CSCI 5902 Adv. Cloud Architecting
Fall 2023
Instructor: Lu Yang

Module 9 Implementing Elasticity, Hight Availability, & Monitoring (Sections 3-4) July 10, 2023

Housekeeping items and feedback

1. Start recording
2. Start your term project
   - Pre-baked application
   - Minor implementations — SG IAM

Question from the last lecture: ASG Desired Capacity

Desired capacity: Represents the initial capacity of the Auto Scaling group at the time of creation. An Auto Scaling group attempts to maintain the desired capacity. It starts by launching the number of instances that are specified for the desired capacity, and maintains this number of instances as long as there are no scaling policies or scheduled actions attached to the Auto Scaling group.

Recap of our last lecture

AWS Academy Cloud Architecting

Module 9: Implementing Elasticity, High Availability, and Monitoring

Module overview

Sections

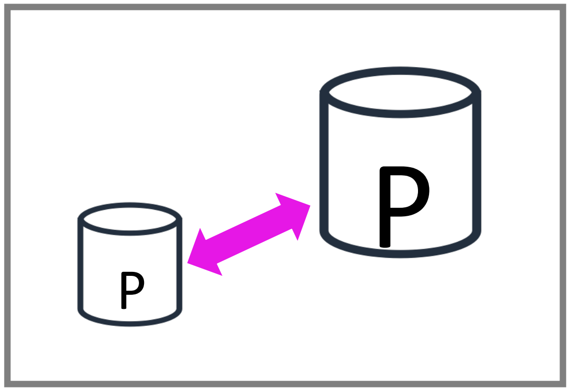
1. Architectural need
2. Scaling your compute resources
3. Scaling your databases
4. Designing an environment that’s highly available
5. Monitoring

Module 9: Implementing Elasticity, High Availability, and Monitoring

Section 3: Scaling your databases

Vertical scaling with Amazon RDS: Push-button scaling

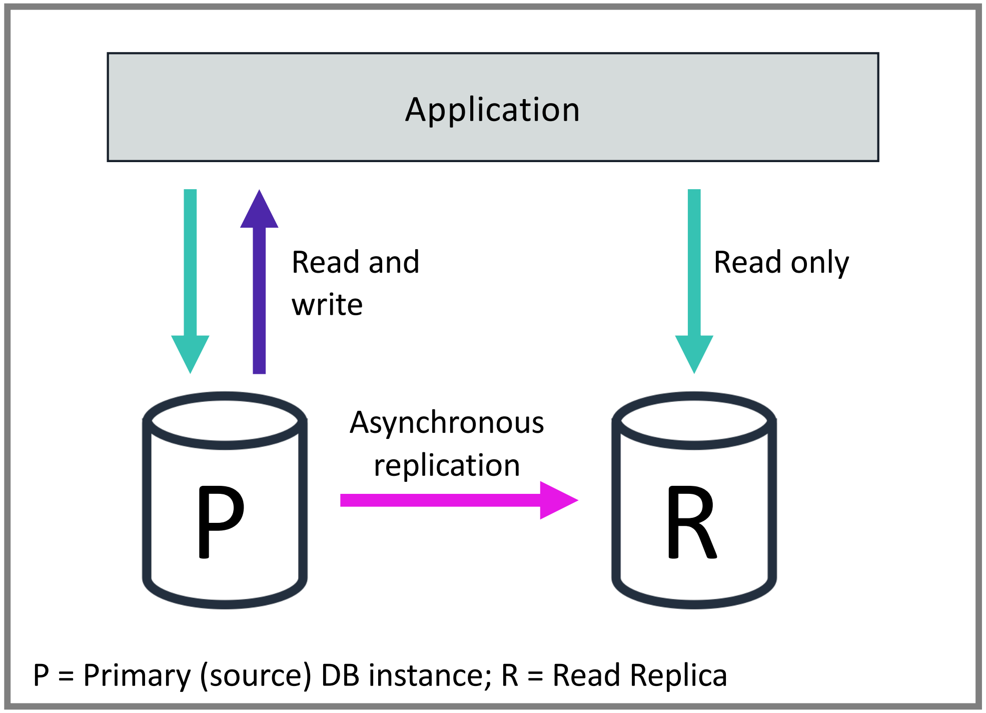
- Scale DB instances vertically up or down
- From micro to 24xlarge and everything in between
- Scale vertically with minimal downtime
  - However, multi-AZ deployment of an Amazon RDS MySQL DB instance can significantly reduce any impact. (https://aws.amazon.com/premiumsupport/knowledge-center/rds-mysql-downtime-impact/)



① instancCe class downtime

② size of storage
↓
online

Horizontal scaling with Amazon RDS: Read replicas

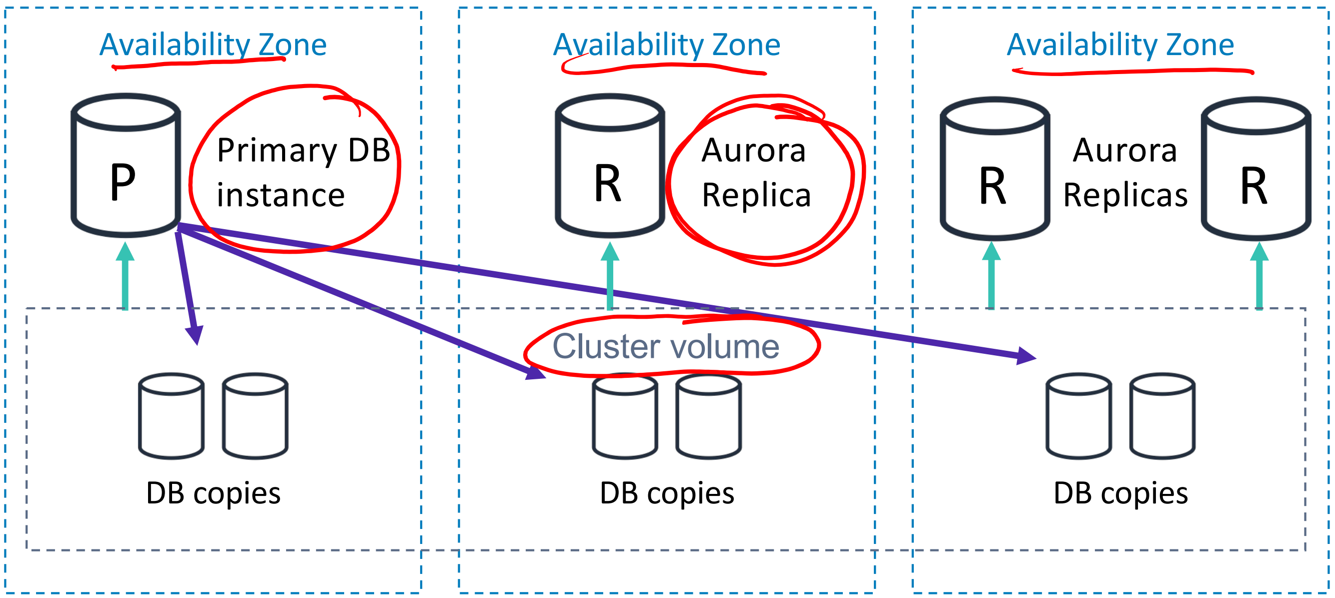


P = Primary (source) DB instance; R = Read Replica

- Horizontally scale for read-heavy workloads
- Up to 15 read replicas for each MySQL, MariaDB and PostgreSQL instance, 5 read replicas Oracle and SQL instance, and up to 15 Aurora replicas
- Replication is asynchronous
- Available for Amazon RDS for MySQL, MariaDB, PostgreSQL, SQL, and Oracle

Scaling with Amazon Aurora

Each Aurora DB cluster can have up to 15 Aurora replicas



Amazon Aurora Serverless

Responds to your application automatically:

- Scales capacity
- Starts up
- Shuts down

WCU
RCU

Pay for the number of Aurora capacity units (ACUs) that are used

Good for intermittent and unpredictable workloads

Horizontal scaling: Database sharding

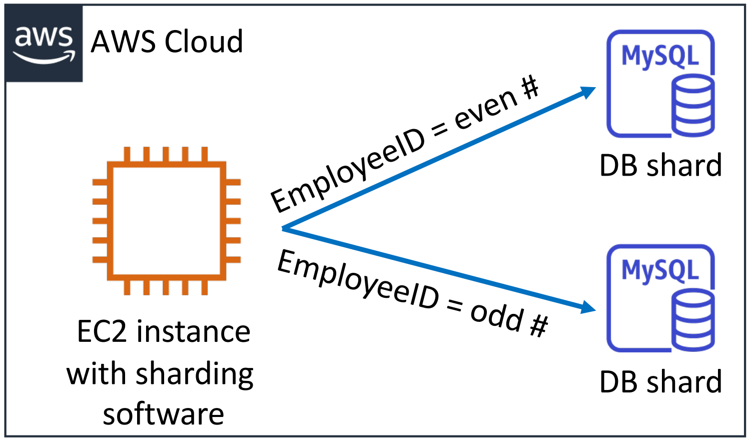
Without shards, all data resides in one partition.

- Example: Employee IDs in one database

With sharding, data is split into large chunks (shards).

- Example: Even-numbered employee IDs in one database, and odd-numbered employee IDs in another database

In many circumstances, sharding improves write performance.



Scaling with Amazon DynamoDB: On-Demand

On-Demand

Pay per request

No more provisioning

Use case: Spiky, unpredictable workloads. Rapidly accommodates to need.

Scaling with Amazon DynamoDB: Auto scaling

On-Demand

Pay per request

No more provisioning

Use case: Spiky, unpredictable workloads. Rapidly accommodates to need.

Auto scaling

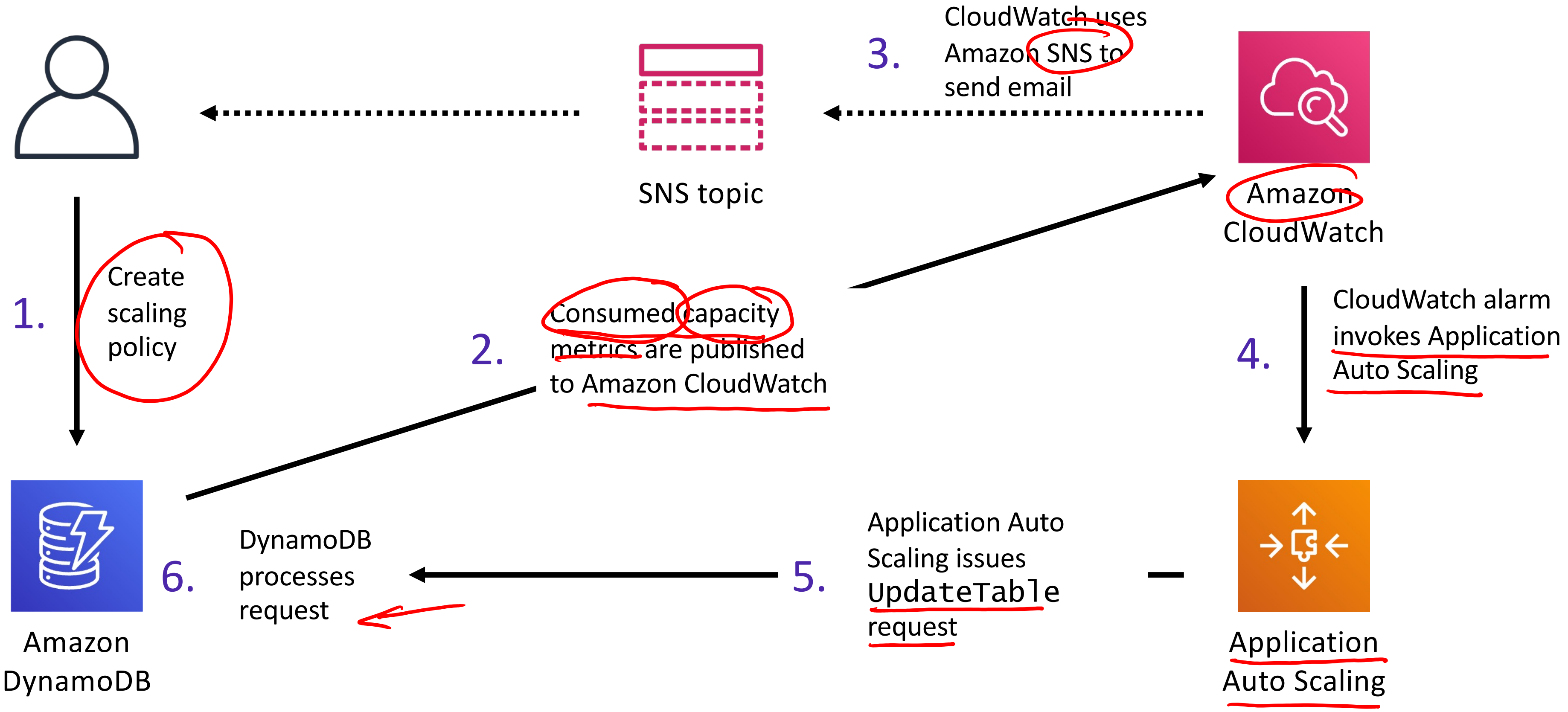
Default for all new tables

Specify upper and lower bounds

Use case: General scaling, good solution for most applications.

predictable

How to implement DynamoDB auto scaling



1. Create scaling policy

2. Consumed capacity metrics are published to Amazon CloudWatch

3. CloudWatch uses Amazon SNS to send email

4. CloudWatch alarm invokes Application Auto Scaling

5. Application Auto Scaling issues `UpdateTable` request

6. DynamoDB processes request

SNS topic

Amazon CloudWatch

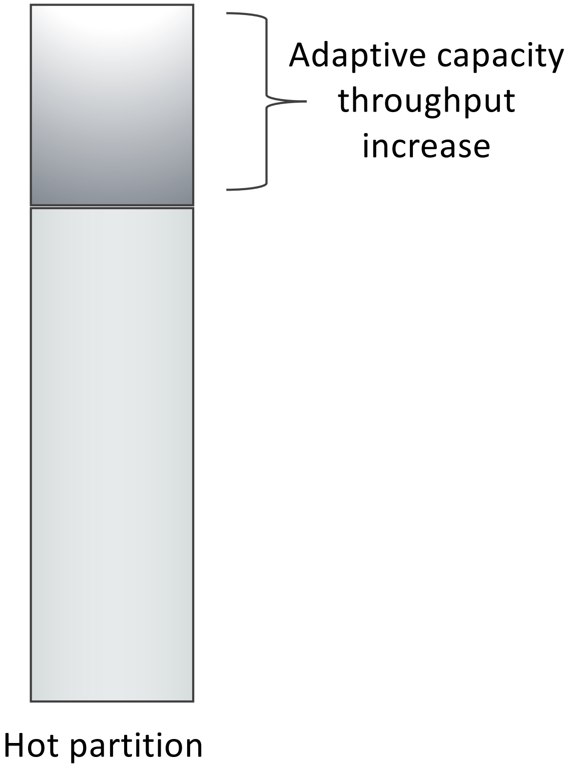
Application Auto Scaling

Amazon DynamoDB

Scaling throughput capacity: DynamoDB adaptive capacity

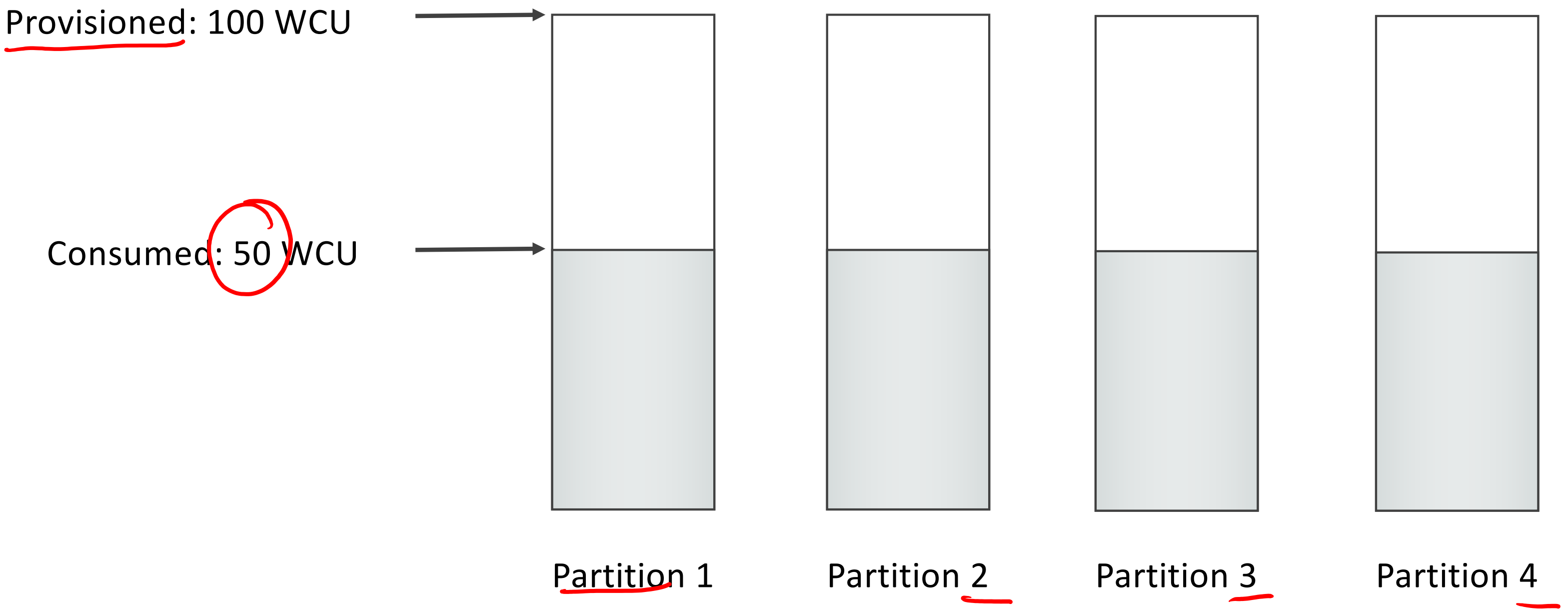
- Enables reading and writing to hot partitions without throttling
- Automatically increases throughput capacity for partitions that receive more traffic*
- Is enabled automatically for every DynamoDB table

*Traffic cannot exceed the table's total provisioned capacity or the partition's maximum capacity.



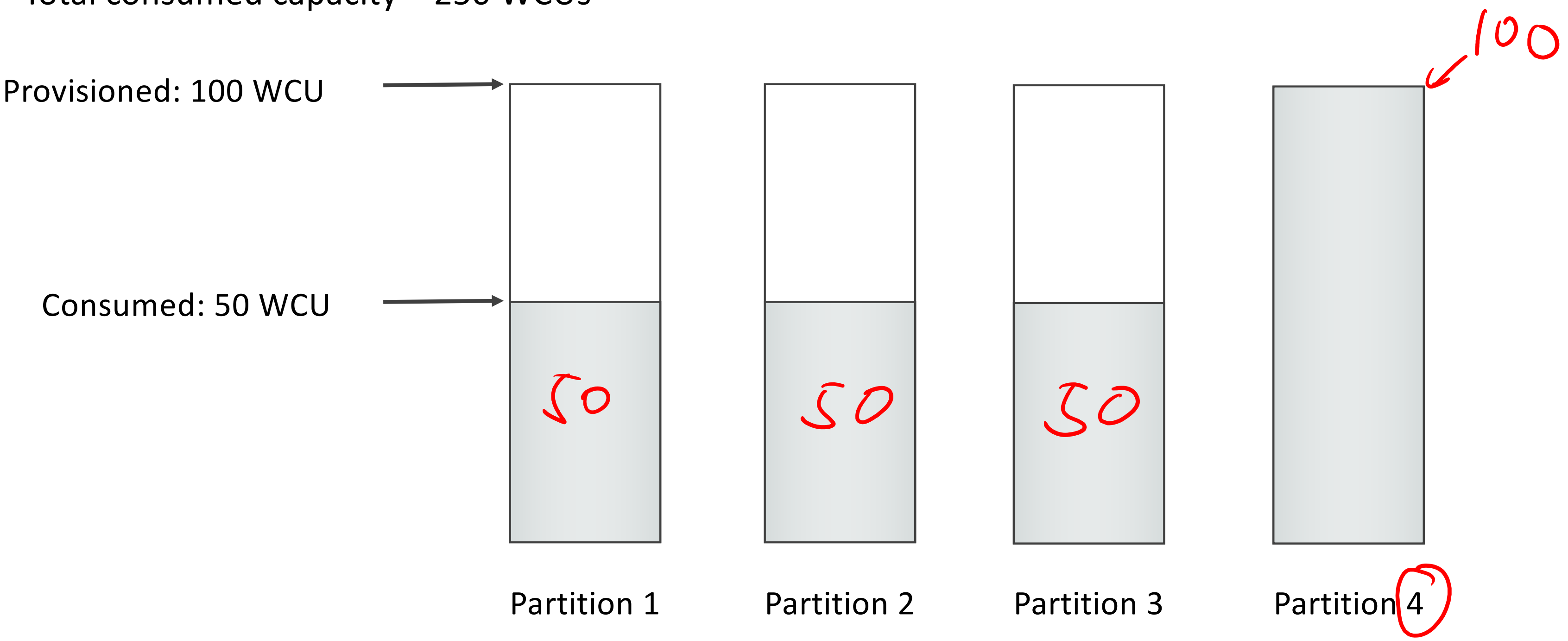
Adaptive capacity example (1 of 3)

Example table with adaptive capacity
Total provisioned capacity = 400 WCUs
Total consumed capacity = 200 WCUs



Adaptive capacity example (2 of 3)

Example table with adaptive capacity
Total provisioned capacity = 400 WCUs
Total consumed capacity = 250 WCUs

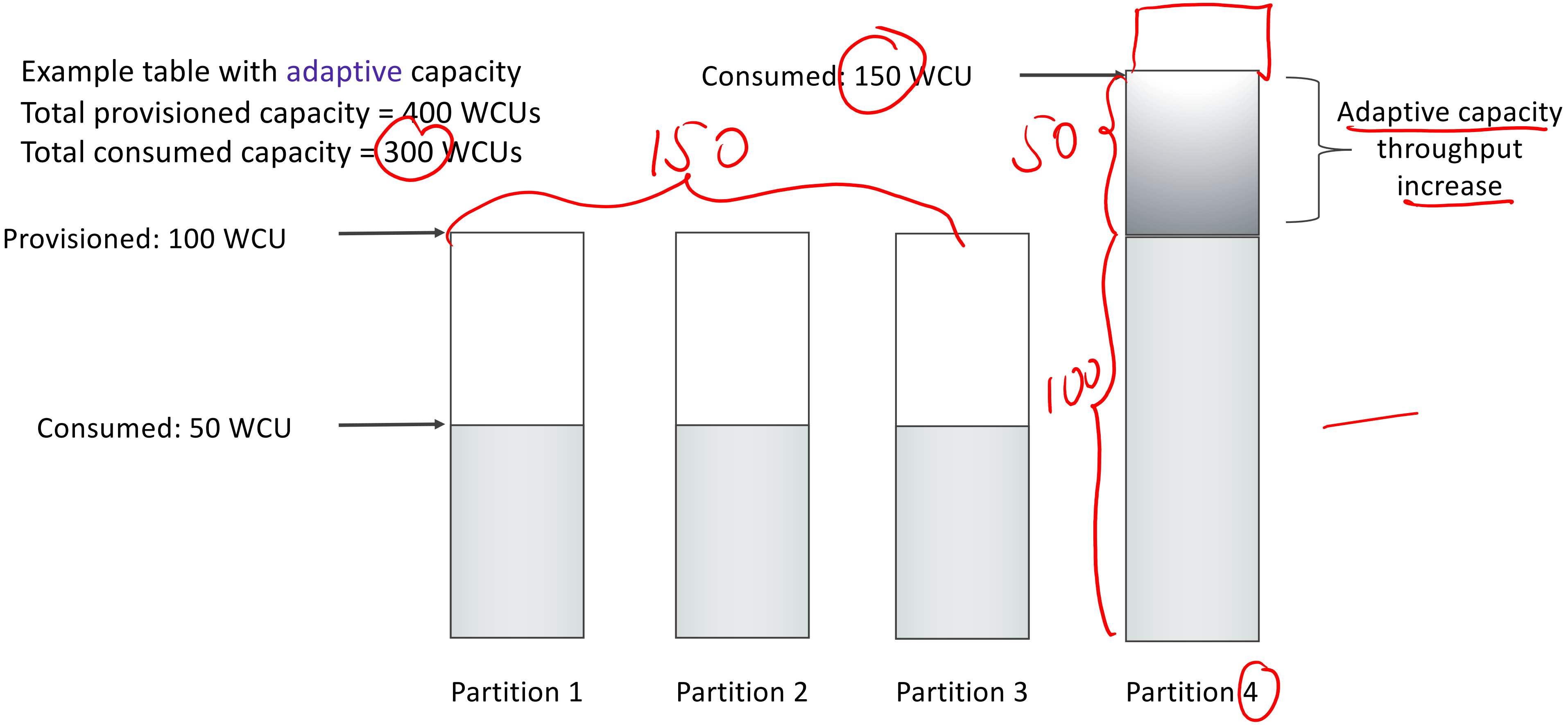


Adaptive capacity example (3 of 3)

Example table with adaptive capacity

Total provisioned capacity = 400 WCUs

Total consumed capacity = 300 WCUs



Provisioned: 100 WCU

Consumed: 50 WCU

Consumed: 150 WCU

Adaptive capacity throughput increase

Partition 1

Partition 2

Partition 3

Partition 4

Cloud Solutions Architecture Discussions

Highly Available Shopping Cart: A Stateful Web Application

A stateful application saves client **session data** (i.e., information about previous client requests, login/authentication status, or "state" data). For some stateful systems, this data is saved on the **server** where the application runs. In enterprise architectures, state data is saved within the **caching tier**, and not on the application server itself. A stateful app still has a **back-end database**, but it uses its saved state data as context when processing subsequent requests from the same client.

https://www.youtube.com/watch?v=u046sTH3jdI

Section 3 key takeaways

- You can use push-button scaling to vertically scale compute capacity for your RDS DB instance
- You can use read replicas or shards to horizontally scale your RDS DB instance
- With Amazon Aurora, you can choose the DB instance class size and number of Aurora replicas (up to 15)
- Aurora Serverless scales resources automatically based on the minimum and maximum capacity specifications
- Amazon DynamoDB On-Demand offers a pay-per-request pricing model
- DynamoDB auto scaling uses Amazon Application Auto Scaling to dynamically adjust provisioned throughput capacity
- DynamoDB adaptive capacity works by automatically increasing throughput capacity for partitions that receive more traffic

Module 9: Implementing Elasticity, High Availability, and Monitoring

Section 4: Designing an environment that’s highly available

aws academy

Highly available systems

- Can withstand some measure of degradation while remaining available
- Have minimized downtime
- Require minimal human intervention
- Recover from failure or roll over to secondary source in an acceptable amount of degraded performance time

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

Elastic Load Balancing

Elastic Load Balancing

A managed load balancing service that distributes incoming application traffic across multiple EC2 instances, containers, IP addresses, and Lambda functions.

- Can be external-facing or internal-facing
- Each load balancer receives a DNS name
- Recognizes and responds to unhealthy instances

Types of load balancers

Application Load Balancer

ALB

HTTP

HTTPS

- Flexible application management
- Advanced load balancing of HTTP and HTTPS traffic
- Operates at the request level (Layer 7)

Network Load Balancer

NLB

TCP, UDP, TLS

- Ultra-high performance and static IP address for your application
- Load balancing of TCP, UDP, and TLS traffic
- Operates at the connection level (Layer 4)

Classic Load Balancer

CLB

PREVIOUS GENERATION

for HTTP, HTTPS, TCP, and SSL

- Load balancing across multiple EC2 instances
- Operates at both the request level and connection level

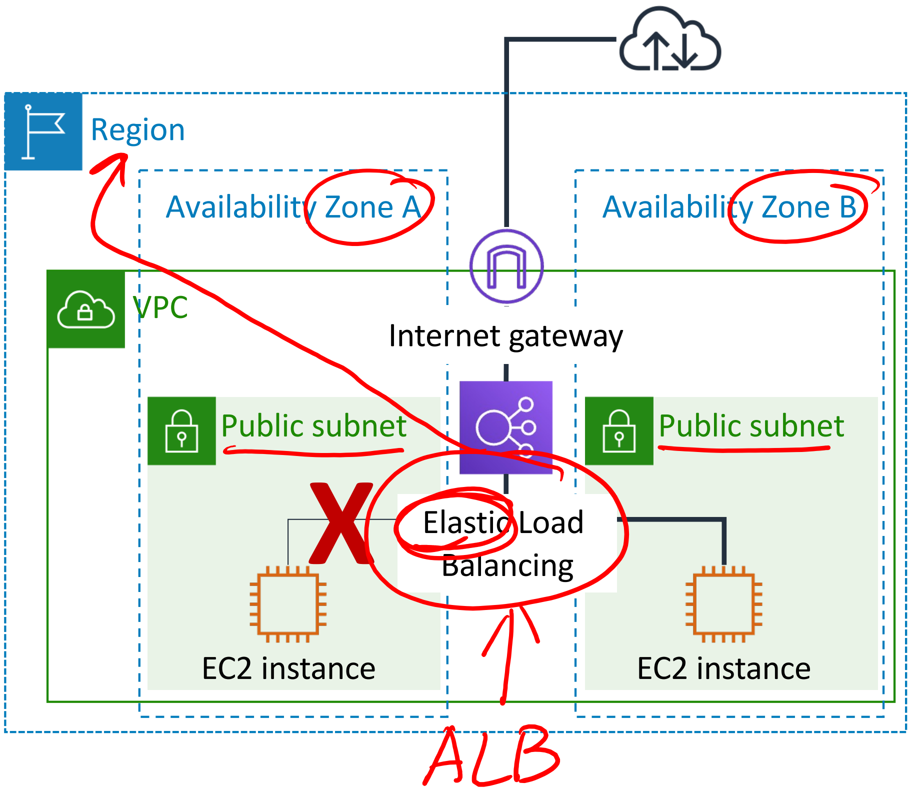
Implementing high availability

aws academy

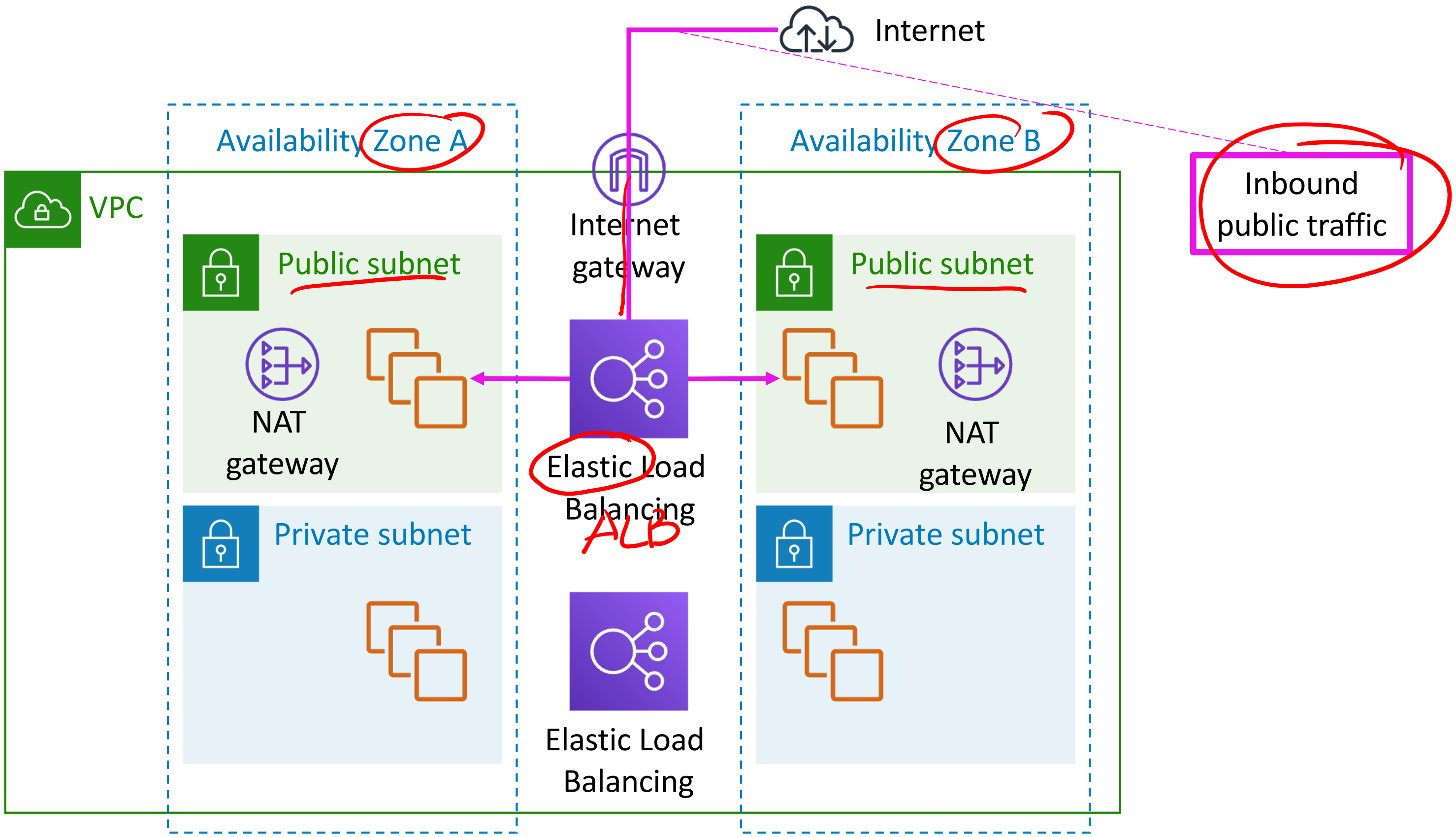
One region < AZ1, AZ2

Start with two Availability Zones per AWS Region.

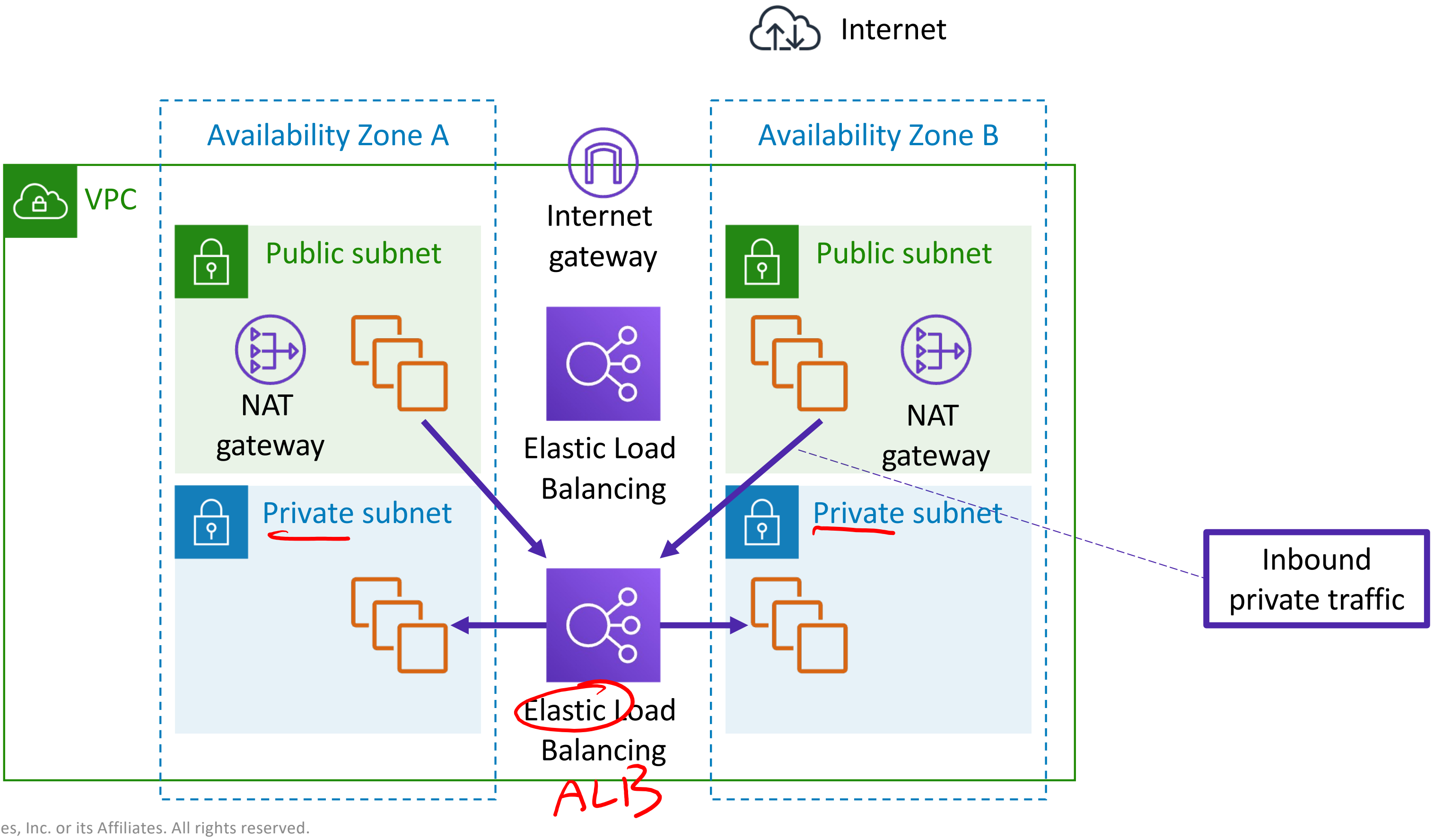
If resources in one Availability Zone are unreachable, your application shouldn't fail.



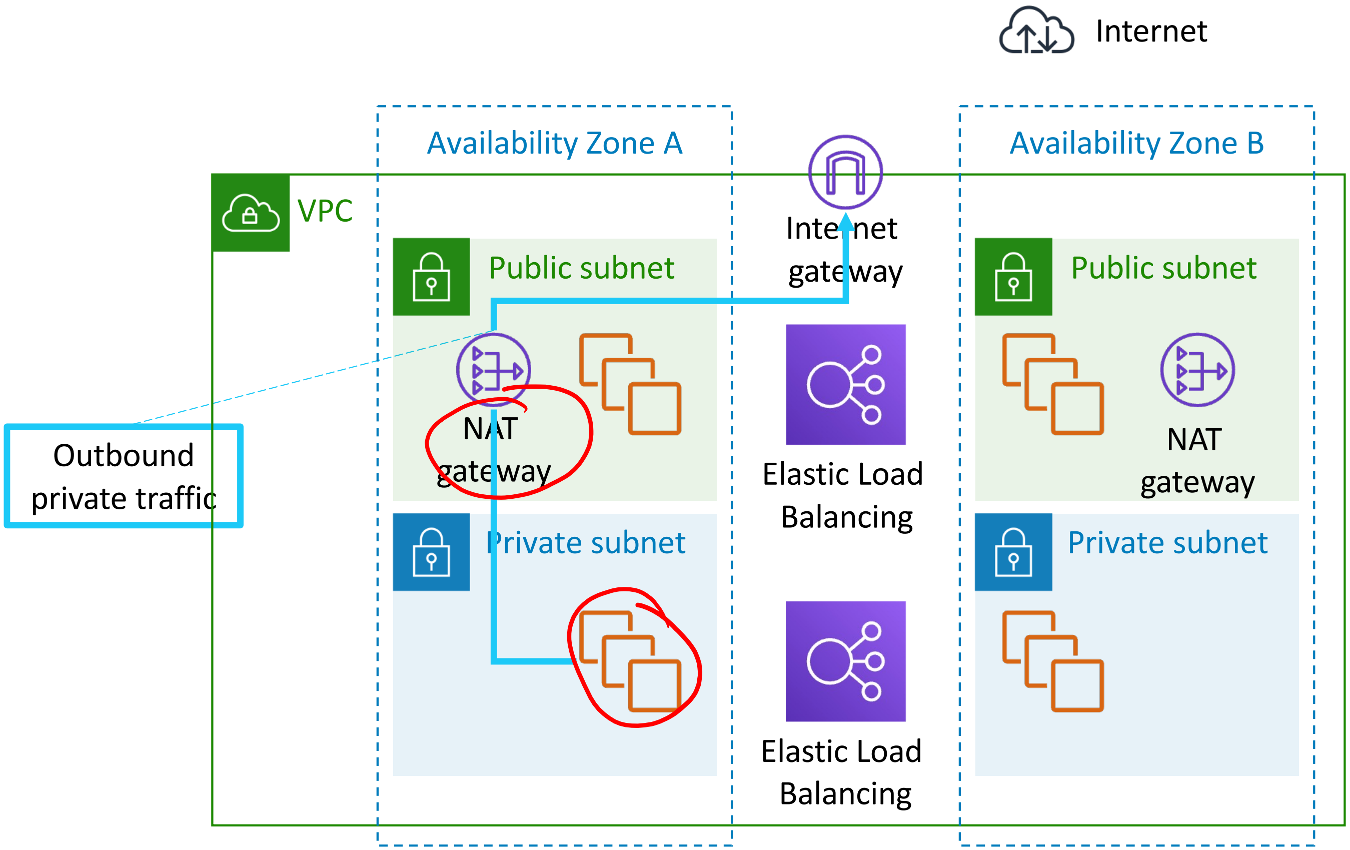
Example of a highly available architecture (1 of 3)



Example of a highly available architecture (2 of 3)



Example of a highly available architecture (3 of 3)



Demonstration: Creating a Highly Available Web Application

Amazon Route 53

Amazon Route 53

Amazon Route 53 is a highly available and scalable cloud DNS service.

- Translates domain names into IP addresses
- Connects user requests to infrastructure that runs inside and outside of AWS
- Can be configured to route traffic to healthy endpoints, or to monitor the health of your application and its endpoints
- Offers registration for domain names
- Has multiple routing options
- The only AWS service that provides 100% availability SLA

Amazon Route 53 - Records

- Route 53 records define how you want to route traffic for a domain
- Record contains:
  - Domain/subdomain name – e.g., dal.com
  - Record Type – e.g., A or AAAA
  - Value – e.g., 1.2.3.4
  - Routing policy – how Route 53 responds to queries
  - TTL – amount of time the record cached at DNS Resolvers
- Route 53 supports the following DNS record types:

  A / AAAA / CNAME / NS

Amazon Route 53 – Record Types

aws academy

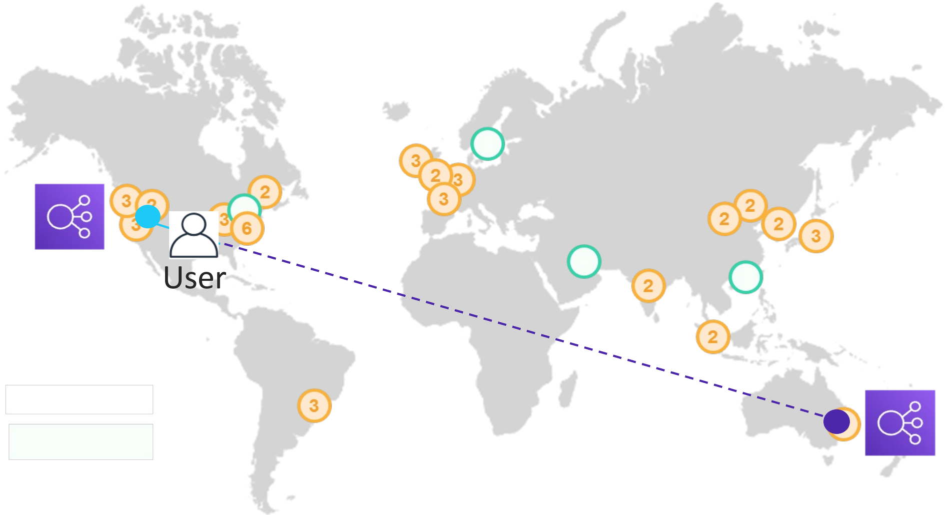
- A – maps a hostname to IPv4
- AAAA – maps a hostname to IPv6
- CNAME – maps a hostname to another hostname
    - The target is a domain name which must have an A or AAAA record
    - Can't create a CNAME record for the top node of a DNS namespace (Zone Apex)
    - Example: you can't create for example.com, but you can create for dev.example.com
- NS – Name Servers for the Hosted Zone DNS

Reference:
https://docs.aws.amazon.com/Route53/latest/DeveloperGuide/ResourceRecordTypes.html

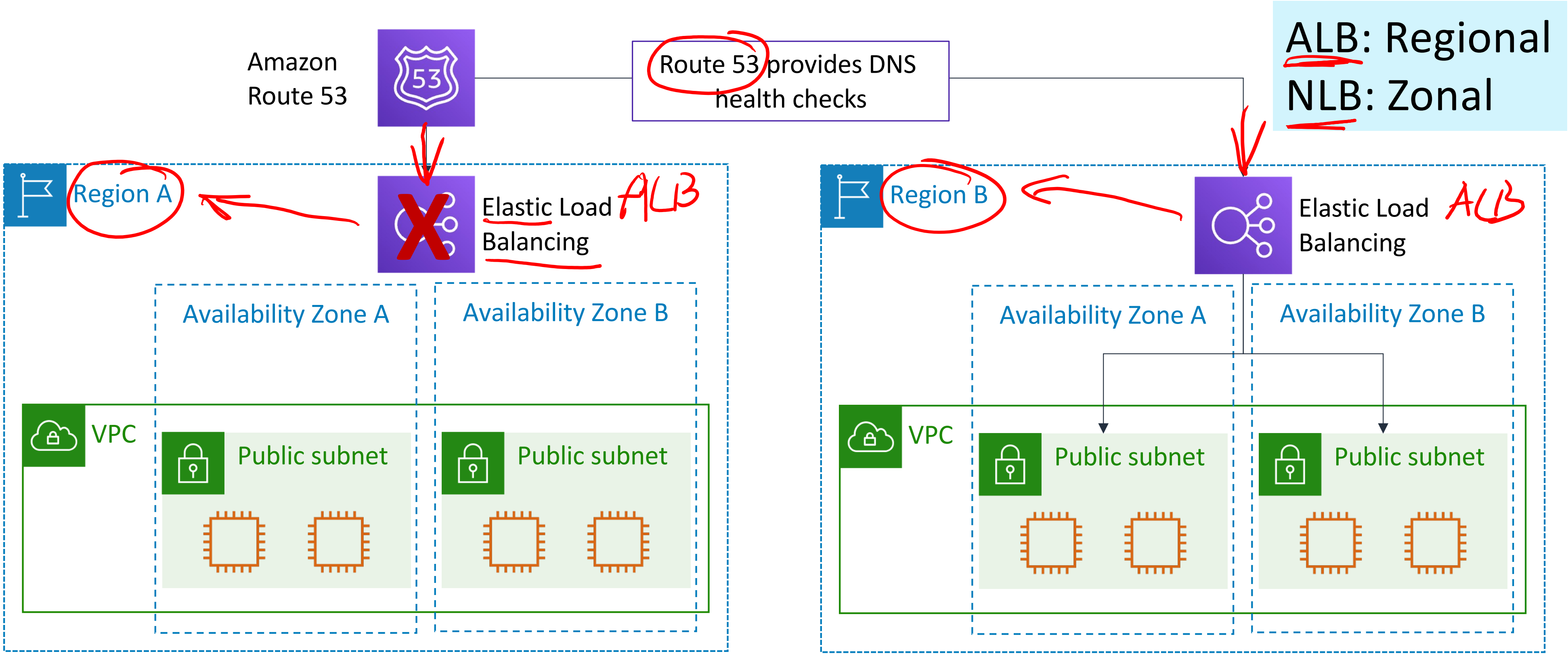
Amazon Route 53 supported routing

Unlike a LB, Route 53 doesn't route any traffic, but only responds to the DNS queries.

- Simple routing
- Weighted round robin routing
- Latency-based routing
- Geolocation routing
- Geoproximity routing
- Failover routing
- Multivalue answer routing



Multi-Region high availability and DNS



Demonstration: Amazon Route 53

More Amazon Route 53 videos:

Route 53 records (https://www.youtube.com/watch?v=10JKpg-eqZU&list=PLt1SIbA8guuusDOIqQuiFKerF_4_nQ_Xs&index=2)

Routing policies (https://www.youtube.com/watch?v=Tfkmd-bBXjQ)

Section 4 key takeaways

- You can design your network architectures to be highly available and avoid single points of failure
- Route 53 offers various routing options that can be combined with DNS failover to enable low-latency, fault-tolerant architectures