

Intelligent and Predictive Failover

- The race against errors by marching towards Resiliency..!!

DEVELOPERWEEK™ MANAGEMENT




TEJA SWAROOP MYLAVARAPU
(*Lead Software Engineer - Capital One*)






AGENDA

1. Terminology Definitions
2. What is Resiliency?
3. RTO, RPO and MTD
4. Infrastructure Premise with Alarms and notifications
5. Failover Mechanism - Active-Active Failover
6. Distributed System tightly intertwined with Downstream Systems
7. Failover Mechanism - Reactive Failover
8. Concept of Predictive Failover
9. Predictive Failover - Intelligent and Deep Health Checks
10. Intelligent Monitoring with ML
11. Synthetic traffic and disaster simulation
12. Chaos Engineering

AWS Infrastructure- Cluster vs EC2 vs Docker Containers

Definitions		
	Container	A Standalone, executable package that includes all the configs needed to run an application in any environment/OS
	EC2 Instance	A virtual server in Amazon Web Services (AWS). Similar to Virtual Machines in Azure
	Database	A logical collection of structured information acting as a storage mechanism.

AWS Infrastructure - Lambdas, Load Balancer and Route53

Definitions		
	Lambda	AWS Lambda is an event-driven, serverless computing platform managed by AWS which can be used for event-driven or web based applications
	Load Balancer	Distributes incoming application traffic across multiple target groups such as EC2 Instances or Lambdas
	Route 53	Scalable Domain Name System (DNS) web service which connects user requests to AWS Components

What is Resiliency?

Textbook Definition: The ability of an application to resist or recover from any disasters related to Infrastructure, downstream systems, traffic spikes, network issues.

It is not only the ability to recover from downtimes, but also the ability to avoid the downtimes and still be consistent with security, consistency and performance.

Resilience = Brand Value

Overall low Customer Impact, low failures and lower wait time = Better trust + increase in Customer confidence in general.

Companies prioritize their investments on Infrastructure and Resiliency, but the Question is - “Are they doing it right?”

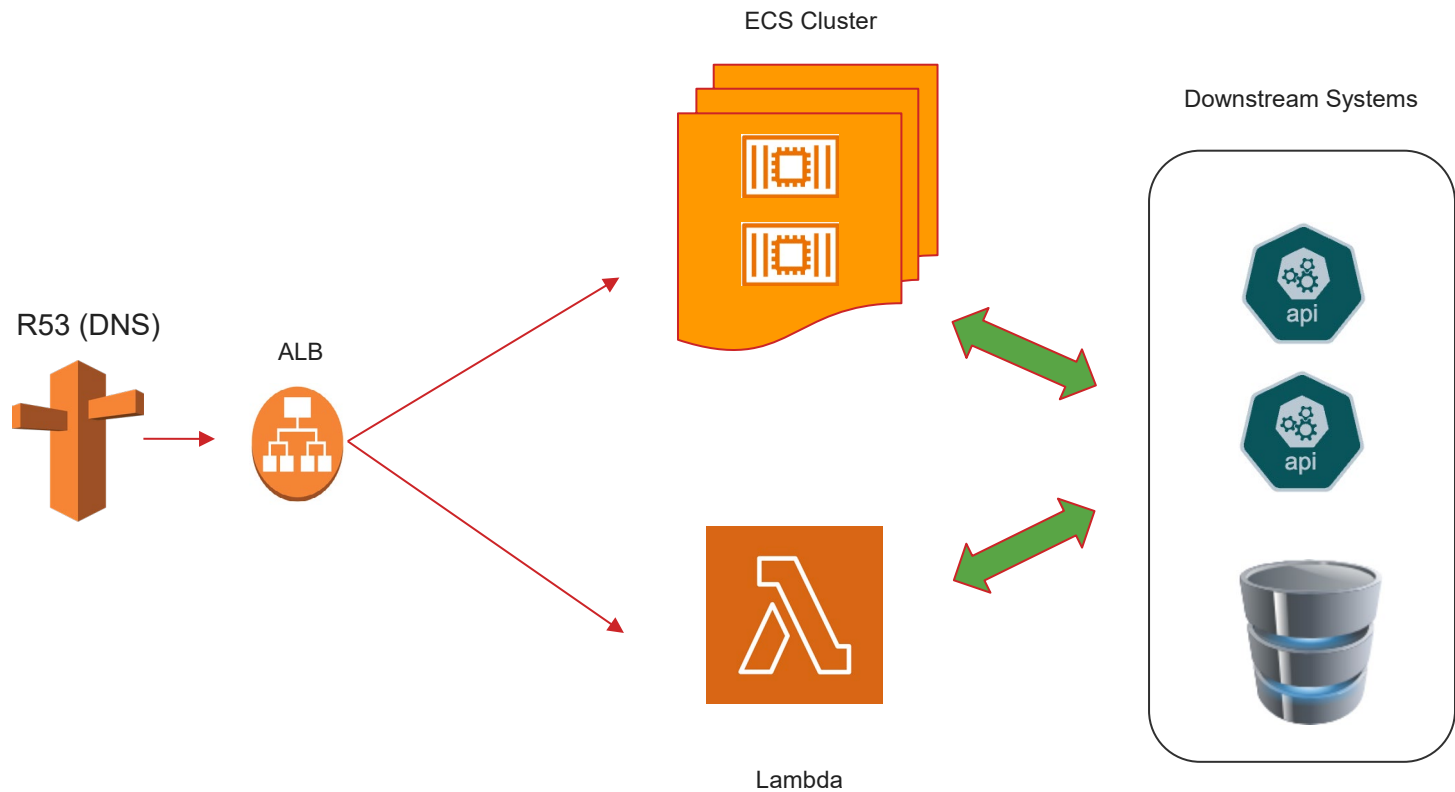
What is RTO, RPO and MTD?

Recovery time objective (RTO): The time that an application or a system can be down without causing significant damage to the business. It also refers to the time spent restoring the application and its data to resume normal business operations after a significant incident.

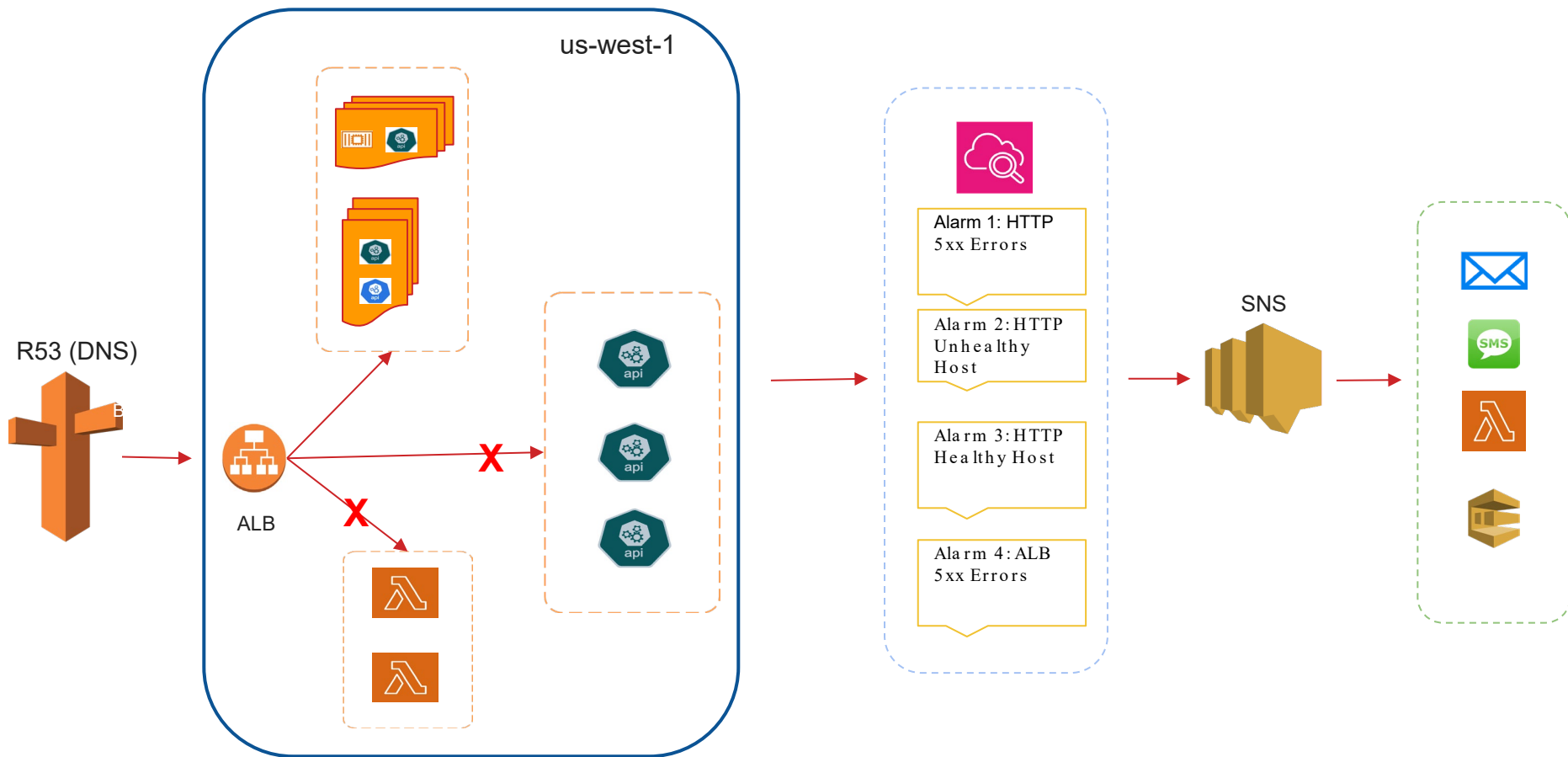
RPO(Recovery Point Objective): With the measure of time, the permissible amount of data that can be lost during a disaster. It is also the age of files that must be recovered from backup for the resumption of normal operations

MTD (Maximum Tolerable Downtime): The maximum amount of downtime that can occur in an organization without causing significant harm to the enterprise mission

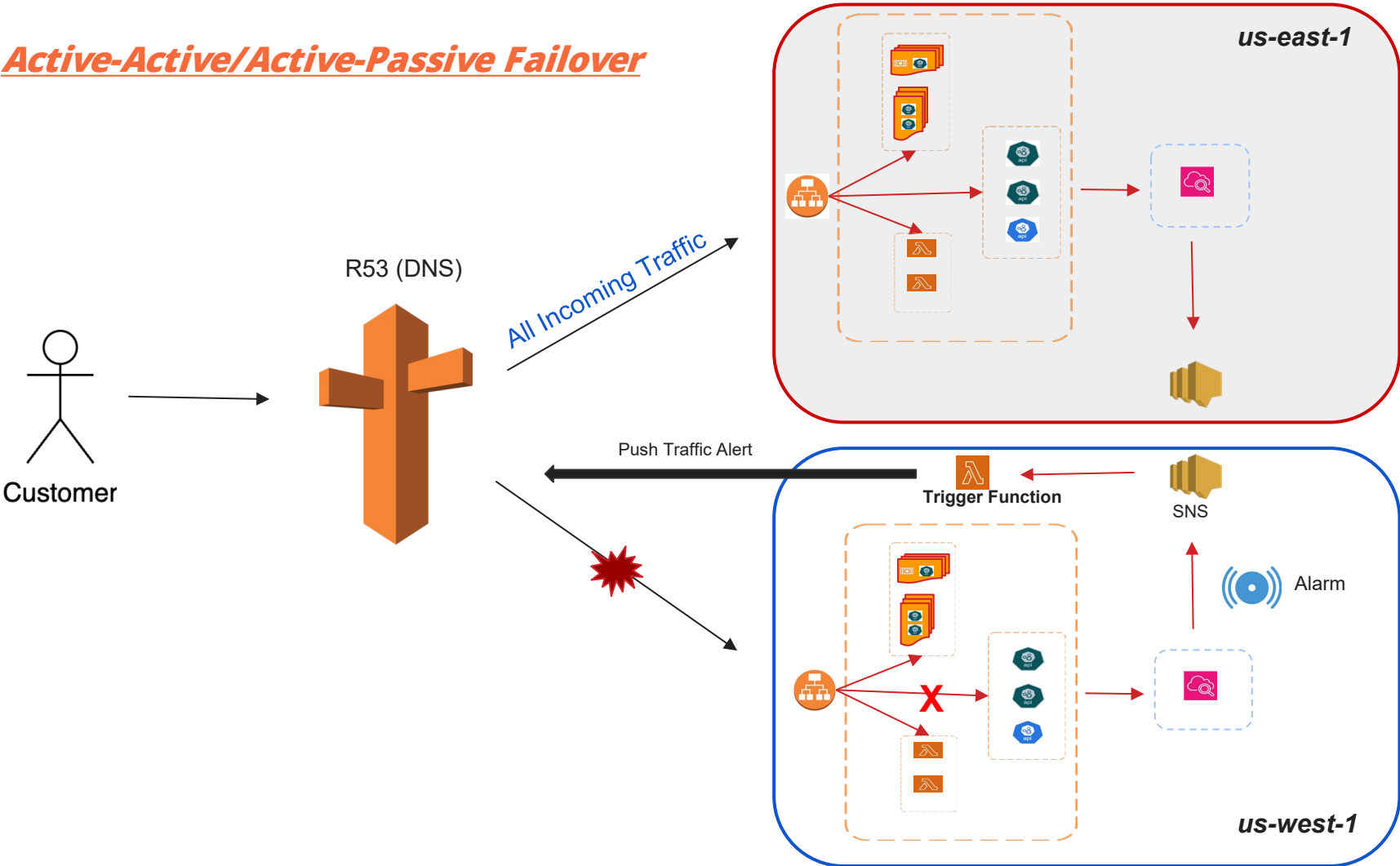
Infrastructure premise - Route 53, Load Balancer and Target Groups



Infrastructure with Alarms & Notifications



Active-Active/Active-Passive Failover

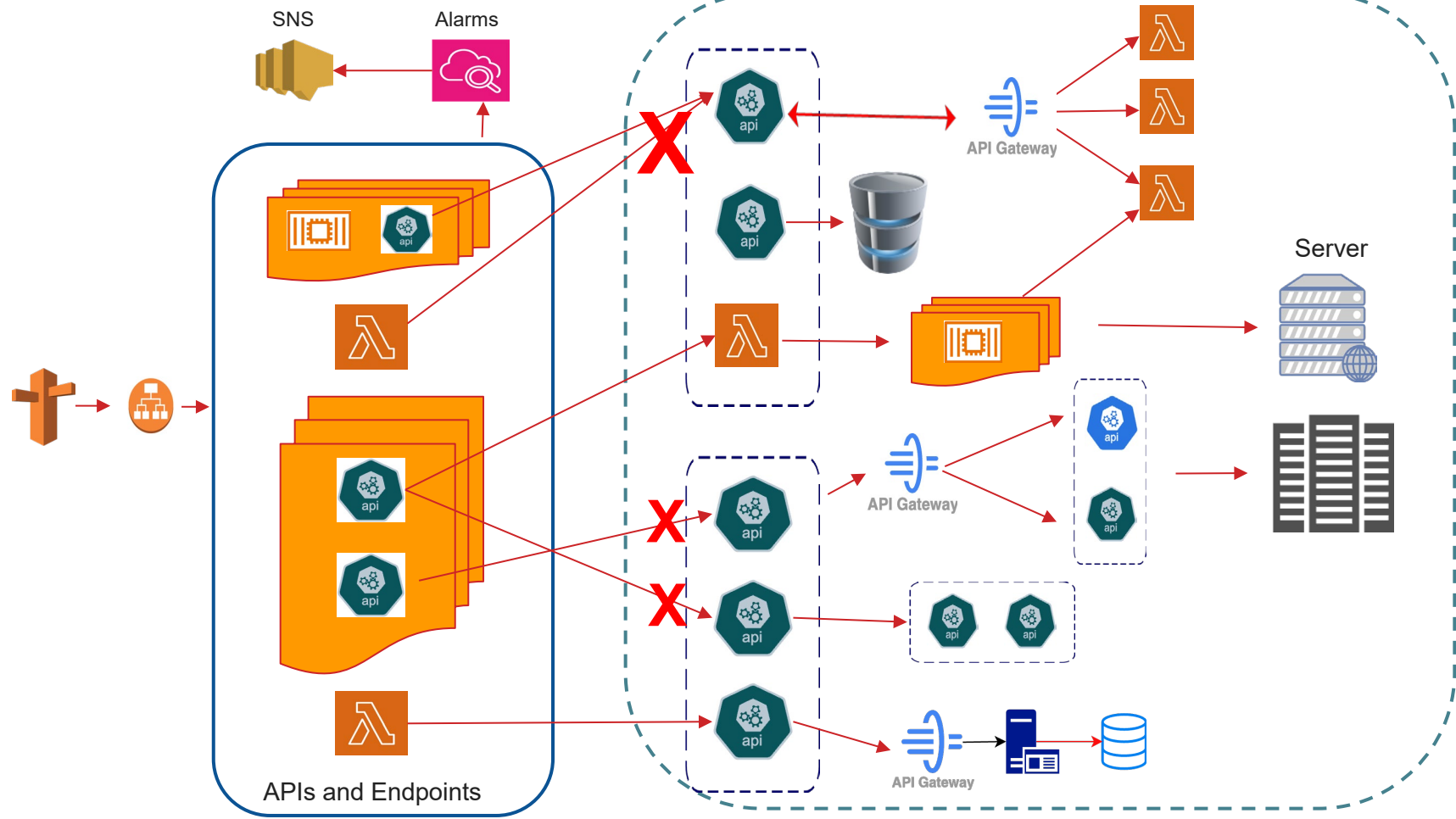


Trigger Function

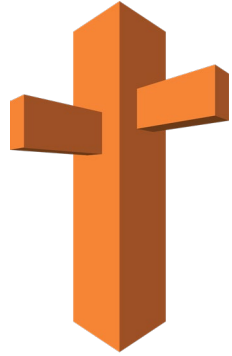
A function which can do the following:

1. Have a list of Alarms that it can track to trigger a failover or a failback.
2. Push the R53 traffic to the other region and fail the Health Check associated with it.
3. Ensure that the traffic stays in the other region till the current disrupted region is back to healthy.
4. Continuously monitor the Alarms for healthy and unhealthiness
5. Help failback the traffic back to normal depending on the thresholds.

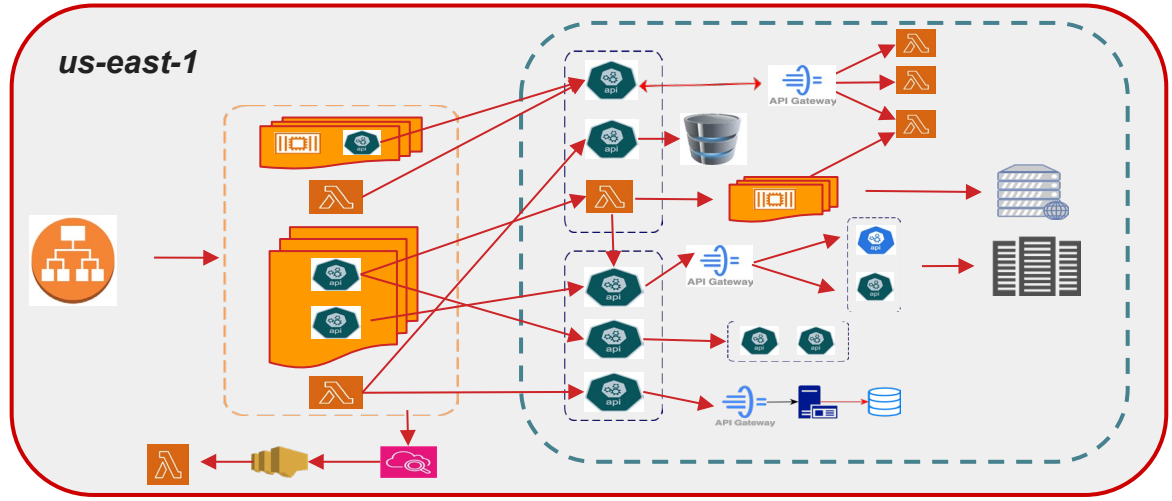
Inter-twined Downstream Systems



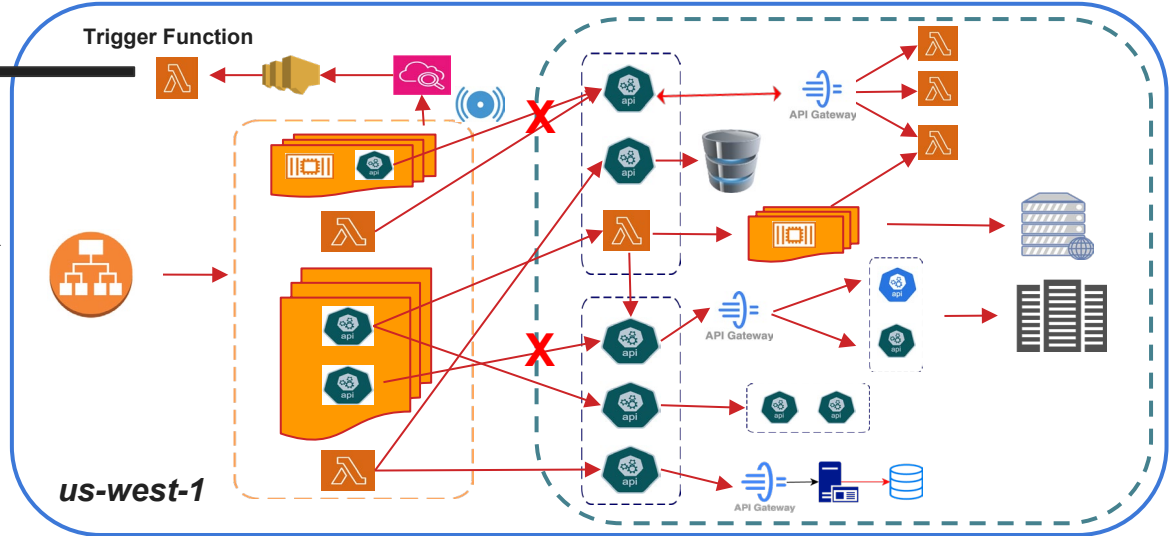
Reactive - Failover - Downstream Disruption



All Incoming Traffic



Push Traffic Alert



Predictive Failover..!!

Can we predict/detect failures?

- *Monitor Deep Health Checks* : A Health check configuration where an endpoint tracks its corresponding deep health checks it is connected to, which in turn tracks its down stream's deep health checks.
- *Monitor Interconnected Health Checks* : For small systems, monitor the individual health checks of all the inter connected systems.

These configurations may prevent/detect errors even without actual traffic and help predict failures.

Predictive Failover - Intelligent and Deep Health Checks

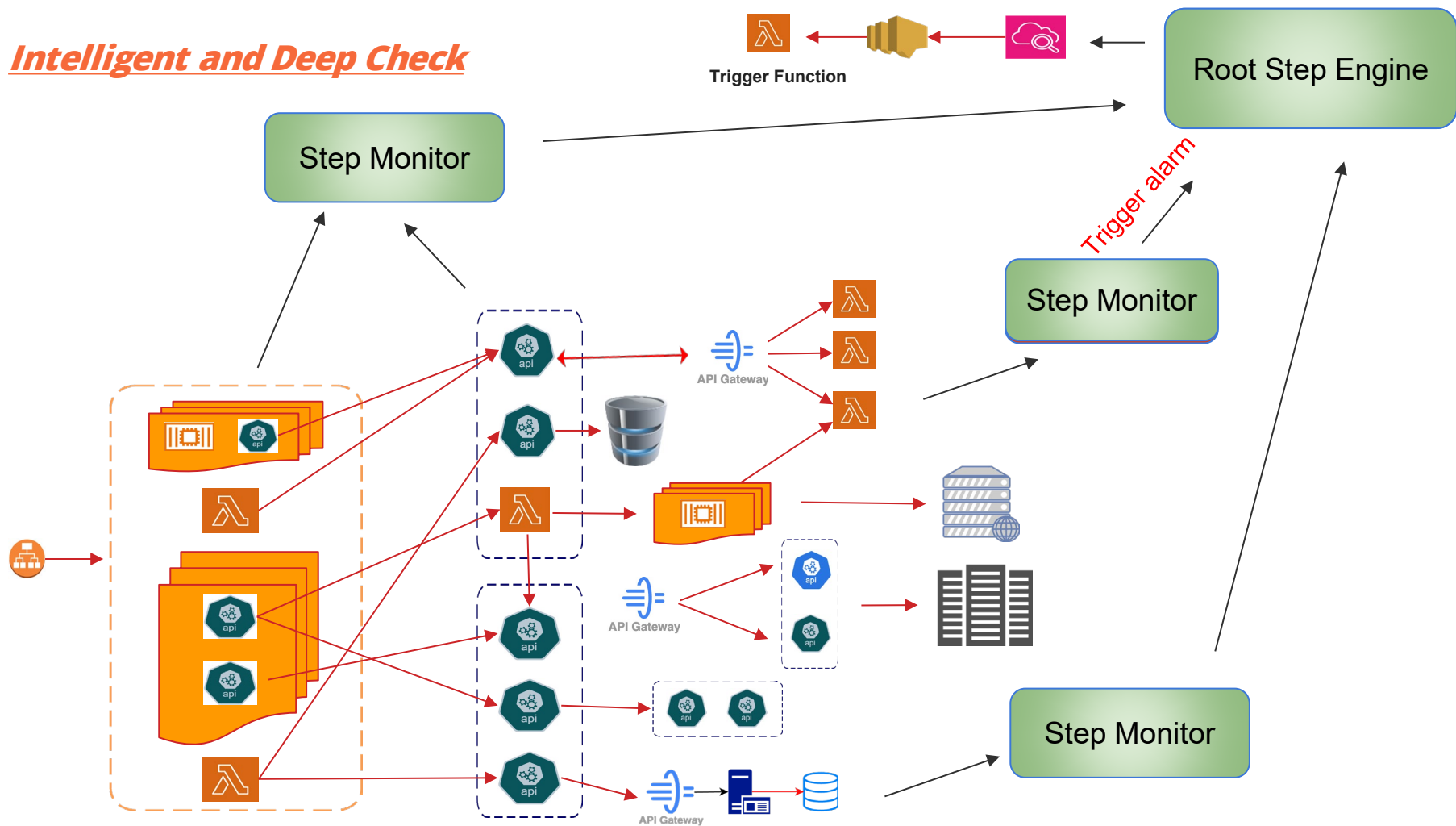
Step Monitor

- Tracks all the Downstream System for Errors, Anomalies and Disruption.
- Tracks deep health checks of the downstream systems.
- Incorporate “Thresholds & Frequency” based monitoring where each endpoint can have different thresholds to trigger Alarms/distress signals.
- Continuously monitor irrespective of the client traffic.
- Detect errors even without actual traffic, reduce customer impact.
- Send Distress Signals to Root Monitors

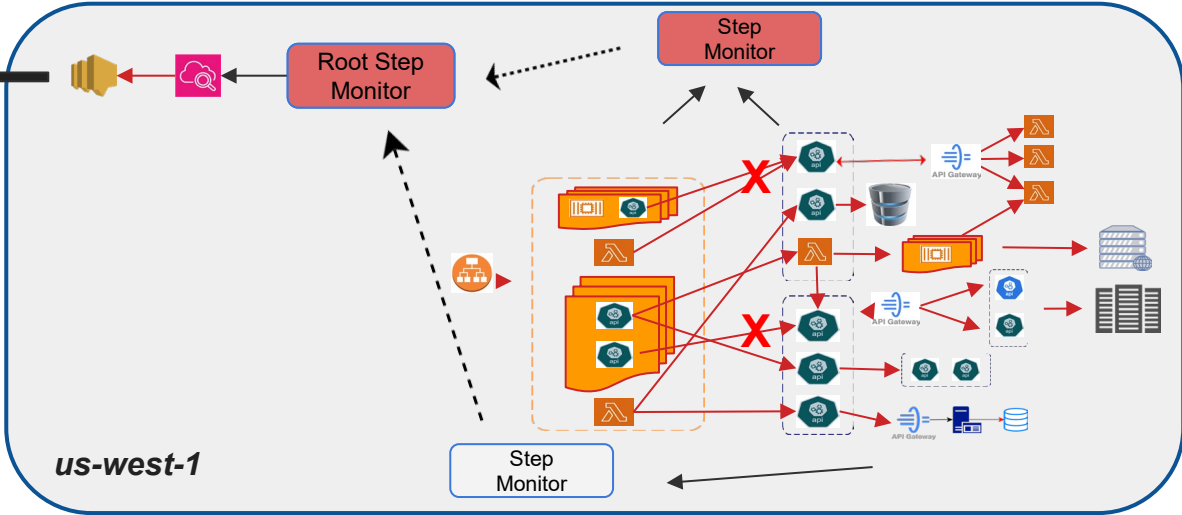
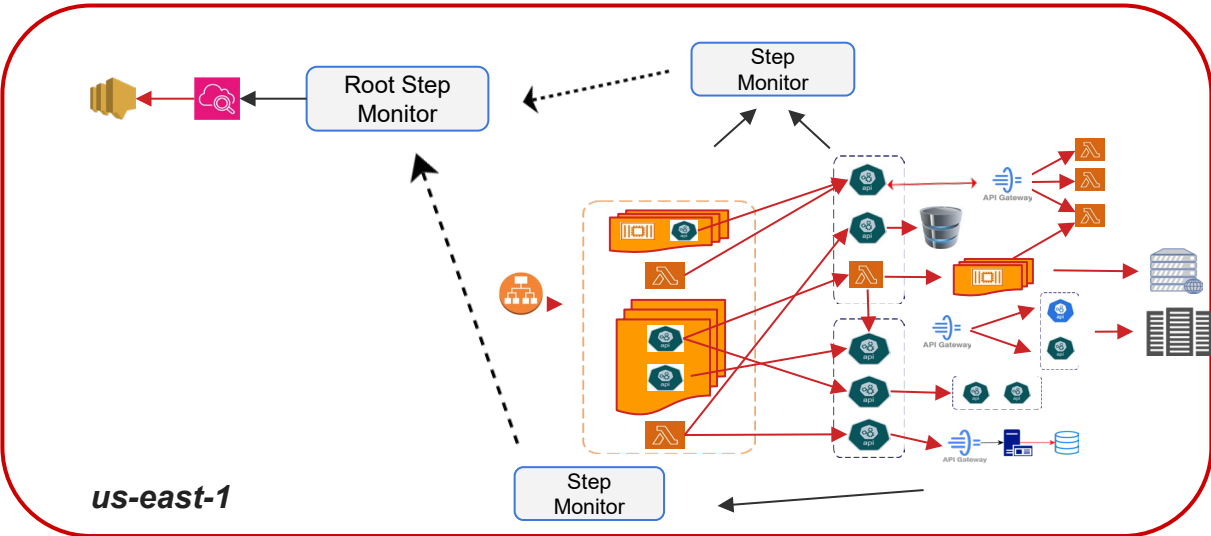
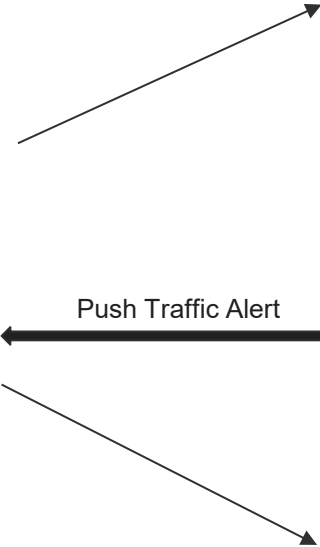
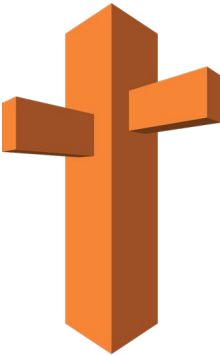
Root Step Engine:

- Pro-Actively gathers and monitors all the Step Monitor Health.
- Pro-Actively react to the Distress Signal and reach out to other region's Smart Root Monitors for Active Smart Failover
- Confirm the non-availability of its child Step Monitor irrespective of Active Traffic coming in.
- Pro Actively take decisions on triggering the appropriate Alarms to notify the failure of downstream systems.
- Evaluate the Latency against all the other Active Regions and smartly failover to the appropriate Region
- Failover the traffic to the next best region.

Intelligent and Deep Check



Predictive Failover



Intelligent Monitoring - Applying Machine Learning:

- ❑ Monitors can have Analytics or Machine Learning Intelligence introduced where they grow over time and smartly figure out based on the history and knowledge on how a particular downstream system can futuristically fail over time and raise the alarm on a predetermined basis and manoeuvre the failure.
- ❑ They can anticipate that a certain Downstream System deploys their Application during a certain time and flip the traffic to the other region during their Scheduled deployment.
- ❑ They can also determine that historically at a certain time an Application fails consistently in a certain region and can flip the traffic to the other region and so on.
- ❑ Based on auto healing approach of the downstream system, the Step-up monitors can alternate the frequency and thresholds of the Alarms.

Testing Strategies and disaster simulation

- Chaos Engineering: Apply the best principles of Chaos Engineering by disrupting your individual Application and regions.
- Performance Testing: Perform different types of Performance testing such as Stress, Load and Endurance testing to test for Auto Scaling and Performance of your Systems.
- Participate in AWS Gamedays.
- Synthetic Traffic: Send required amount of Synthetic traffic into the Infrastructure and simulate disaster mechanisms by pulling down the appropriate APIs and Applications and test for resiliency.

Chaos Engineering

Cornerstone Experiments:

- Deleting Lambda Concurrency
- Deleting the EC2 Instances
- Database Connectivity Disruption
- Apache Kafka - Interrupt the network on one AWS Region
- Disk Space Failures
- Denying Traffic to VPC

Chaos Engineering Tools (can be used independently)

- AWS CLI
- AWS SDK
- AWS Fault Injection Simulator
- AWSResilienceHub
- Chaos Monkey
- Chaos Toolkit
- Chaos Blade and etc

Key Takeaways

Setup your Infrastructure

Understand Resiliency and
dependency of Downstream Systems

Setup Trigger Function for Reactive
Failover

Validate Routing, Alarms, Failover
and Failback

Configure Predictive Monitoring
with Deep Health Checks

Perform Chaos Engineering – Repeat
Multiple Times

Be Bold – Break your Infra and march
towards Robust Resilient Systems

THANK YOU!