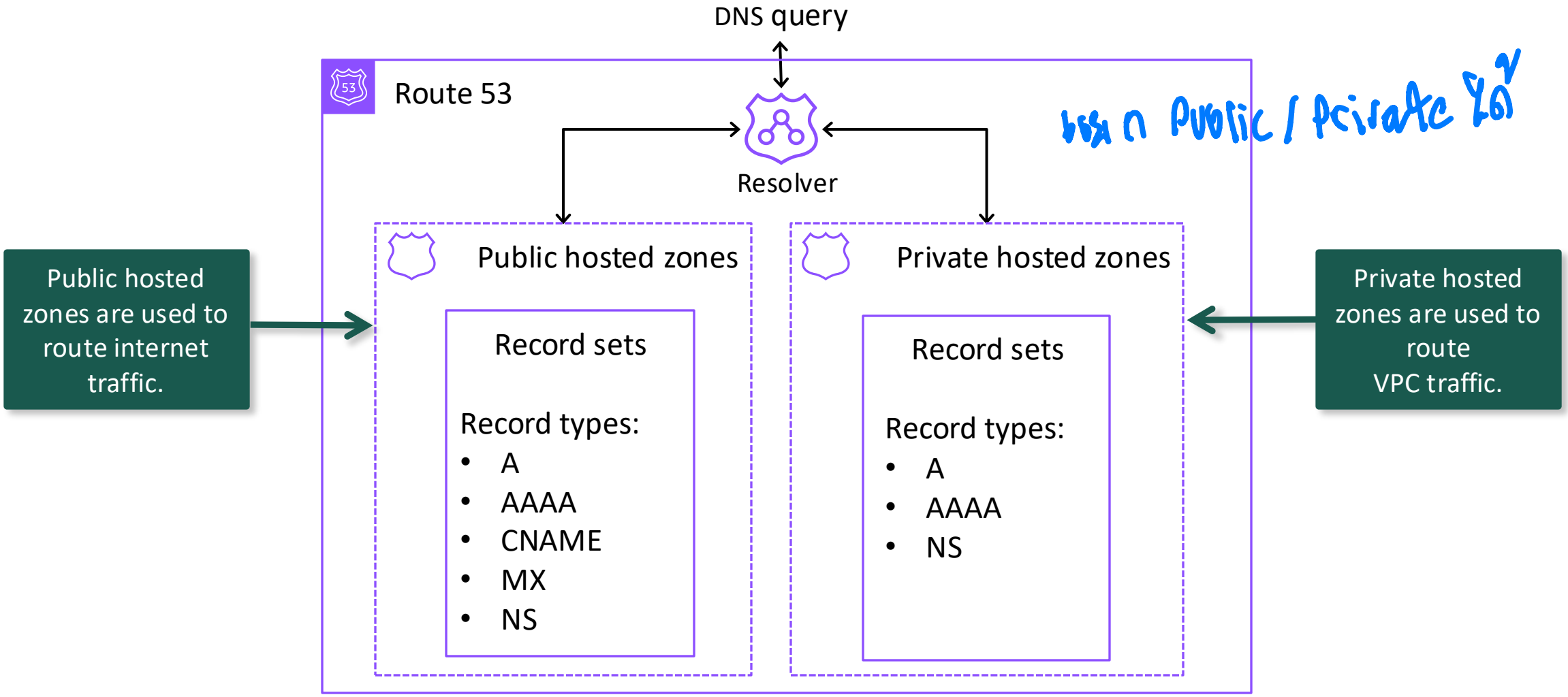
- This demonstration uses Amazon Route 53.
- In this demonstration, you will see how to do the following:
  - Use Route 53 to direct traffic to two availability zones for use in failures.
  - Define a Route 53 Health Check.
  - Send Route 53 Health Check alerts to an administrator.

Demo: Amazon Route 53: Geolocation Routing

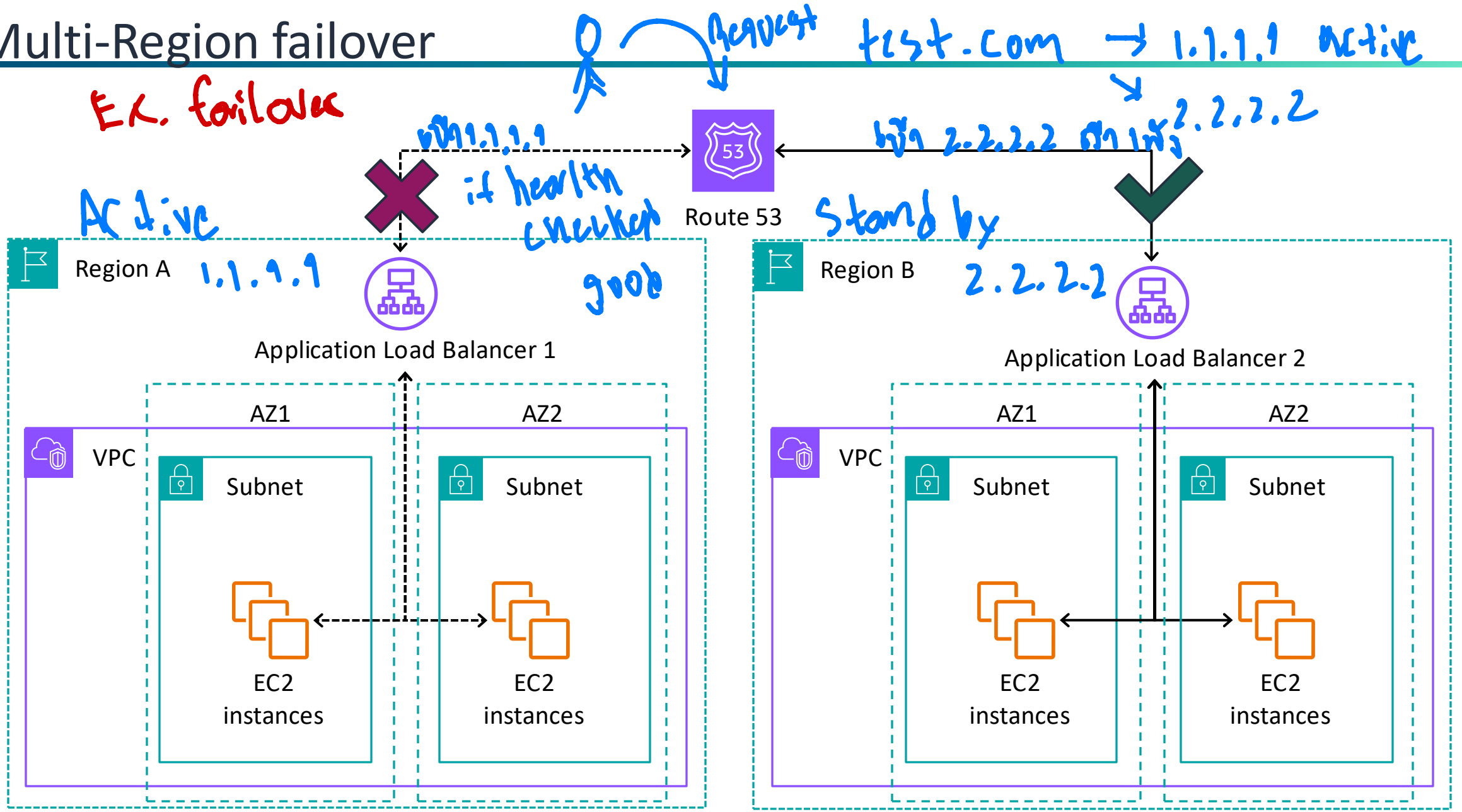


- This demonstration uses Amazon Route 53.
- In this demonstration, you will see how to do the following:
  - Use geolocation routing to serve traffic based on geographic location.
  - Localize content for a geographic groups of users.

Route 53 public and private hosted zones



Multi-Region failover



Key takeaways: **Using Route 53 to create highly available environments**

- Route 53 is a DNS service that does the following:
  - Manages domain name registrations
  - Provides hosted zones and authoritative name servers
  - Performs DNS routing and health checks
- Route 53 supports multiple routing options, including the following:
  - Simple routing
  - Weighted routing
  - Latency routing
  - Failover routing
  - Geoproximity routing
  - Geolocation routing
  - Multivalue answer routing
  - IP-based routing

Guided Lab:
Creating a Highly Available Environment (High availability lab)

High availability lab tasks:

- In this lab, you perform the following main tasks:
  - Inspect a provided VPC.
  - Create an Application Load Balancer.
  - Create an Amazon EC2 Auto Scaling group by using a launch template.
  - Update security groups.
  - Test the application.
  - Test the application for high availability.
- This lab includes the following optional tasks:
  - Make the database highly available.
  - Configure a highly available NAT gateway.

Debrief: High availability lab

- How did you determine which instances your Application Load Balancer should send traffic to?
- How did you determine the characteristics of the instances that would be launched in your Auto Scaling group?

Café lab: Creating a Scalable and Highly Available Environment for the Café (High availability café lab)

The evolving café architecture: version 6

Architecture Version	Business reason for update	Technical Requirements and Architecture Update
V1	Create a static website for a small business.	Host the website on Amazon S3.
V2	Add online ordering.	Deploy the web application and database on Amazon EC2.
V3	Reduce the effort to maintain the database and secure its data.	Separate the web and database layers. Migrate the database to Amazon RDS on a private subnet.
V4	Enhance the security of the web application.	Use Amazon VPC features to configure and secure public and private subnets.
V5	Create separate access mechanisms based on role.	Add IAM groups and attach resource polices to application resources. Add IAM users to groups based on role.
V6	Ensure that the website can handle an expected increase in traffic.	Add a load balancer, implement auto scaling on the EC2 instances, and distribute compute and database instances across two Availability Zones.
V7	Module 11 info	
V8	Module 14 info	

High availability Café lab tasks

- In this lab, you will perform the following main tasks:
  - Add a load balancer
  - Implement auto scaling on the EC2 instances
  - Distribute compute and database instances across two Availability Zones
- Open your lab environment to start the lab and find additional details about the tasks that you will perform

Debrief: High availability café lab

- Where did you determine the AMI, instance type, key pair, and security groups that would be used to launch EC2 instances in your Auto Scaling group?
- When you conducted a stress test on your instances, why did the Auto Scaling group launch more instances?

Applying AWS Well-Architected Framework principles to highly available systems

Implementing Monitoring, Elasticity, and High Availability

Well-Architected Framework best practices for highly available systems

Best practice approach: Failure management – use fault isolation to protect your workload

Best practices

Deploy the workload to multiple locations.

Automate recovery for components constrained to a single location.

Best practice approach: Failure management – design your workload to withstand component failures

Best practices

Fail over to healthy resources.

Cloudwatch alarm

Send notifications when events impact availability.

Best practice approach: Compute and hardware

Best practices

Scale your compute resources dynamically.

Key takeaways: Applying AWS Well-Architected Framework principles to high availability

- Deploy the workload to multiple locations.
- Automate recovery for components constrained to a single location.
- Fail over to healthy resources.
- Send notifications when events impact availability.
- Scale your compute resources dynamically.

Module wrap-up

Implementing Monitoring, Elasticity, and High Availability

Module summary

This module prepared you to do the following:

- Examine how reactive architectures use Amazon CloudWatch and Amazon EventBridge to monitor metrics and facilitate notification events.
- Use Amazon EC2 Auto Scaling within an architecture to promote elasticity and create a highly available environment.
- Determine how to scale your database resources.
- Identify a load balancing strategy to create a highly available environment.
- Use Amazon Route 53 for DNS failover.
- Use the AWS Well-Architected Framework principles when designing highly available systems.

Considerations for the café

- Discuss how the café lab in this module answered the key questions and decisions presented at the start of this module for the café business.

Module knowledge check

- The knowledge check is delivered online within your course.
- The knowledge check includes 10 questions based on the material presented on the slides and in the slide notes.
- You can retake the knowledge check as many times as you like.

Sample exam question

A web application gives customers the ability to upload orders to an Amazon S3 bucket. The resulting Amazon S3 events initiate an AWS Lambda function that inserts a message into an Amazon Simple Queue Service (Amazon SQS) queue. A single Amazon EC2 instance reads the messages from the queue, processes them, and stores them in an Amazon DynamoDB table partitioned by unique order ID. Next month, traffic is expected to increase by a factor of 10, and a solutions architect is reviewing the architecture for possible scaling problems. Which component is MOST likely to need re-architecting to be able to scale to accommodate the new traffic?

Identify the key words and phrases before continuing.

The following are the keywords and phrases:

- A single EC2 instance
- Able to scale

Sample exam question: Response choices

A web application gives customers the ability to upload orders to an Amazon S3 bucket. The resulting Amazon S3 events initiate an AWS Lambda function that inserts a message into an Amazon Simple Queue Service (Amazon SQS) queue. A single Amazon EC2 instance reads the messages from the queue, processes them, and stores them in an Amazon DynamoDB table partitioned by unique order ID. Next month, traffic is expected to increase by a factor of 10, and a solutions architect is reviewing the architecture for possible scaling problems. Which component is MOST likely to need re-architecting to be able to scale to accommodate the new traffic?

Choice	Response
A	Lambda function
B	SQS queue
C	EC2 instance
D	DynamoDB table

Sample exam question: Answer

The answer is C.

Choice	Response
C	EC2 instance

Thank you

Corrections, feedback, or other questions?
Contact us at https://support.aws.amazon.com/#/contacts/aws-academy.