Design Fundamentals

EC2 Design

Security Group Design

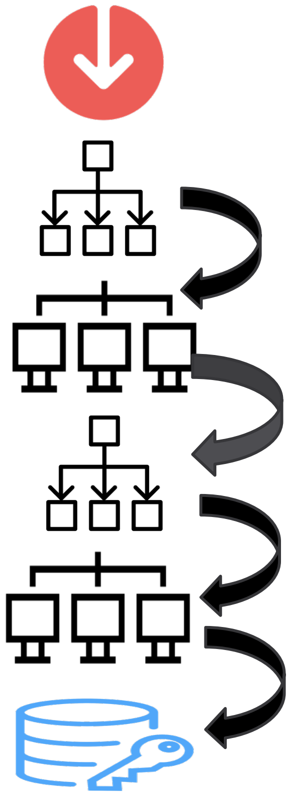
ALB Security Group — **Allow: Port: 443** — **Outbound: Web Tier SG**

Web Tier Security Group — **Allow: Port: 80** — **Outbound: ALB SG**

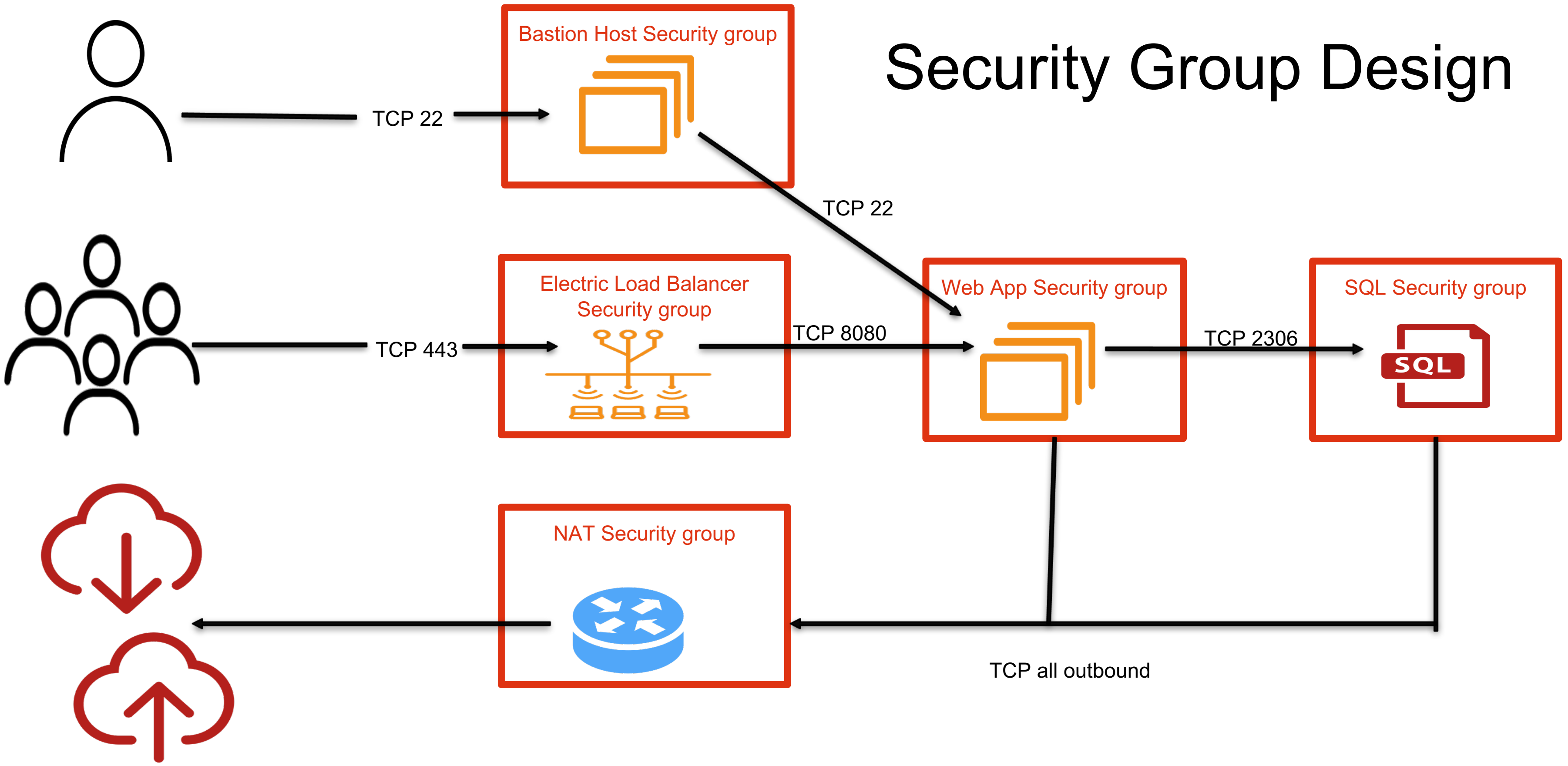
ALB Security Group — **Allow: Port: 80** — **Outbound: App Tier SG**

App Tier Security Group — **Allow: Port: 80** — **Outbound: DB Tier SG**

DB Tier Security Group — **Allow: Port: 3030**



Security Group Design



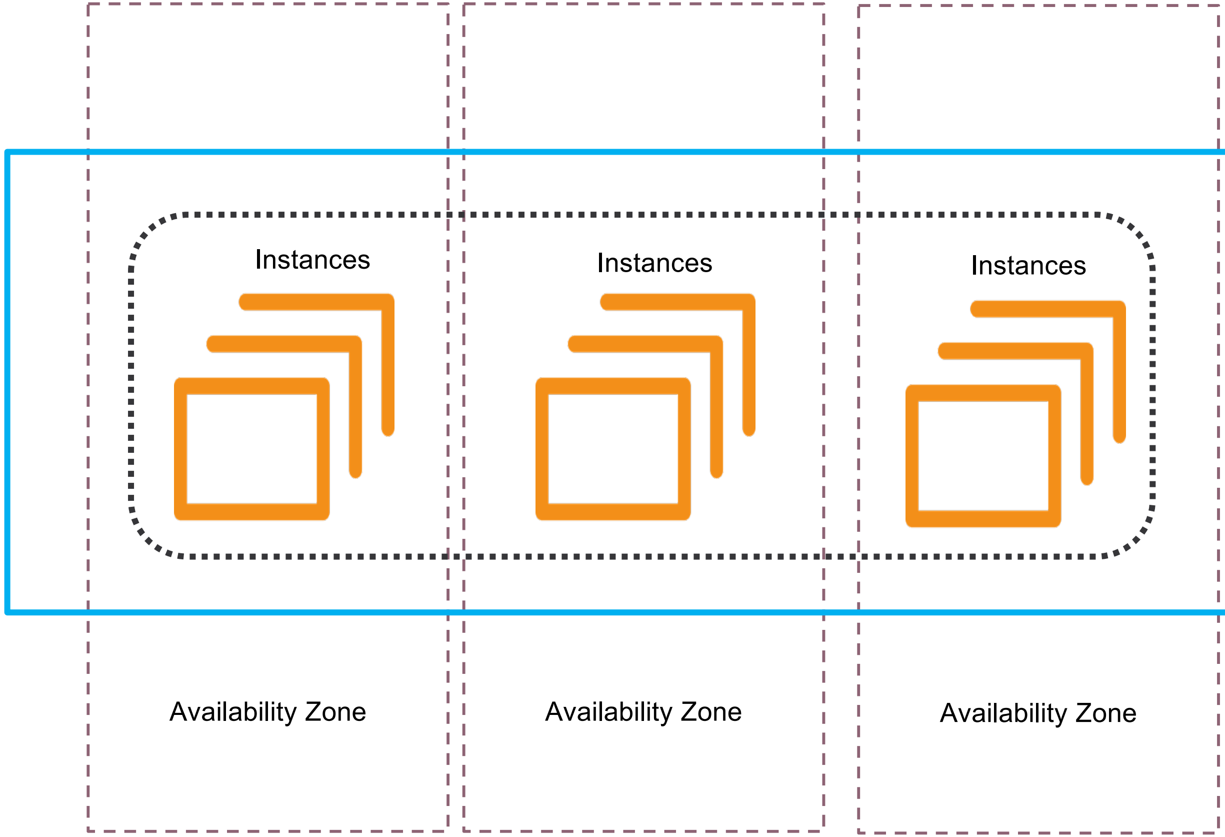
Availability based on Requirements

- A properly designed architecture meets requirements but is not over-engineered
- The appropriate level of availability is defined by the acceptable RTO and RPO
- RTO – How long will the system be unavailable when failure occurs?
- RPO – How much data loss is acceptable?
- Always look for single points of failure in any architecture design
- Think about each failure domain and the impact of each failure
- A single point of failure may be acceptable in a test environment

Designing for Computer Failure

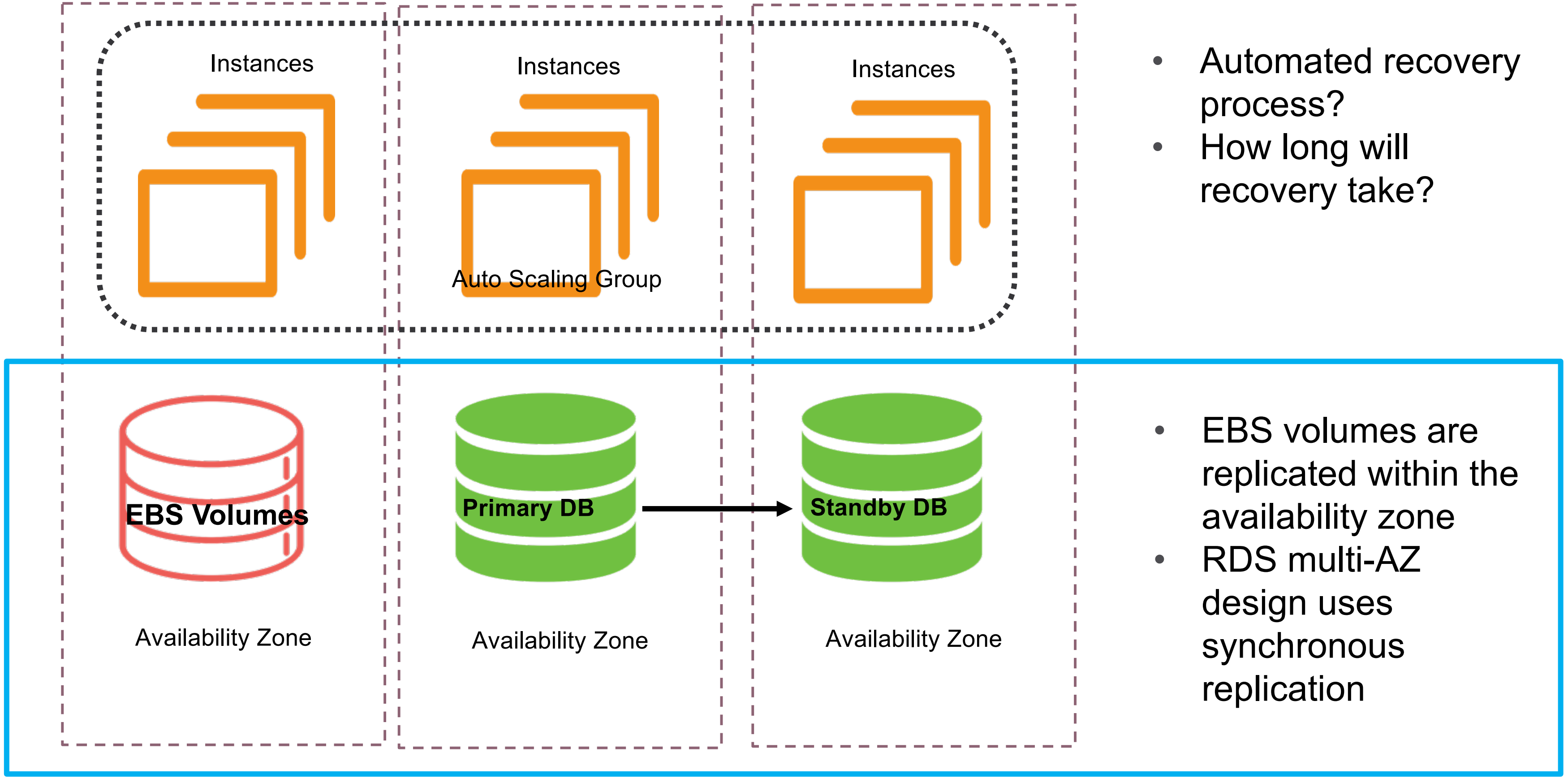
- A single EC2 instance has no high-availability
- Auto Scaling groups, and Auto Recovery help with instance availability
- Auto scaling groups with minimum / maximum set to 1 ensure new instances are created when the original instance fails
- EC2 Auto Recovery recovers supported EC2 instances due to hardware or network problems restarting the instance on a different host
  - The same instance ID, IP addresses, and EBS volumes are retained
- Route 53 health checks can check the health of ELB, and route traffic to another regional resources when failures occur

Architecting for Compute failure



- Auto Scaling groups
- Elastic Load Balancing
- EC2 Auto Recovery
- Route 53 Health Checks

Architect Availability based on Requirements



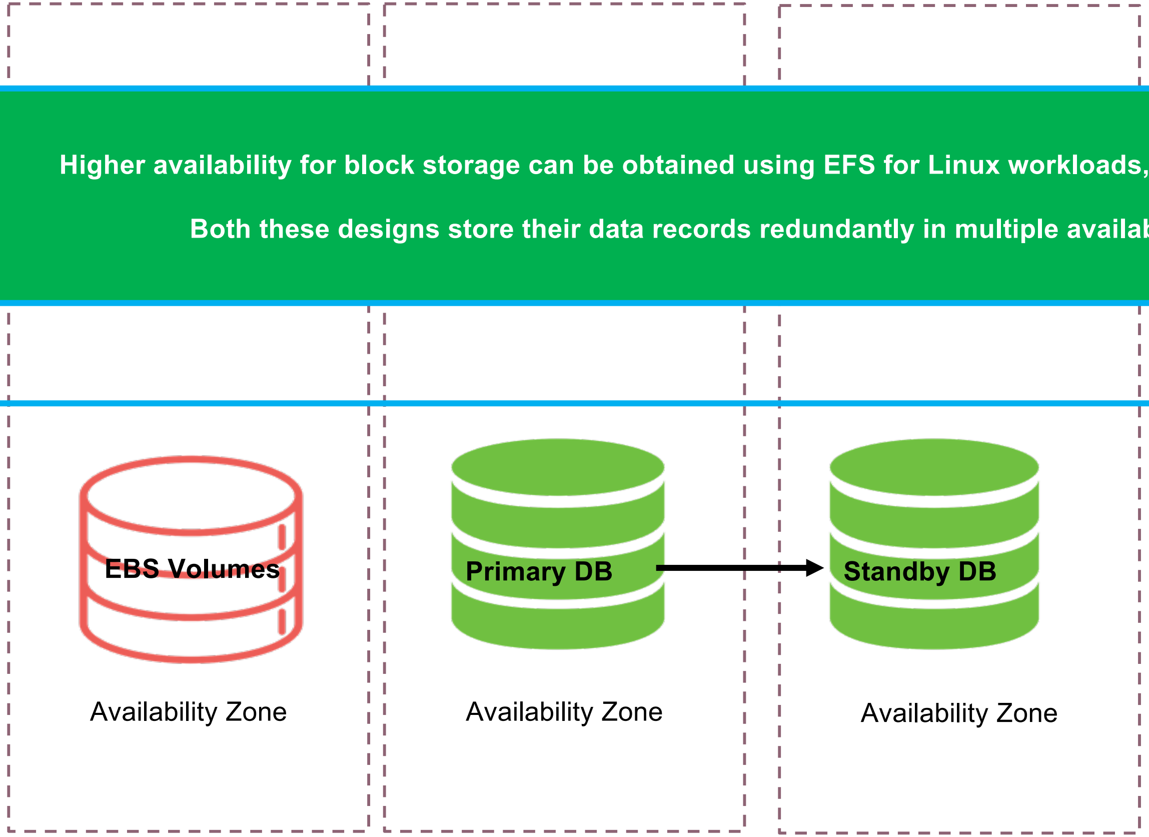
- Automated recovery process?
- How long will recovery take?

- EBS volumes are replicated within the availability zone
- RDS multi-AZ design uses synchronous replication

When Data components Fail

Higher availability for block storage can be obtained using EFS for Linux workloads, or FSx for Windows.

Both these designs store their data records redundantly in multiple availability zones



- RDS multi-AZ
- Aurora: cluster shared storage
- EBS snapshots
- ElastiCache – Read replicas

AWS EBS Redundancy

EBS Volumes

- Replicated within the availability zone to multiple servers
- High-durability volume (io2) (99.999)
- EBS Multi-Attach with io1 volumes to 16 Nitro-based EC2 instances within the same availability zone
- EBS Snapshots with Data Lifecycle Manager
- Elastic Volumes allow dynamic volume sizing
- Copy EBS snapshots across Regions

S3 Redundancy

S3 Object Storage

- 99.999999999% durability of objects over a given year
- Redundantly store objects on multiple devices across a minimum of three Availability Zones in an Amazon S3 Region before returning SUCCESS
- Replicate S3 buckets across a single Region (SRR) or across Regions (CRR)
- Use S3 Replication Time Control
- Enable versioning to preserve, retrieve, and restore every version of every object
- Create tags to help control access
- S3 Object Lock and WORM protection

RDS Redundancy

RDS

- Automatic backups for point in time recovery of DB instances up to 35 days
- Periodic data backups along with transaction logs allow restoration of a database instance to any second during the defined retention period
- Copy automated snapshots for custom restore
- Manual DB snapshots
- Create read replicas for serving read traffic or disaster recovery

Load Balancing

Load-Balancing FYI

- Classic load balancer – operates at Layer 4
  - TCP or HTTP / HTTPS
  - Supports SSL offload
- Network load balancer operates at Layer 4
  - TCP
  - NLB routes connections to targets (EC2 and containers)
- Application load balancer – operates at Layer 7
  - Routes traffic to instances and targets based on the content of the request
  - HTTP / HTTPS
  - Supports SSL offload
  - Supports WAF (Web Application Firewall)

Each load balancer ordered is assigned a traffic profile with a prescribed amount of throughput capacity

Internet or Internal Facing

Internet Facing ELB

- ELB nodes have Elastic IPs
- Traffic routed to private IP addresses (eth0) of EC2 instance
- Require a public subnet in each AZ where the ELB is required
- Requires an IGW

Internal Only ELB

- ELB nodes have private IPs
- Traffic routed to private IP addresses (eth0) of EC2 instance
- Does not require an IGW

DNS name format: <name>-<id-number>.<region>.elb.amazonaws.com

ELB Design

- Health Checks
  - Unhealthy instances stop receiving traffic; network and application health checks (URL)
- Security
  - Managed by Security Groups (ALB,CLB) in hosted VPC
- Failover - if no healthy targets in AZ failover to another AZ
- Connection draining – deregistration /registration of EC2 instances
- Cross-zone load balancing supported
- Integrates with Auto Scaling and CloudWatch

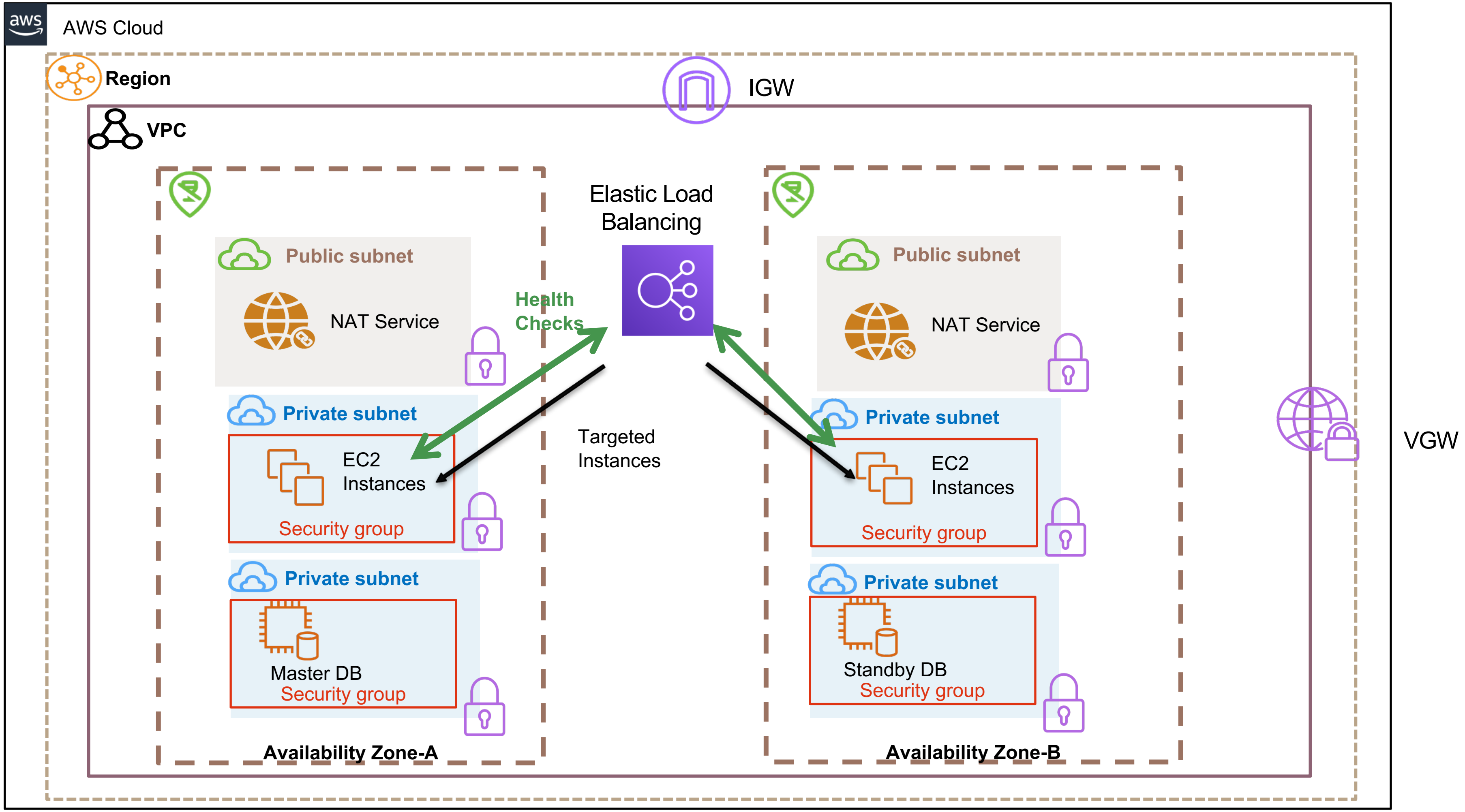
Health Checks

- Configured for each target group against the registered targets
- Health checks ensure your resource is available and ready to accept user requests
- A registered target is typically defined as *healthy* or *unhealthy*
- A newly added target to the target group is defined as *initial*; once its health check is successful, the target is defined as *healthy*
- When the target is being removed and connection draining is underway, the target is marked as *draining*

Connection Draining

Connection draining keeps existing connections open until the client closes them

Prevents new requests from being sent to the instances that are tagged as unhealthy

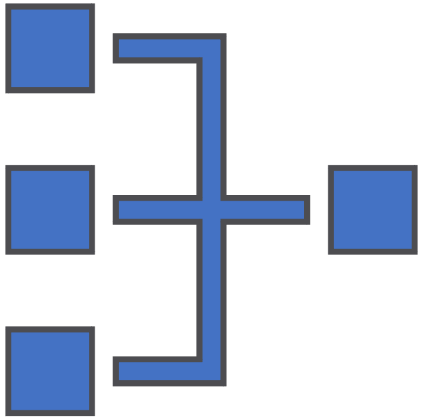

AWS Cloud
Region
IGW
VPC
Elastic Load Balancing
Health Checks
Targeted Instances
Public subnet
NAT Service
Private subnet
EC2 Instances
Security group
Private subnet
Master DB
Security group
Availability Zone-A
Public subnet
NAT Service
Private subnet
EC2 Instances
Security group
Private subnet
Standby DB
Security group
Availability Zone-B
VGW

Application Load Balancer Features

- Load balancing at layer 7
- HTTPS support
- Performs SSL offloading
- Authentication (Cognito)
- Sticky session support
- Access logging available
- Web application firewall support (WAF)
- Delete protection

ALB Routing

- Content-based, path-based, and host-based routing are supported
- Connection requests can be directed to individual services based on the type of the request
- Routing to multiple domains hosted in multiple target groups is supported based on the host field or URL path of the HTTP header



Content Based Routing

- ALB can route requests based on the content of the request in the host field: host-based or path-based
- Host-based is domain name-based routing (app.com)
- Host field contains the domain name and optionally the port number
- You can also create rules that combine host-based and path-based routing
- Path-based is URL based routing app.com/image

Sticky Session Support

- The load balancer can bind the user’s active session to a specific instance, ensuring that all users’ requests are sent to the initial instance
- After a request is routed to a target, a cookie is generated by the load balancer and returned to the client
- When the back-end server that the user is connected to fails, the load balancer automatically chooses a new healthy instance and moves the user to the new instance for the remainder of the session
- Consider using a central storage location for user session information, using a DynamoDB table or a hosted ElastiCache cluster

Network Load Balancer

- A Network Load Balancer (NLB) is designed for supporting private resources using the TCP protocol
- The NLB doesn’t support SSL offload, HTTPS, or WAF
- NLB supports end-to-end encryption using TLS

Classic Load Balancer

- Operates at layer 4 and 7; HTTP / HTTPS, SSL and TCP
- Does not support Elastic IPs – resolved with DNS name
- Supports SSL termination
- Cookie based sticky sessions are supported by load balancer or applications
- Supports health checks / Connection draining
- Enable or disable cross-zone load balancing
- Increases on application load require pre-warming

Autoscaling

AWS Services that use Auto Scale

- EC2 - Scale up or down based on demand
- EBS - Automatically add or remove instances from target groups
- DynamoDB - Increase or decrease provision to read and write capacity
- RDS storage - Scalable storage options
- Aurora - Dynamically adjust the number of Aurora replicas provisioned for a DB cluster
- Elastic Beanstalk - application and infrastructure

EC2 Auto Scaling Benefits

- Availability – EC2 Auto Scaling helps you ensure that you have the correct number of EC2 instances available to handle the load for your application
- Fault Tolerance – auto scaling detects unhealthy instances, terminates and relaunches an instance to replace the unhealthy one
- Management of costs – dynamically increase and decrease capacity as required; only pay for the computer that you need

Auto Scale Concepts

- Create collections of EC2 instances, called Auto Scaling groups
- Provide automatic horizontal scaling (scale-out) for your instances across availability zones
- Trigger to either launch or terminate instances
- Auto Scaling can span multiple AZs within the same AWS region
- Auto Scaling integrates with ELB and CloudWatch

Scaling Options

- Maintain – keep a minimum number of instances running
- When will auto scaling scale? (out or in)
  - Launch instances when application is under increased load
  - Terminate instances when application is not under load
  - CloudWatch metrics: (CPU usage, Incoming network traffic)
- Scheduled: Scale out / in at specific times
- Dynamic scaling: scale using CloudWatch metrics
- Target tracking or Simple scaling
  - Add / remove fixed number of instances +2
  - Add / remove percentage of existing capacity + 20%
- Step Scaling Policies – multiple policies within the same policy

Scaling Policy Options

Scaling Policy	**Details**
Target tracking scaling	Increase or decrease the capacity of the ASG based on a target value for specific cloud watch metric
Step scaling	Increase or decrease the capacity based on a set of scaling percentages
Simple scaling	Increase or decrease based on a single scaling adjustment

Target Tracking Scaling Policy

- Select the scaling metric and target value
- Cloud watch alarms trigger scaling policy
- The scaling policy adds or remove compute capacity to keep the metric target close to the desired target
- Example: Keep the average aggregate CPU utilization of your Auto Scaling group at 60 percent
- Example: Keep the request count per target of your Application Load Balancer target group at 2000 for your Auto Scaling groups target value

Step Scaling policy

- Define a lower boundary for the metric value selected
- Define an upper boundary for the metric value selected
- Define the amount to scale based on the scaling adjustment type
- Change capacity by a specific capacity value

Simple Scaling Policy

- Dynamic scaling using a single metric
- Define high and low thresholds for the alarms
- Define how many instances to add or remove
- After a simple scaling activity is started, the policy must wait for the scaling activity or health check replacement to complete
- The cooldown period needs to expire before Auto scale can responding to additional alarms

Cooldown Period

- A scaling cooldown helps auto scaling groups from launching or terminating additional instances before the effects of previous scaling actions are launched
- After an Auto Scaling group launches an instance, it pauses for a default or custom period of time

Termination Policy

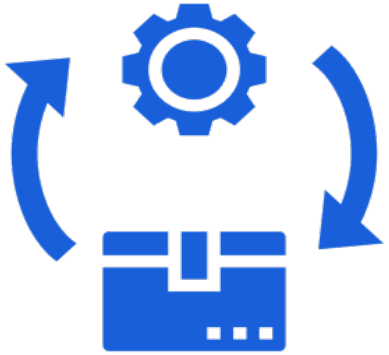
- Termination policies control which instances are terminated first when a scale-in event occurs.
- Auto scaling first determines which AZ has the most EC2 instances
  - Any instances that are protected from scaling in
  - Allocation - which of the two types (Spot or On-Demand) should be terminated.
  - Which instances are closest to the next billing hour.
  - Use the oldest launch template unless there are instances that use a launch configuration

Replacing Auto Scaling Instances

- **Operating lifetime parameters for EC2 instances can be defined**
- Instances can be automatically replaced that have been in service for the maximum amount of time defined
- The minimum value that can be set is seven days
- Instances are replaced one at a time with a pause in between replacements
- **Instant refreshes can be used to replace instances with updated configurations**
- A set of instances are taken out of service and terminated; a set of instances with the new configuration are then launched

Lifecycle Hooks

- Perform a custom action just before when EC2 instances launch or terminate when under the control of Auto Scale
- **At a scale out event:**
- One or more instances are launched
- Instances move from the Pending state to the Pending: Wait state
- Next you perform your custom action (the lifecycle hook action)
- **At termination:**
- Instances that scale in move from the InService state to the Terminating: Wait state
- Next you perform your custom action (the lifecycle hook action)



Auto scale based on an Amazon Simple Queue Service (SQS) queue.

Compute

Measure the acceptable number of messages in the queue per EC2 instance in the Auto Scaling group

Design Project

Scenario: Auto Scaling with CPU Utilization CloudWatch Alarms

Auto Scaling Group:

- Minimum = 4
- Maximum = 16

Auto Scaling Policy:

- CPU Utilization > 60 %
- Add 4 web servers = Double the policy

Multi-tier Design

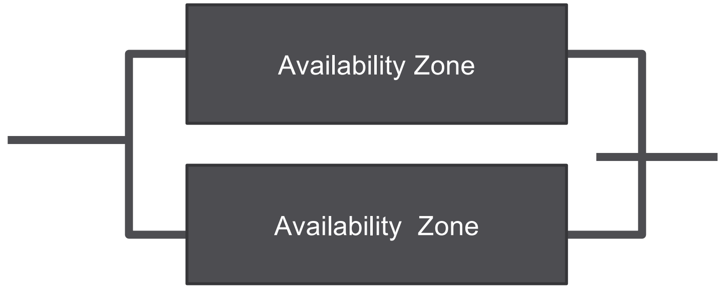
Hard Dependencies

- An interruption in the dependent system translates to an interruption of the application
- Desired workload availability: 99.9 %
- Independent System 1 – 99.95 %
- Independent System 2 – 99.0 %

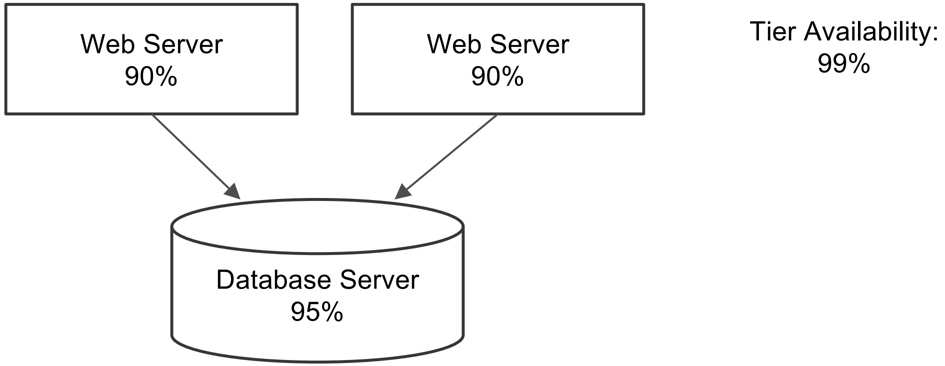
98.81 %

Redundant Components

- Use independent redundant components such as Availability Zones
- Availability equals 100 % minus the product of the independent component failure rates (AZ failure rate = 99.9 %)
- The resulting system availability is 99.9999 %

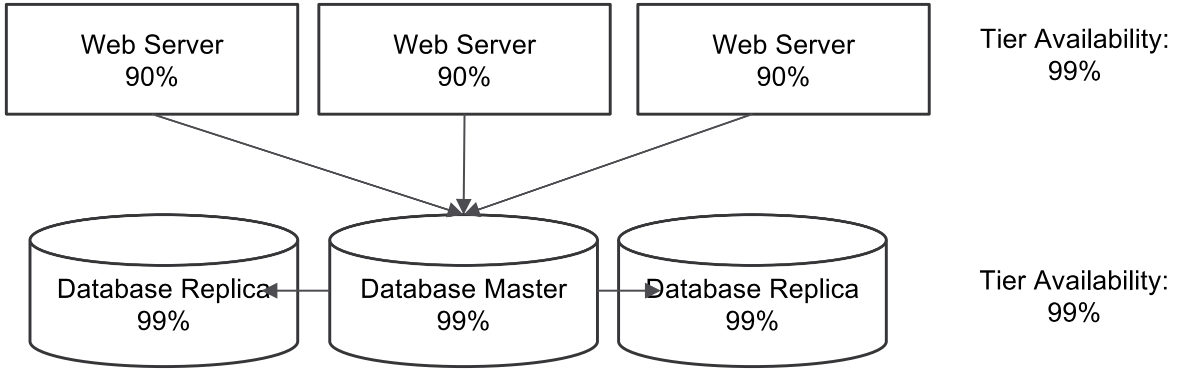


SLA Required: 99.5%



Total Availability: 0.95 *0.99 = 94.%

SLA Required: 99.5%



Total Availability: 0.999 *0.999 =
99.8.%

RPO and RTO at the Application Tier

- Auto Scaling provides an acceptable RTO for the web or application tier compute layer
  - Store the application state outside the web or application tier
  - Use SQS, DynamoDB, ElastiCache, or Kinesis
- When using a manual recovery, standard tasks should be automated with User Data, and EC2 Roles

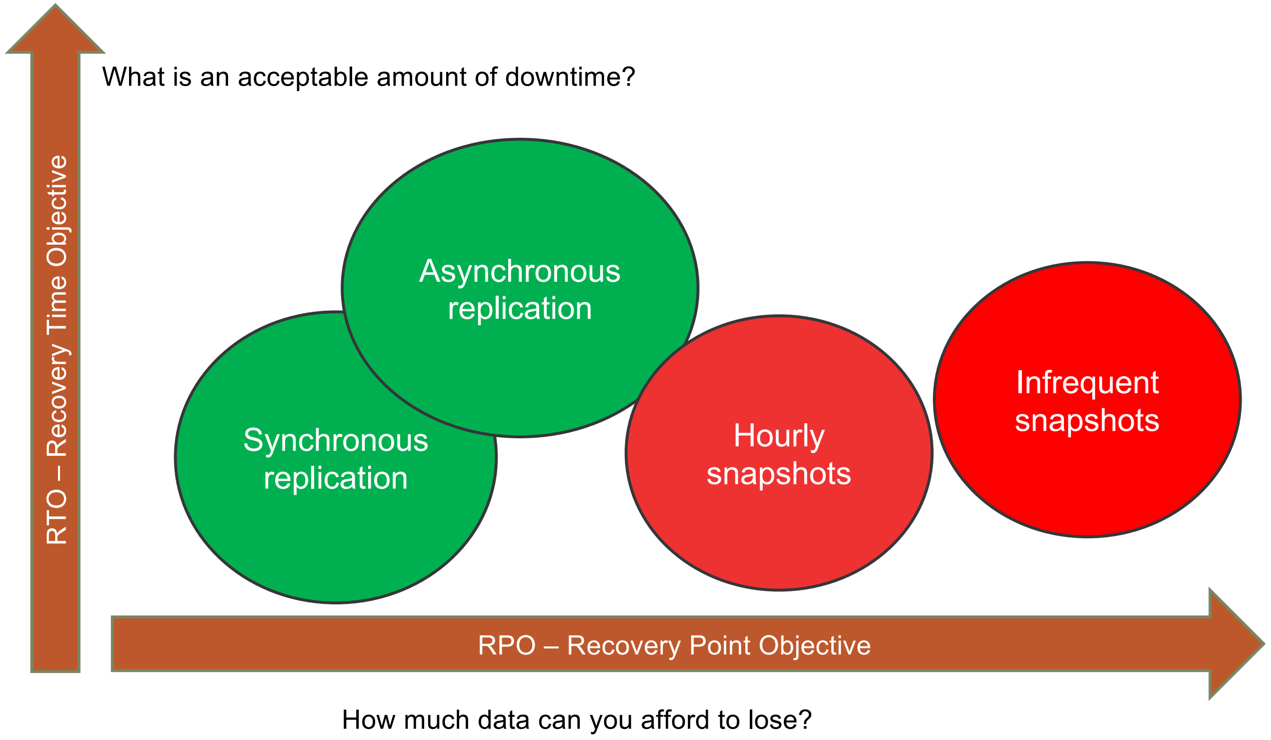
Disaster Recovery: RPO and RTO

- Data storage layer – for data records with little change create snapshots on a schedule
  - New EBS volumes can be created from a snapshot

- Asynchronous Replication
  - Asynchronous replication has a slower recovery point objective than synchronous replication
  - Asynchronous recovery requires switching to an additional copy of data
  - Asynchronous replication can be used within a region, across regions, and in hybrid designs
  - Primary and secondary data copies are both online

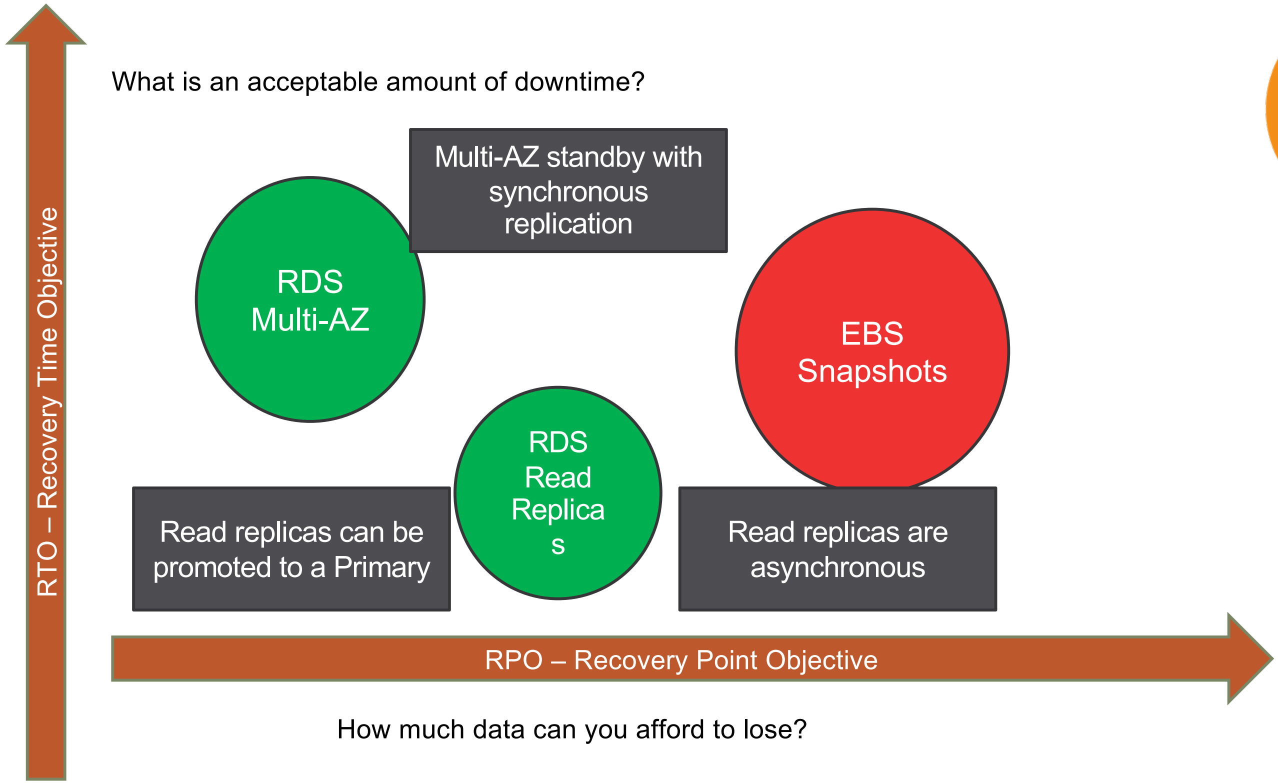
- Synchronous replication is a lower RPO than asynchronous
  - Synchronous replication is used within a region across availability zones
  - Across regions, synchronous replication would suffer due to latency

Disaster Recovery based on RPO and RTO



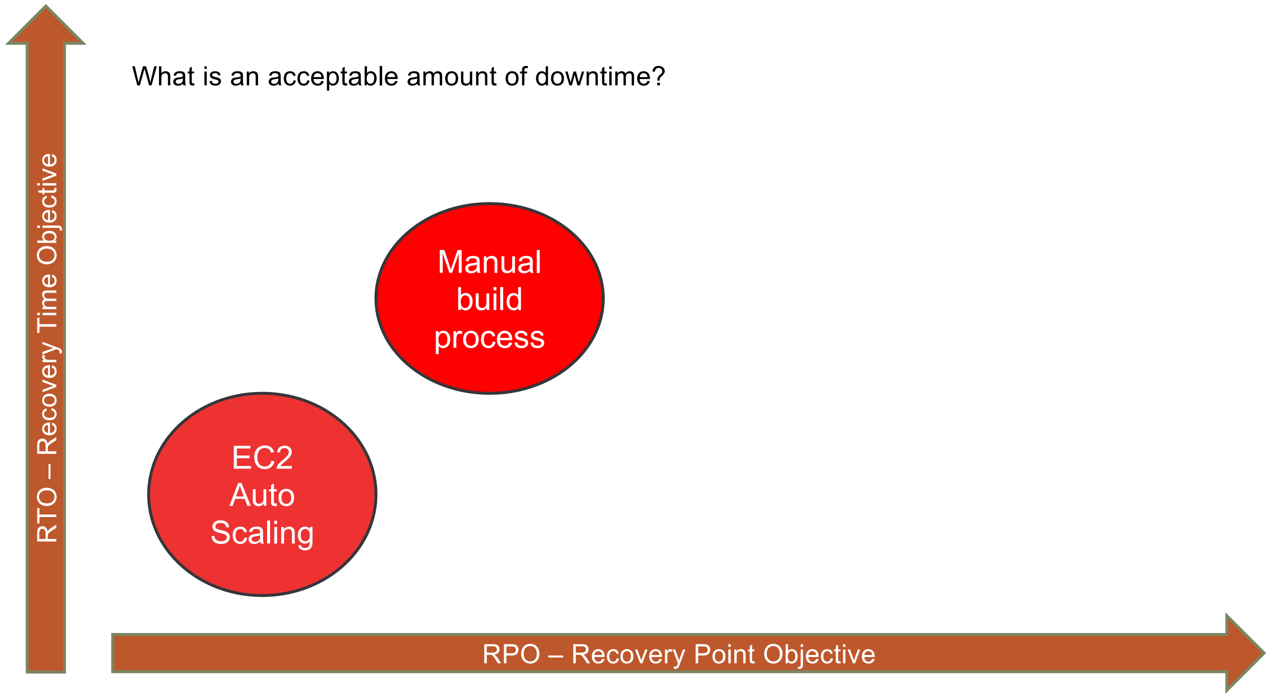
What is an acceptable amount of downtime?
RTO – Recovery Time Objective
Asynchronous replication
Synchronous replication
Hourly snapshots
Infrequent snapshots
RPO – Recovery Point Objective
How much data can you afford to lose?

Disaster Recovery based on RPO and RTO



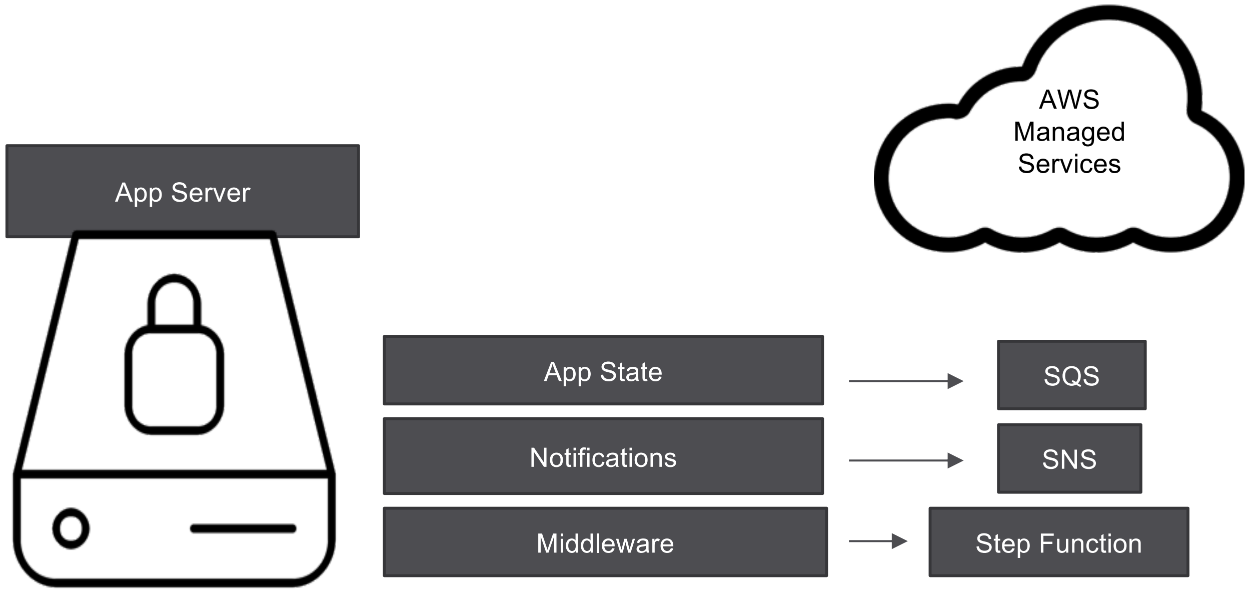
What is an acceptable amount of downtime?
Multi-AZ standby with synchronous replication
RDS Multi-AZ
EBS Snapshots
RDS Read Replicas
Read replicas can be promoted to a Primary
Read replicas are asynchronous
RTO – Recovery Time Objective
RPO – Recovery Point Objective
How much data can you afford to lose?

Disaster Recovery based on RPO and RTO


What is an acceptable amount of downtime?
RTO – Recovery Time Objective
Manual build process
EC2 Auto Scaling
RPO – Recovery Point Objective

How much data can you afford to lose?

Stateless Design


App Server
AWS Managed Services
App State
Notifications
Middleware
SQS
SNS
Step Function

Asynchronous Integration

- Loose coupling between services
- Useful for designs that do not need immediate response
  - One component generates the event
  - Another component processes the event
- No direct point-to-point interaction between application components

Stateless Apps

- Stateless components have no knowledge of any activity
- Compute resources are independent components for processing
- Stateless defines a “loose coupling”, or “asynchronous integration

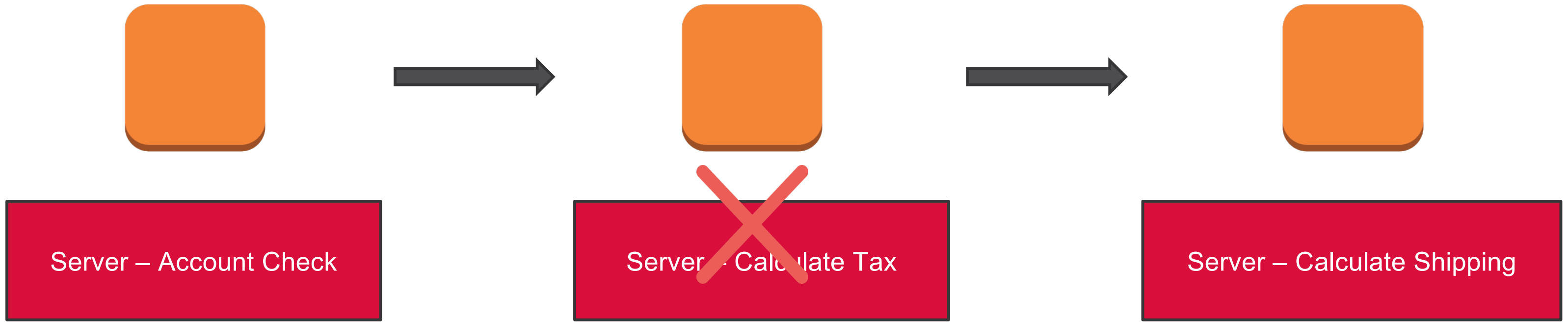
SQS

- Simple Queue Service
- Polling modes: Push, Pull
- Text-only
- Message lifecycle
- Asynchronous work queues

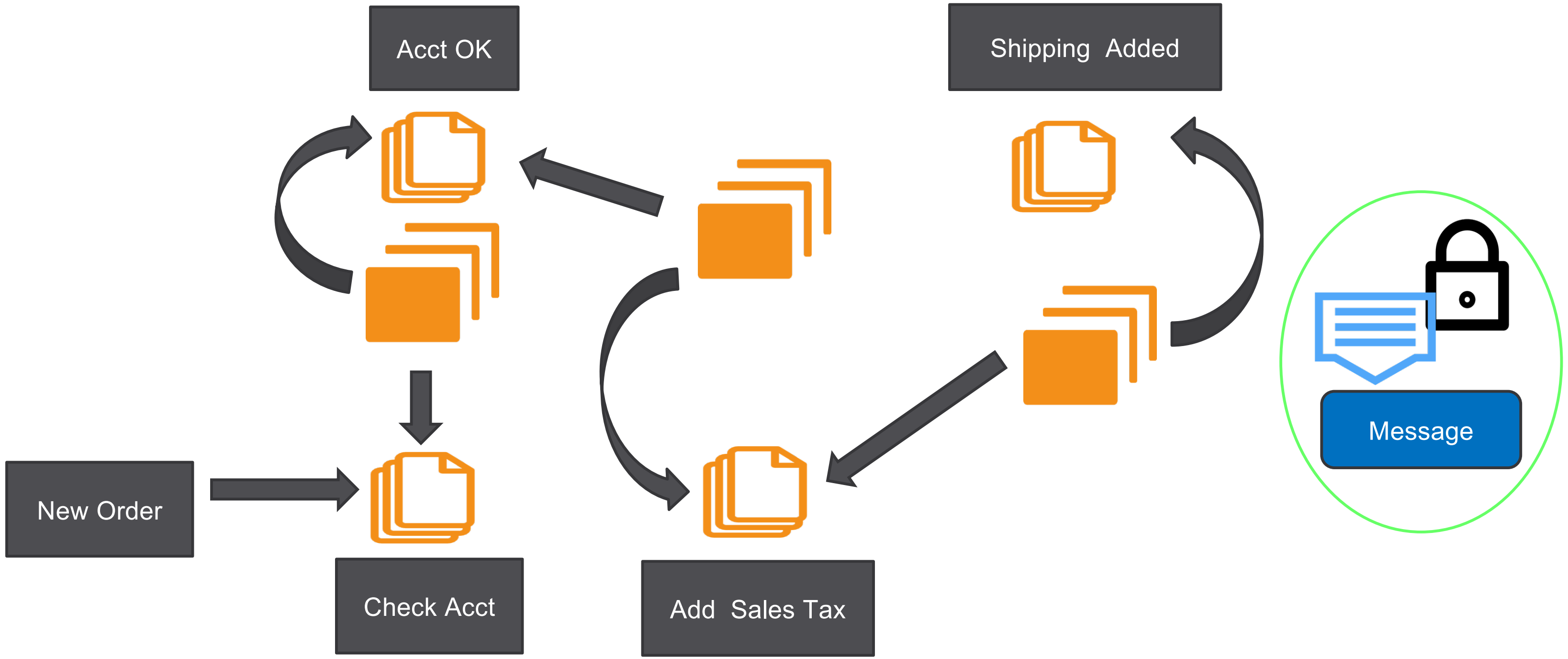
SNS

- Push model
- Subscribe to topic
- Delivery mechanisms: HTTP / HTTPS, SQS, SMS, Email
- Lambda

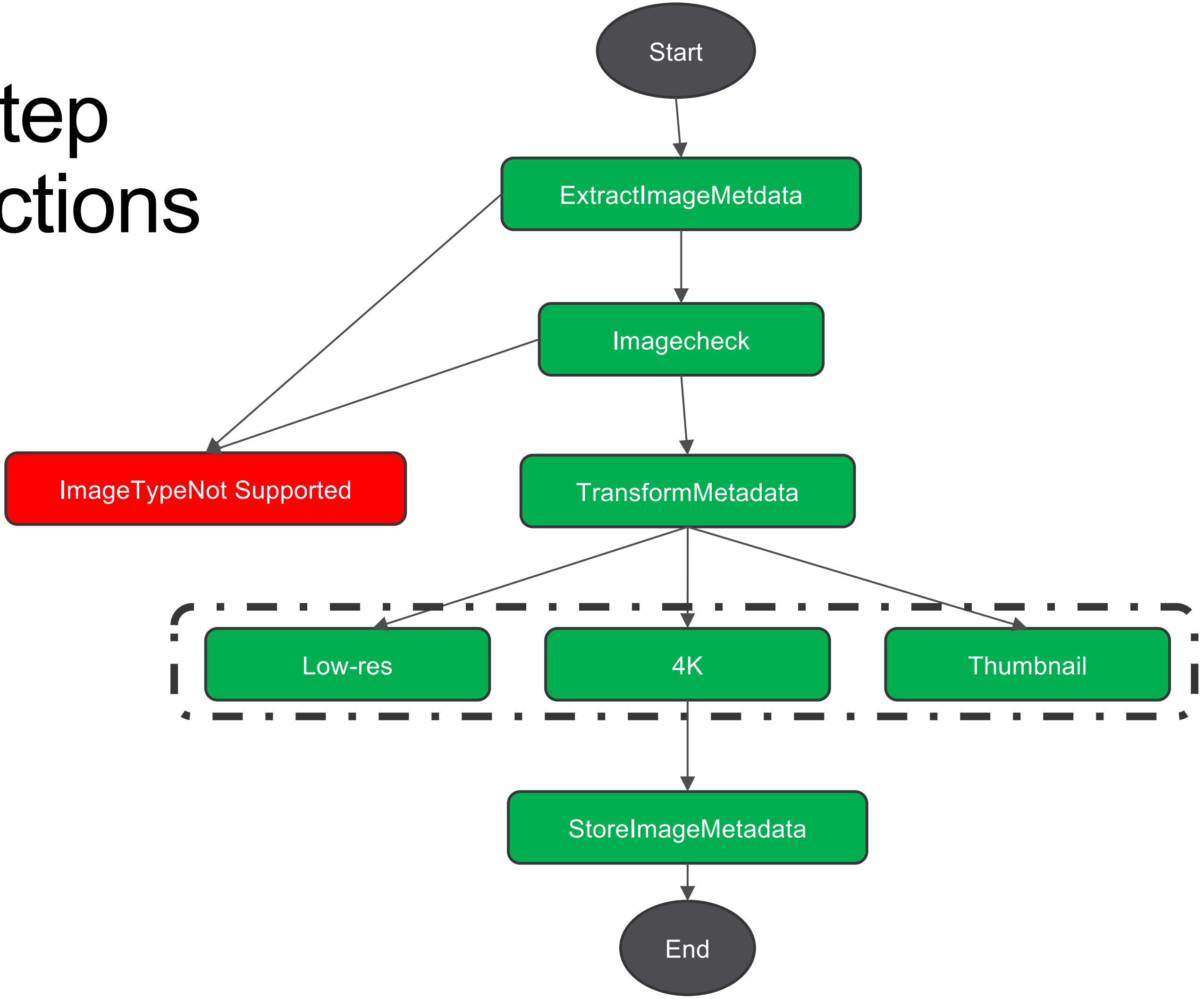
Tight Coupling



Loose Coupling



Step Functions



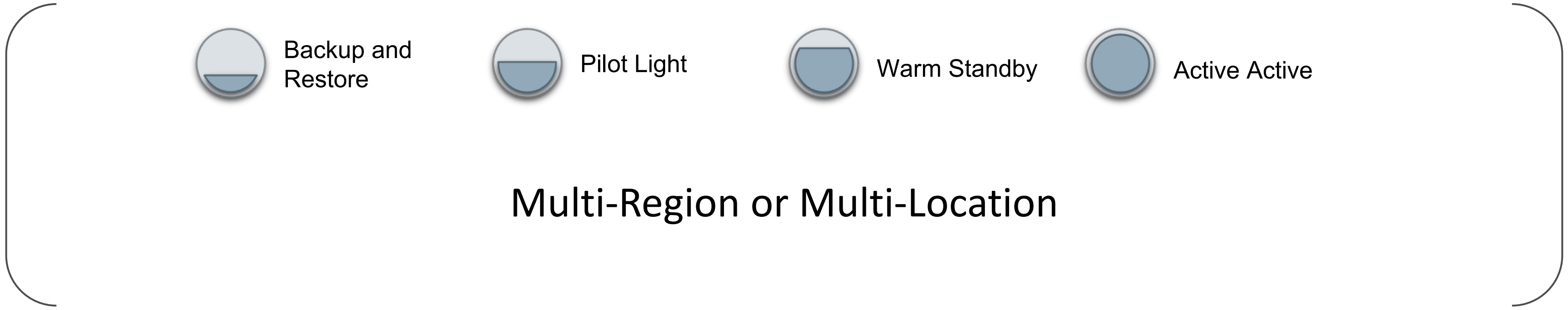
Multi-Region

Multi-region-active-active

=

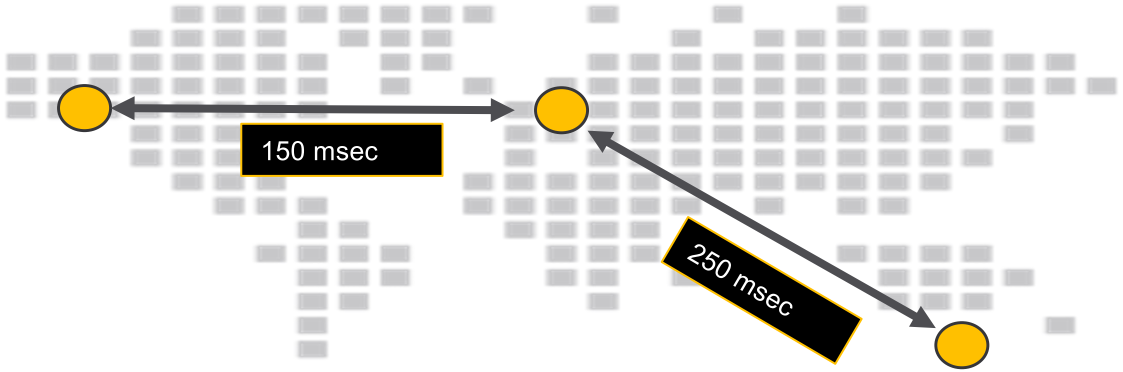
Two Active full stack AWS Regions

Business Continuity Strategies



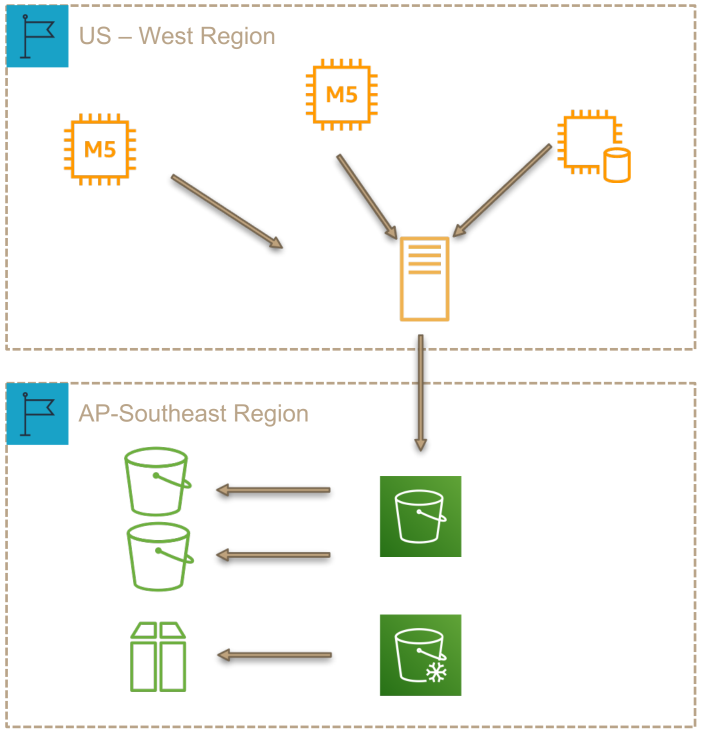
Why Multi-Region?

- Business continuity / Disaster Recovery
- Customer base geographically distributed
  - Improved latency for application users
  - Meet regulatory compliance for data records



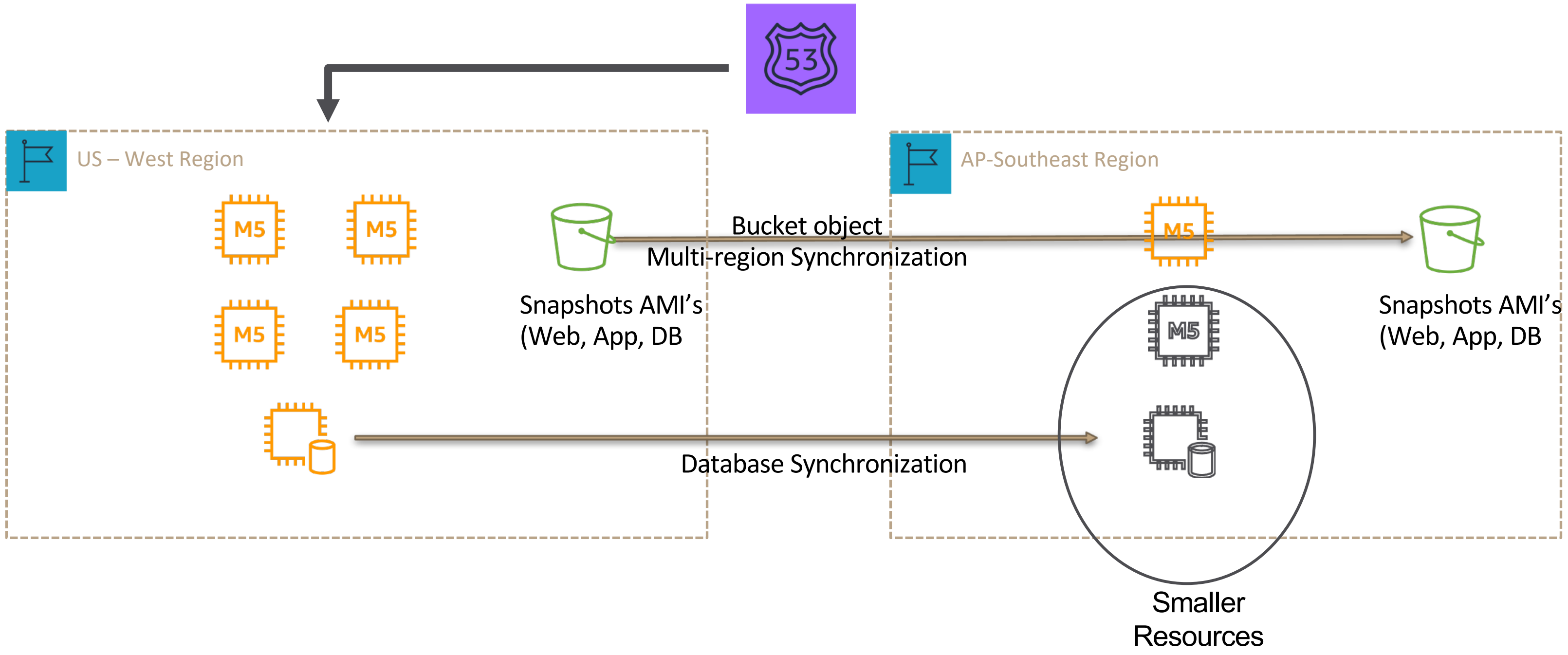
Multi-Region Backup and Restore

- Backup to another AWS Region
- RDS managed database
- S3 or S3 Glacier
- High durability storage in multiple physical locations



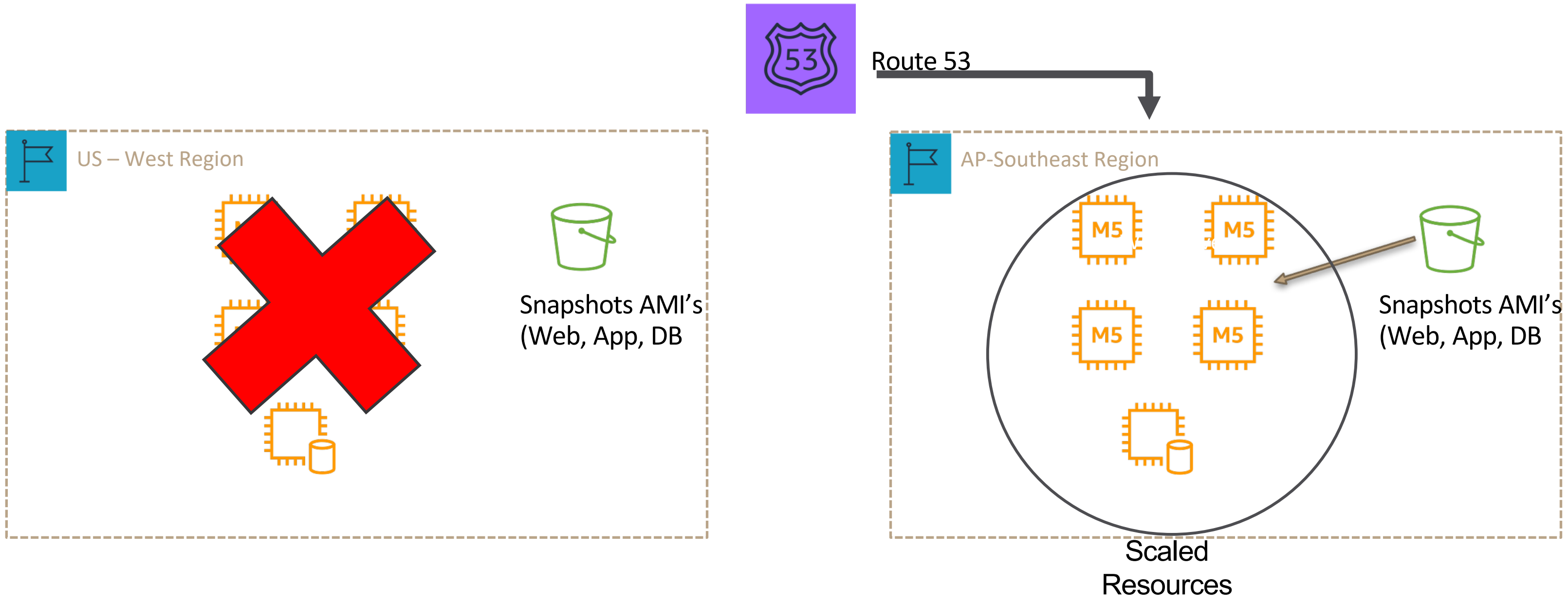
Multi-Region Pilot Light

Scale redundant site during failure

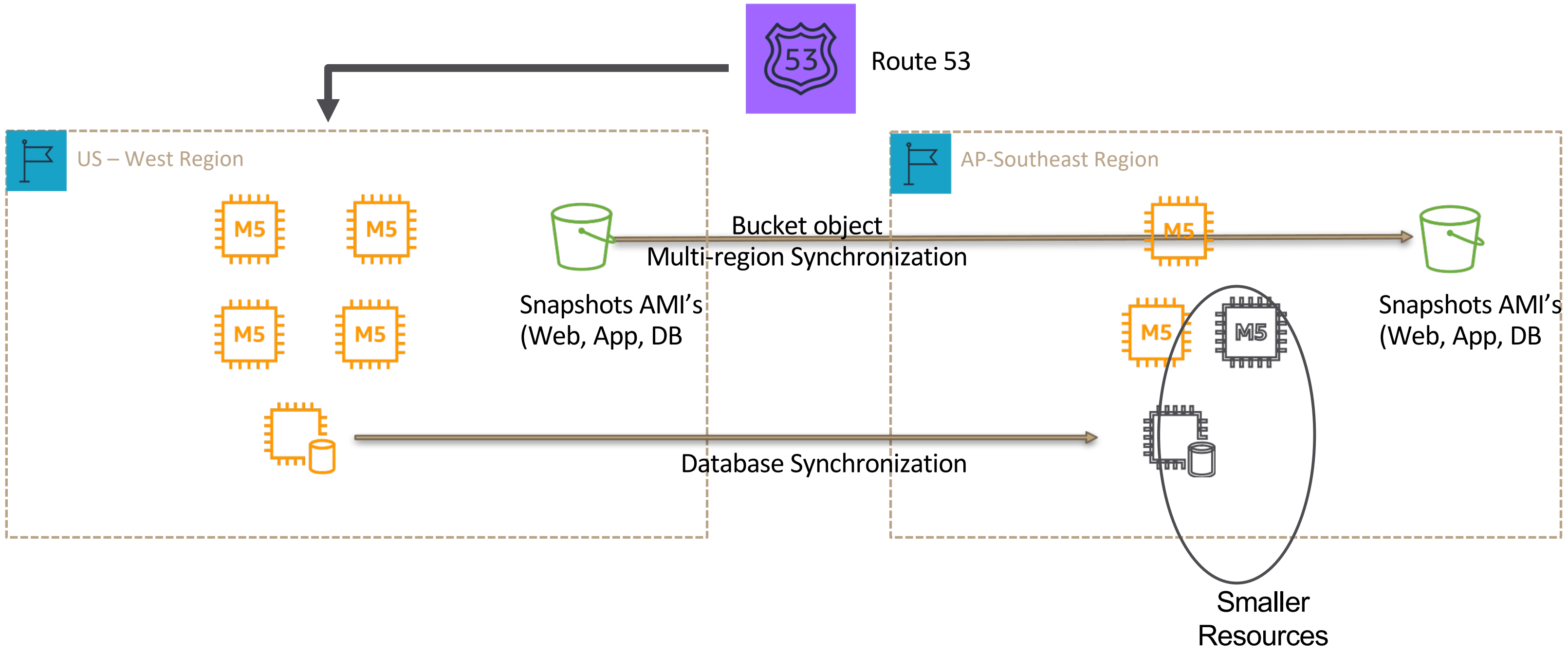


Multi-Region Pilot Light

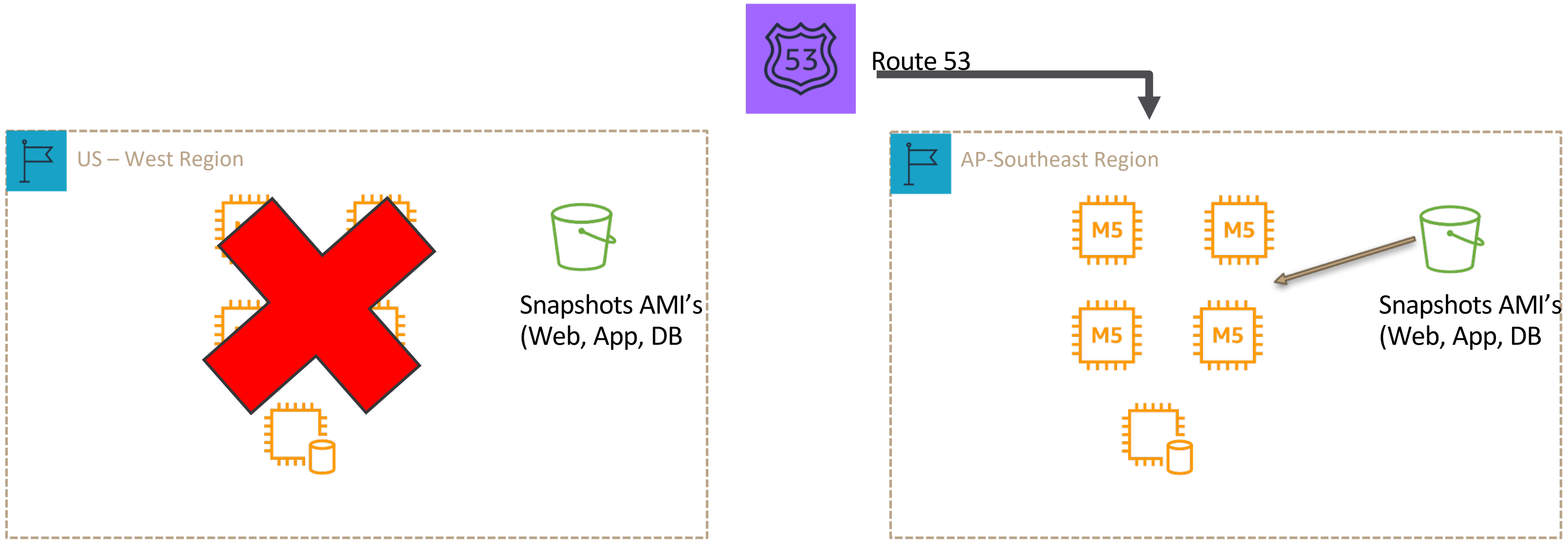
Scale redundant site during failure



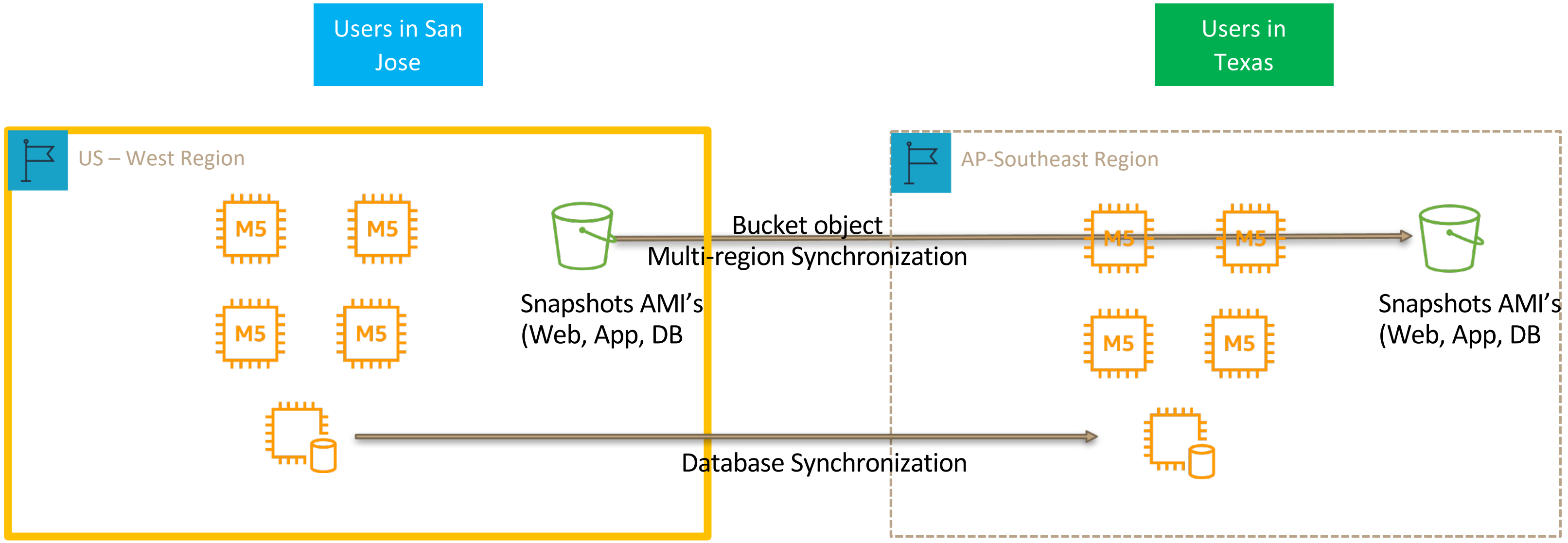
Multi-Region Warm Standby



Multi-Region Warm Standby



Read Local – Write Global



What We Covered

- How to architect solutions using AWS services
- Design solutions based on AWS’s well architected framework
- Design patterns for scale, reliability, and high-availability
- Best practices based on AWS recommended architectural principles

Appendix

Route 53 Routing

Route 53 Features

- Resolve with recursive DNS for VPC and on-premise networks
- Global traffic management; direct users to the correct endpoint
  - Geo-location, health, latency, and weighted round robin
- Health checks and monitoring of resources
- Zone apex support for CloudFront distribution and S3 hosted websites
- Domain Registration

Route 53 Hosted Zones

Public Hosted Zone

- Create hosted public zone
- Add domain name
- Add name server records

Private Hosted Zone

- Change VPC settings to true
  - enableDnsHostnames
  - enableDnsSupport
- Create hosted private zone
- Add domain name
- Associate VPC with the hosted zone
  - Private hosted zone for Amazon VPC

Route 53 Routing

- API Gateway – route traffic to your hosted APIs
- CloudFront – route traffic for your domain to your CloudFront distribution
- EC2 instances – route traffic to websites hosted by EC2 instances
- Elastic Load Balancing – route traffic to your classic, application, or network load balancer
- Elastic Beanstalk – route traffic to your application (elastic beanstalk environment)
- RDS – associate your domain name with the domain name of your database instance
- S3 – route traffic to an S3 bucket, or hosted website
- VPC – route traffic to an interface endpoint

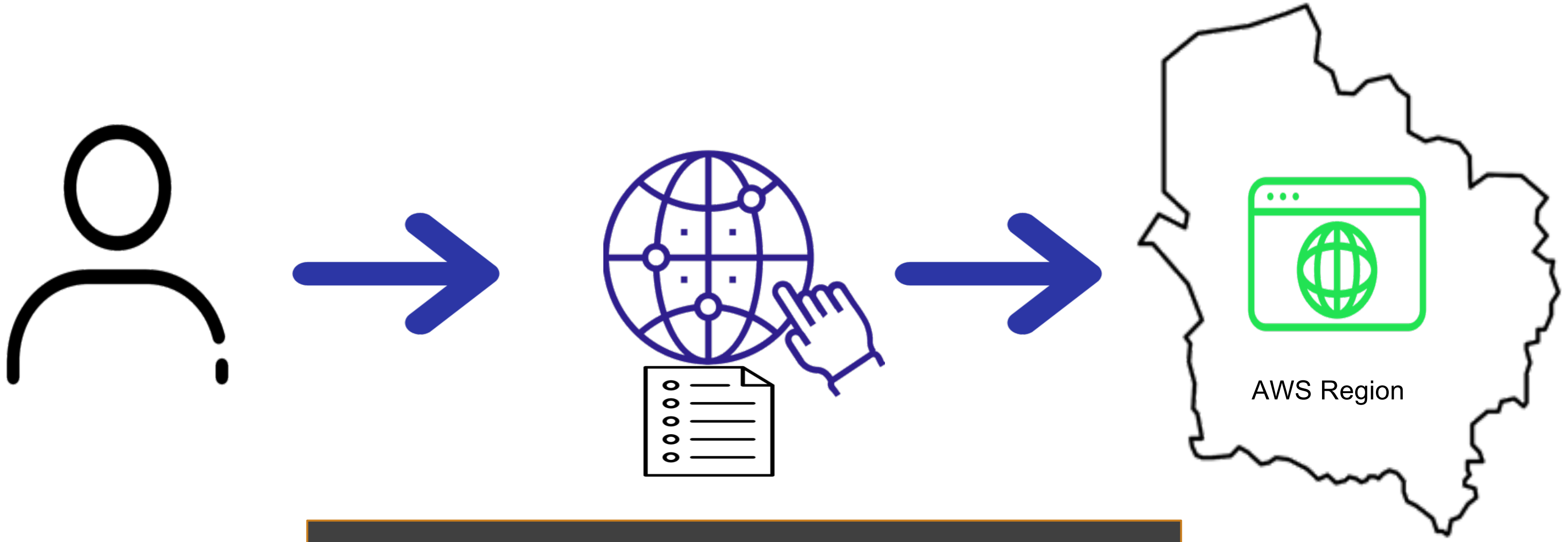
- CNAME (canonical name record) points to a DNS record hosted anywhere
- A CNAME record can’t be used for resolving zone apex / naked domain names
- Alias – a Route 53 specific record that maps to Amazon load balancers, CloudFront distributions, Elastic Beanstalk environments, or S3 buckets configured as websites
- Alias records can be created for a zone apex

Routing Policy Types

- Simple
- Weighted
- Latency
- Failover
- Geolocation

Simple Routing Policy

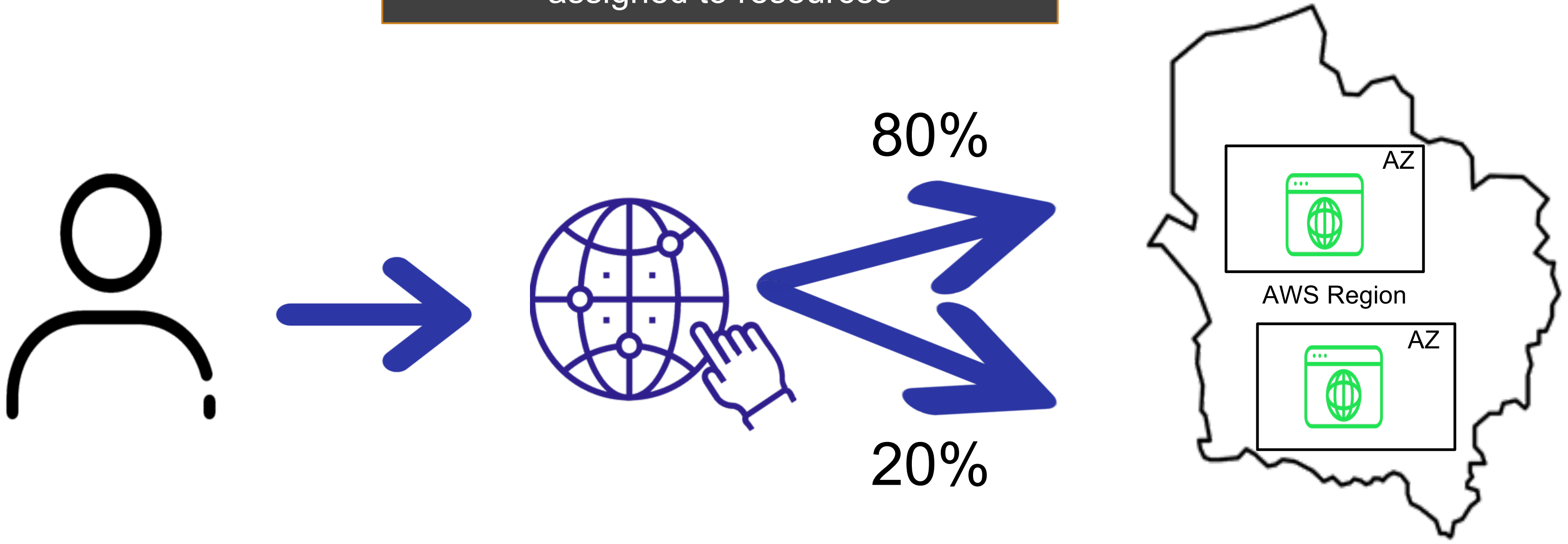
The default routing policy when a new record set is created in Route 53



Used when you have a single Web server

Weighted Routing Policy

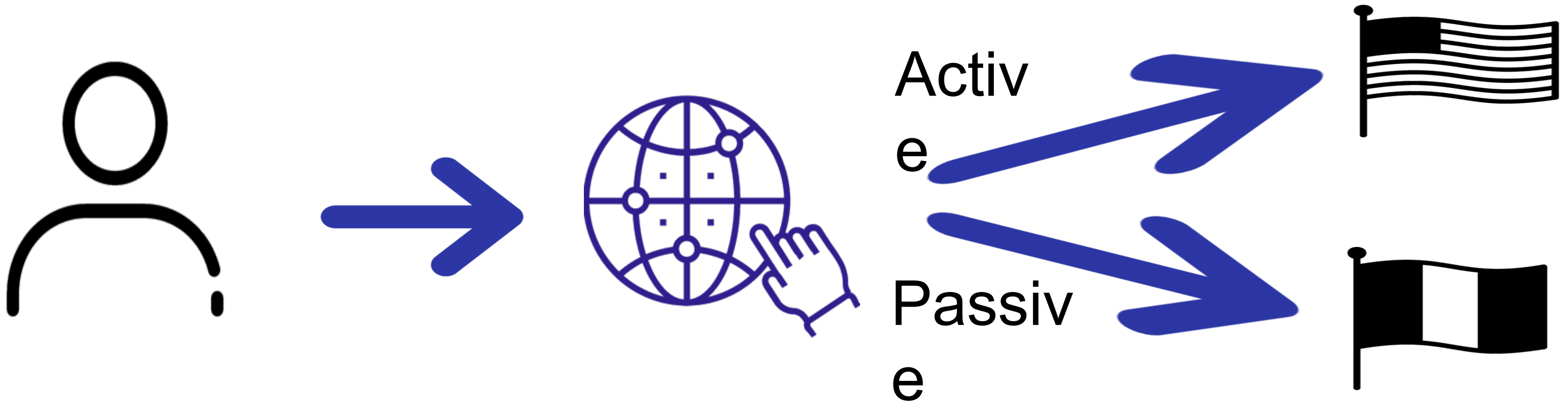
Split traffic based on different weights assigned to resources



Assign 20% of your traffic to one AZ and 80% to the other AZ

Failover Routing Policy

Highly available ELB load balancers



Create health check status; health of ELB, health of entire site

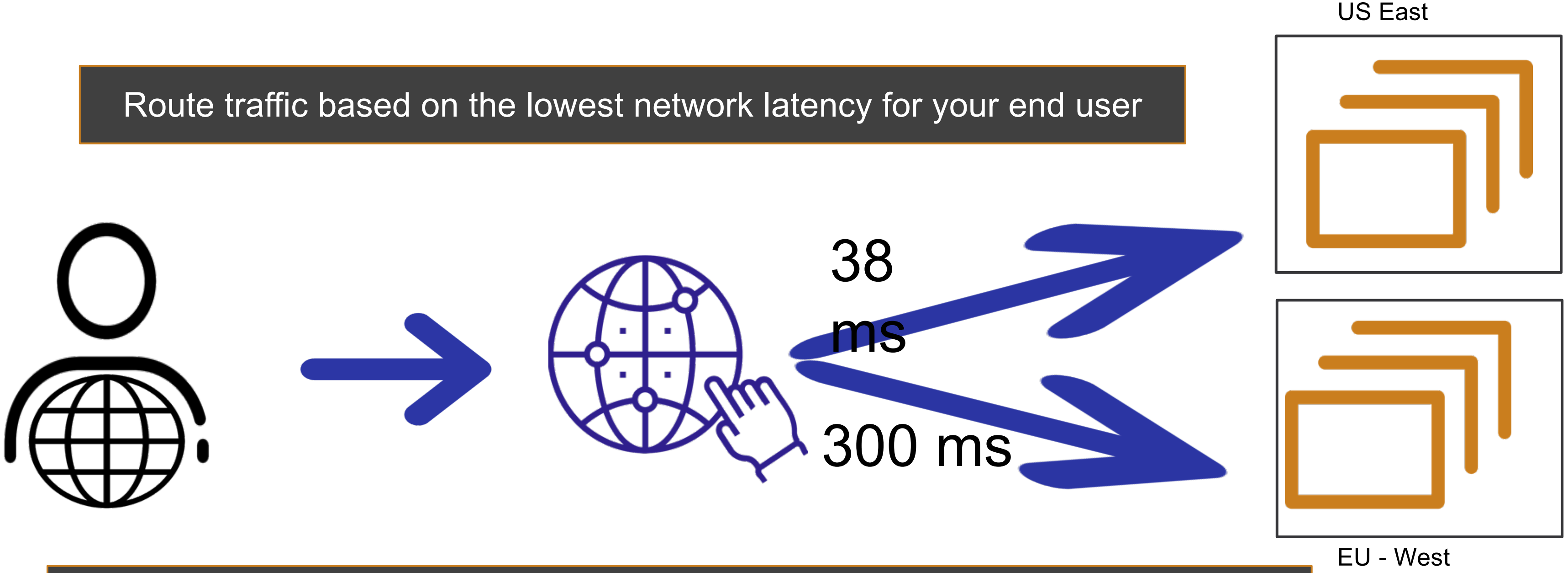
Create record sets, one ELB primary, the other ELB secondary

Route 53 Health Checks

- **Health checks that monitor endpoints**
  - IP address or domain name
  - Automated requests verify resources are available
  - Health checkers interval: 10 seconds or 30 seconds
- **Health checks that monitor other health checks**
  - Create parent health check that monitors the status of other child health checks
  - Compares child health checks against defined minimum number of health checks
- **Health check that monitor CloudWatch alarms**
  - Monitor the status of CloudWatch metrics
  - Data stream is monitored for alarm state

Latency Routing Policy

Route traffic based on the lowest network latency for your end user



Create latency resource record set in each region that hosts your resource

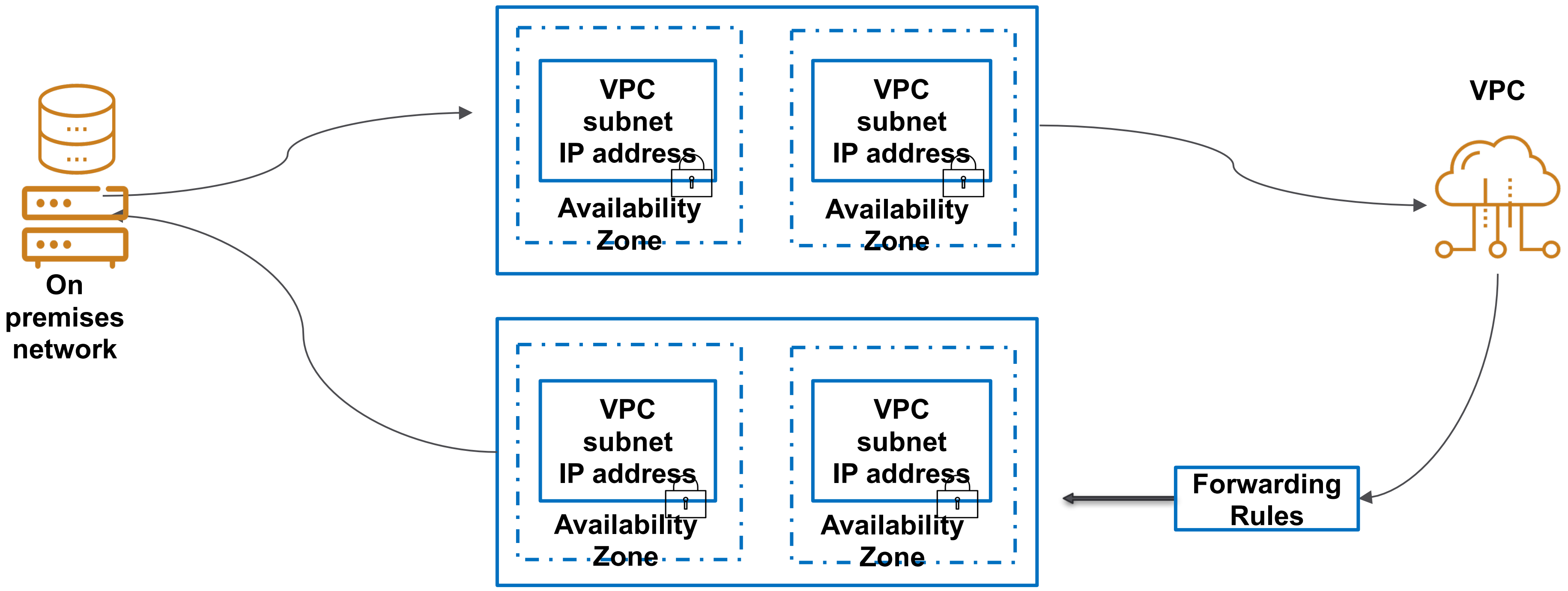
Route 53 selects the latency resource record for the region with the lowest latency

Route 53 Hybrid Routing

- 10.0.0.2 is default reserved VPC address for DNS queries for public and private domains
- New functionality can now resolve hybrid cloud queries
- Route 53 Resolver Endpoints for inbound queries
  - DNS queries from on premise can resolve to AWS hosted domains (VPC)
  - Endpoints are configured with assigned IP address resolvers in each VPC subnet
- Conditional Forwarding Rules for outbound queries
  - On premise domains can be configured as a Route 53 forwarding rule
  - Outbound queries will trigger forwarding to on premise domain

Inbound and outbound private connectivity use VPN or Direct Connect

Route 53 Resolver



Route 53 Alias Records

- Alias records extend DNS functionality
- Route traffic to selected AWS resources; CloudFront, ELB, S3 buckets
- Alias records allow you to route traffic from one record in a hosted zone to another record

Route 53 Alias Records

Route 53 DNS queries alias records:

- API Gateway – multiple addresses for requested API
- VPC interface endpoint – multiple IP addresses
- CloudFront distribution – multiple IP addresses for edge servers
- Elastic beanstalk – multiple IP addresses
- ELB load balancer – multiple IP addresses for the load balancer
- S3 bucket hosting a static website – multiple IP addresses for the S3 bucket