Arquiteturas Resilientes na Nuvem

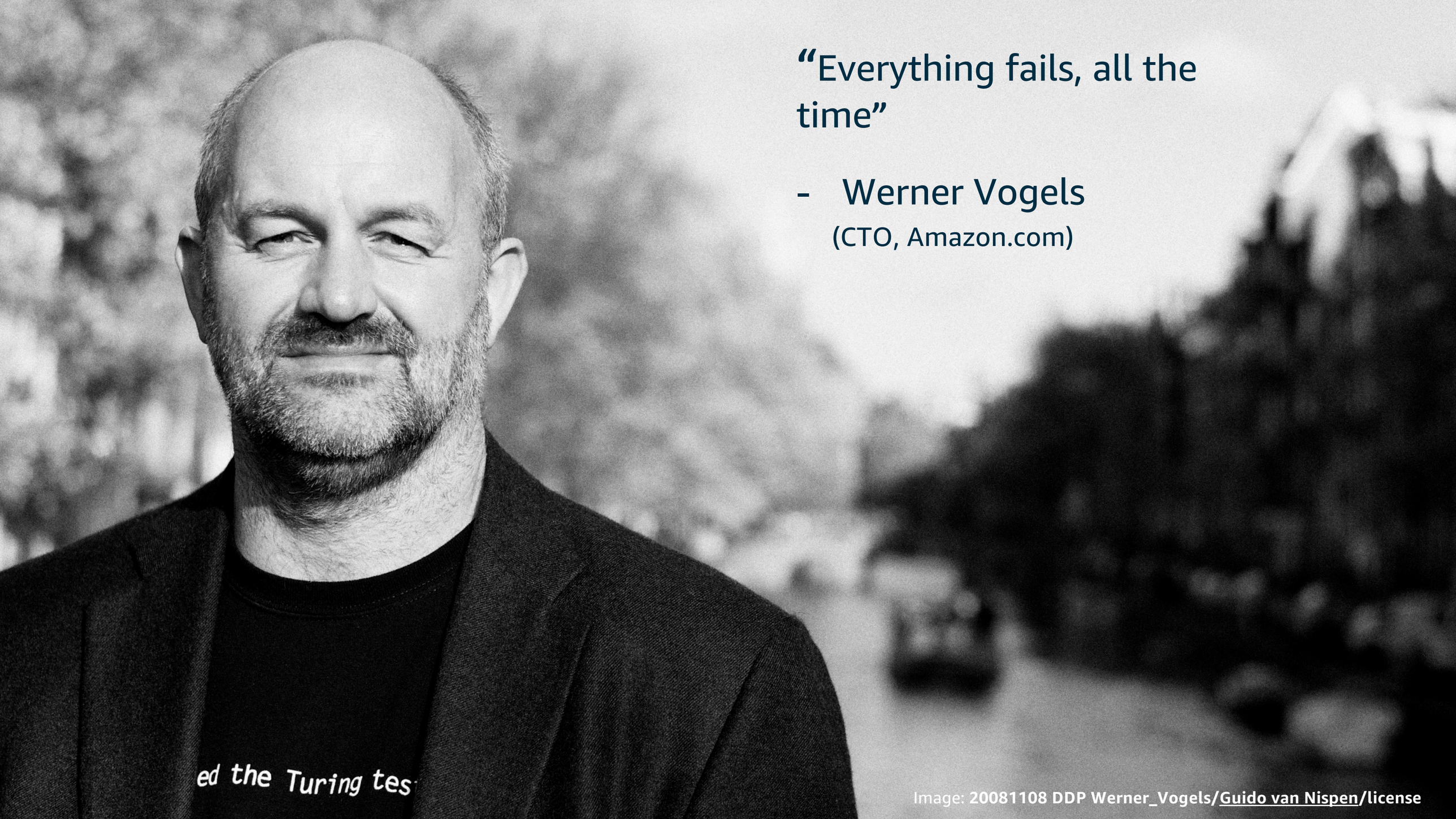
Trilha Arquitetura

Luiz Yanai, Solutions Architect - AWS
Leonardo Piedade, Solutions Architect - AWS

Agenda

- What are we planning for?
- Think resiliently. Principles of Resiliency
- System Architecture Blueprints
- Lessons Learned

aws



“Everything fails, all the time”

- Werner Vogels
(CTO, Amazon.com)

Image: **20081108 DDP Werner_Vogels/Guido van Nispen/license**

Resiliency is the ability for a system to recover quickly and continue operating even when a failure occurs

aws

What are we planning for?

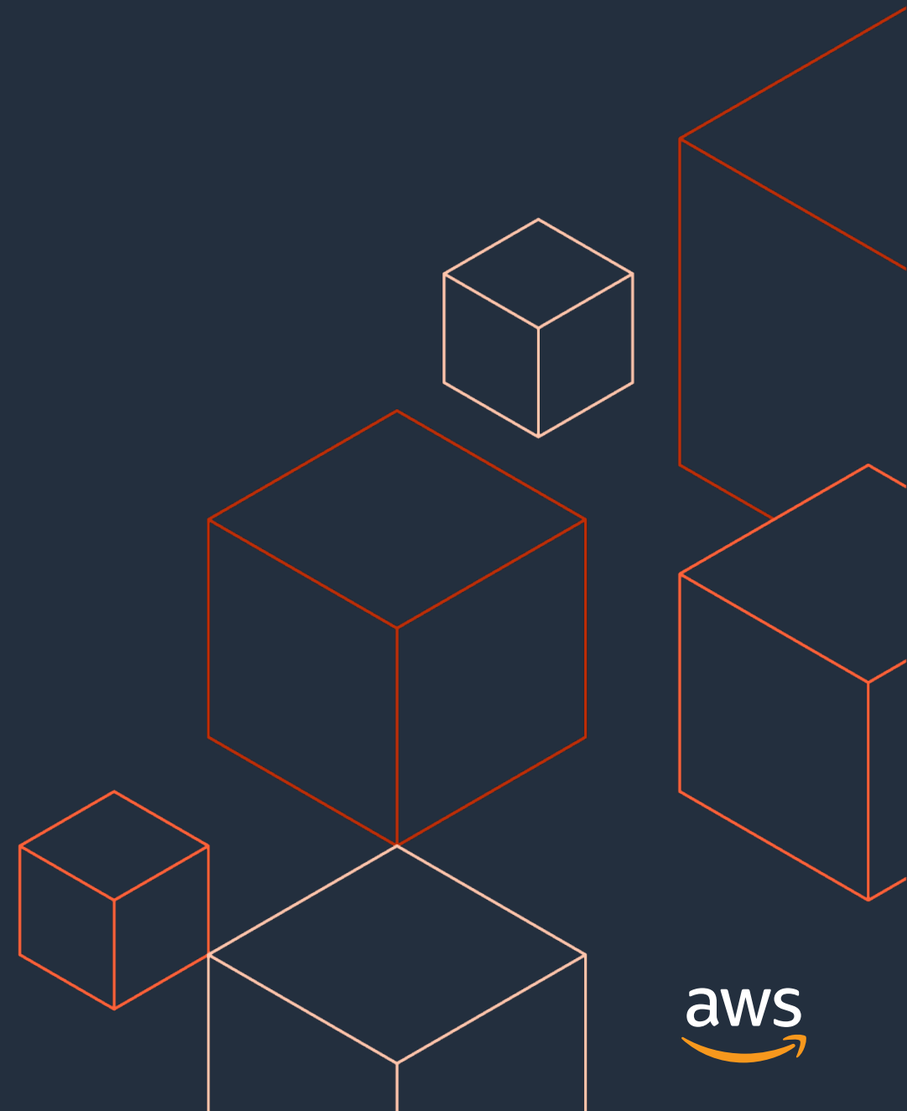
Bad Things Happen

Source: https://www.datacenterdynamics.com/en/news/fire-destroys-ovhclouds-sbg2-data-center-strasbourg/

Source: https://www.forbes.com/sites/lealane/2020/04/04/are-you-ready-for-this-2020-hurricane-forecast-above-average-intensity/

aws

Think Resiliently
Principles of Resiliency



aws

Recovery Point and Recovery Time Objective

aws

Resilient AWS Cloud

Infrastructure

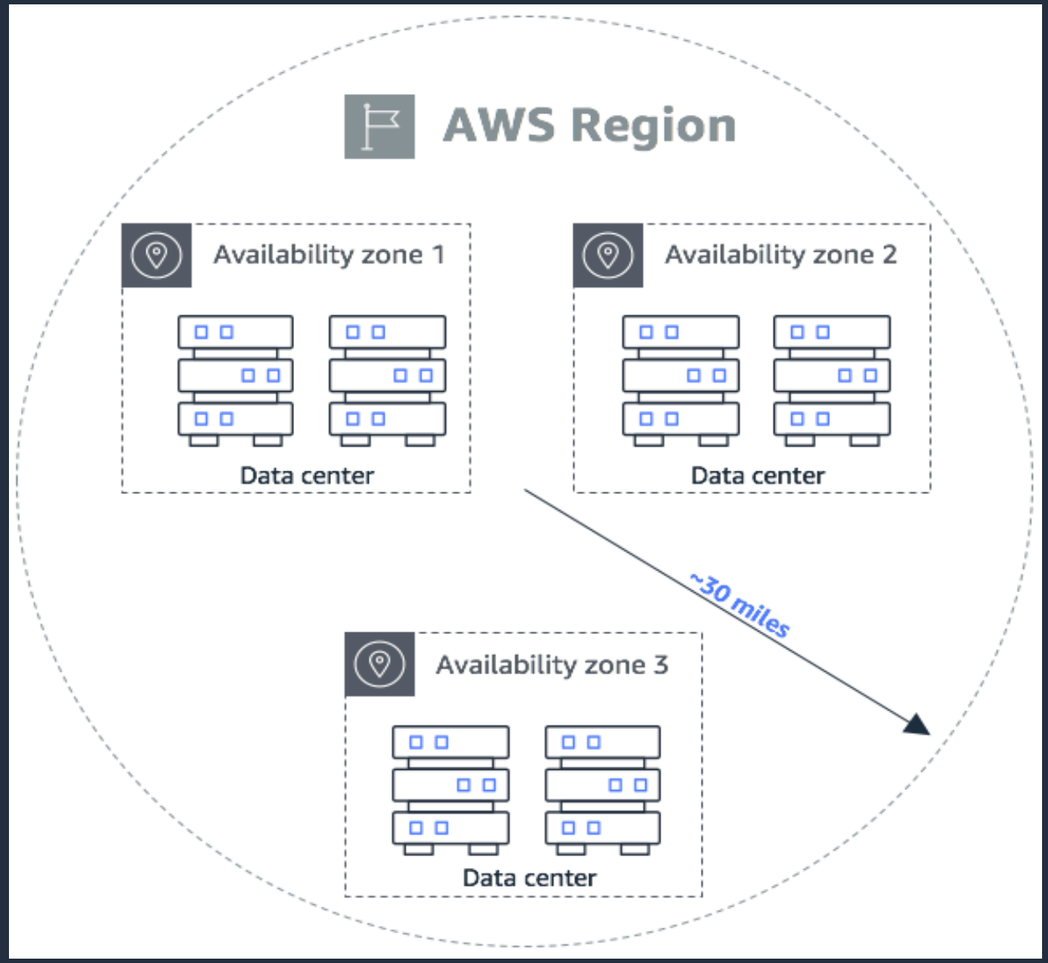
- Regions, AZs

Service Design

- Distributed systems best practices

Understand the AWS Services scope

- Single AZ, Regional, Global, Cross-Regional capability



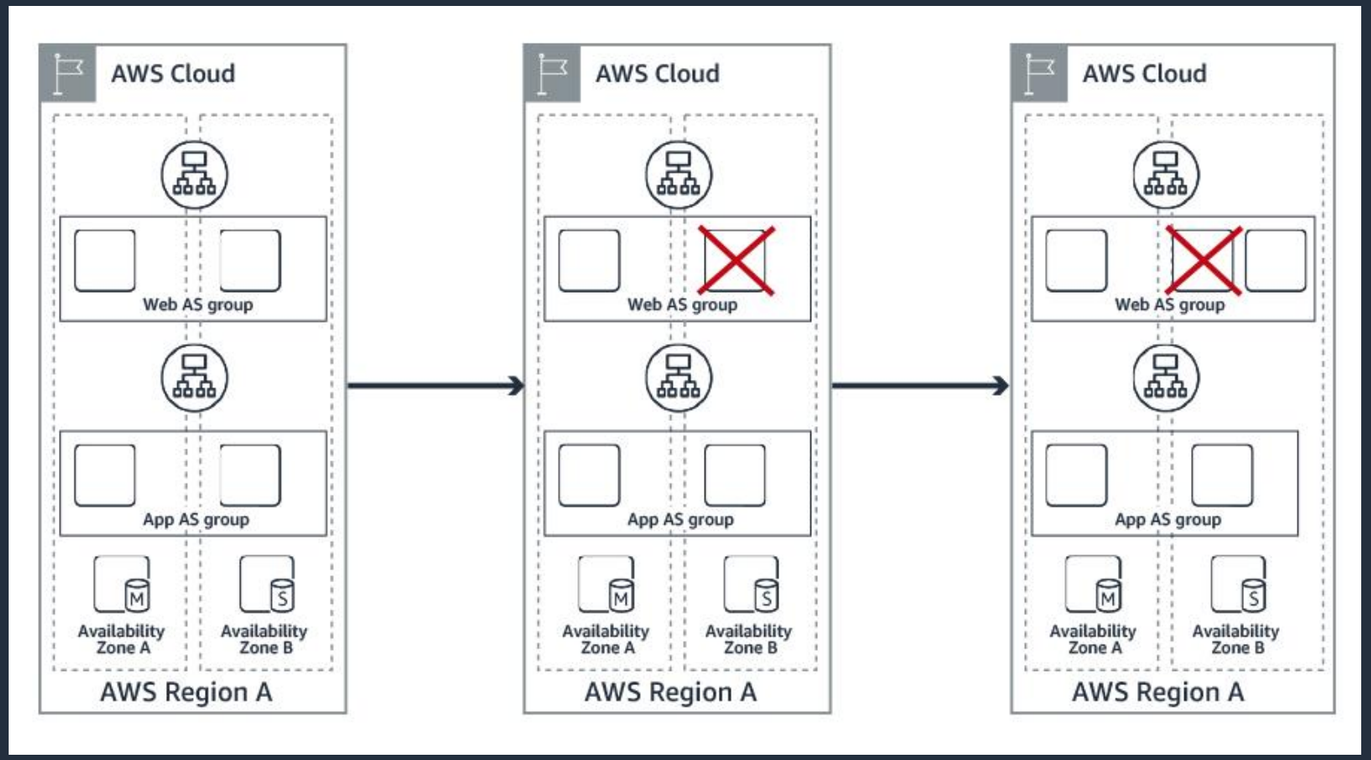
aws

Self-Healing applications

Highly resilient applications must be able to self-heal.

How

- Leverage Microservices app architecture
- Decouple inter-dependencies, loose coupling
- Remove state from app components



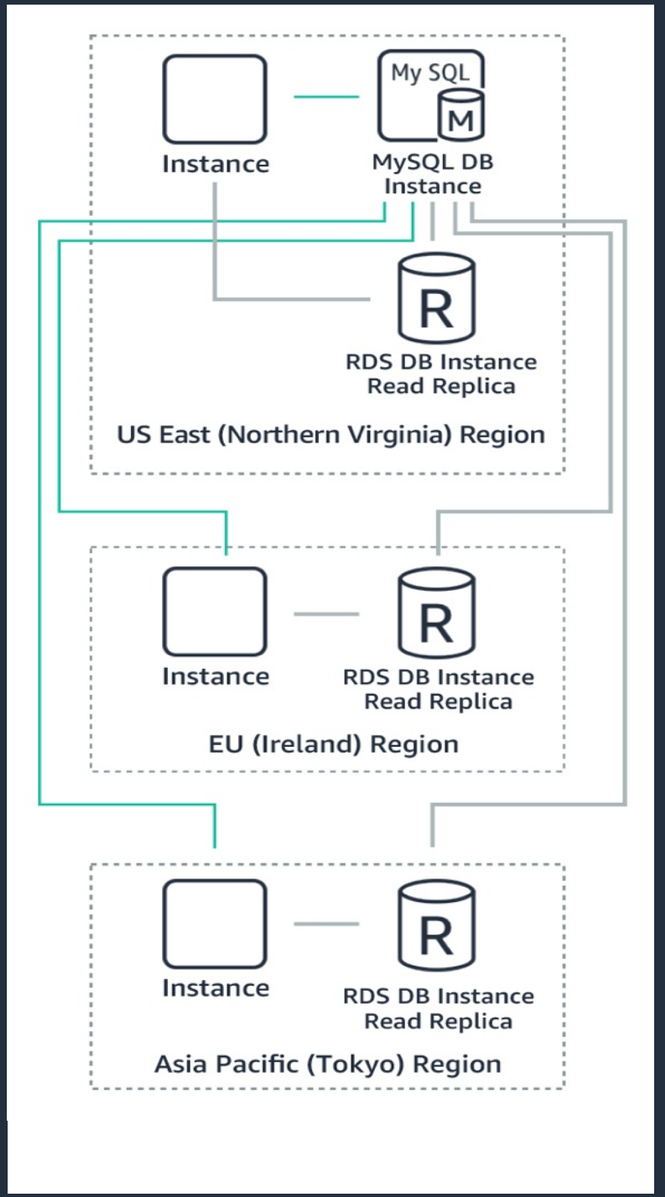
Resilient Data

Must have confidence in the resilience of your data

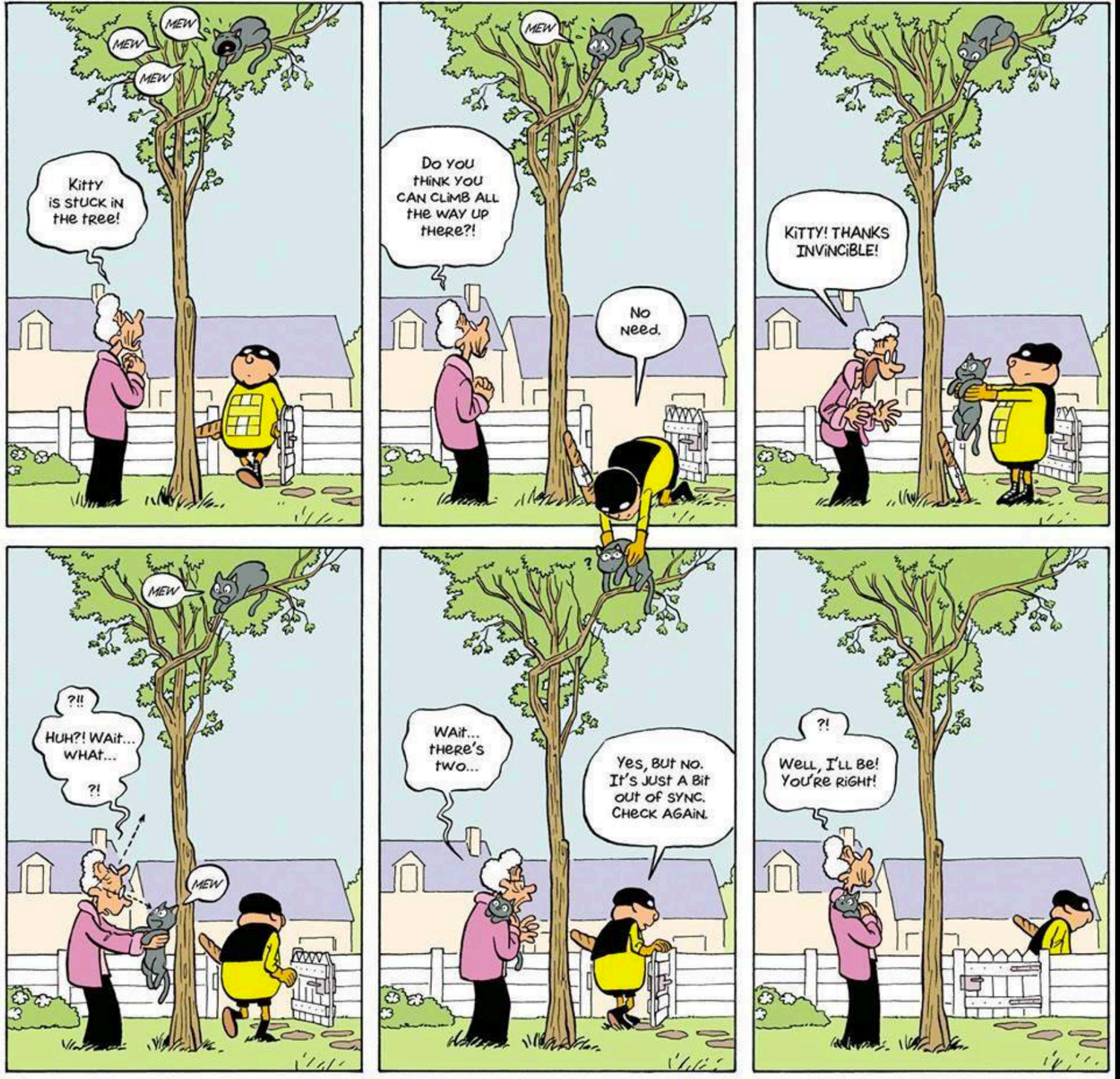
Many forms:

- filesystem,
- block storage,
- databases
- in memory caches

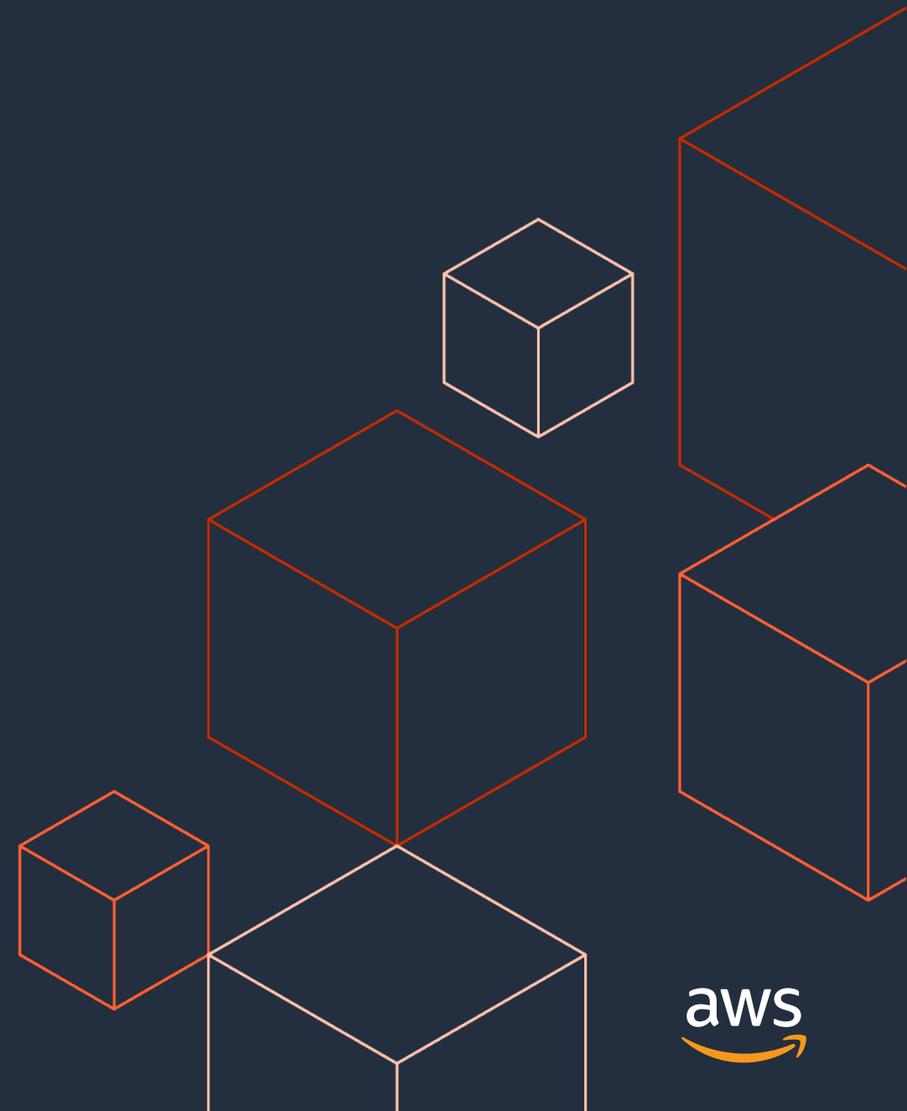
Consider how eventual consistency impacts design



aws


MEW
MEW
MEW
Kitty is stuck in the tree!
MEW
Do you think you can climb all the way up there?!
No need.
Kitty! Thanks invincible!
MEW
?!!
Huh?! Wait... what...
?!
MEW
Wait... there's two...
Yes, but no. It's just a bit out of sync. Check again.
?!
Well, I'll be! You're right!

System Architecture Blueprints



aws

Single AZ

If cost is an important requirement and availability is not a concern

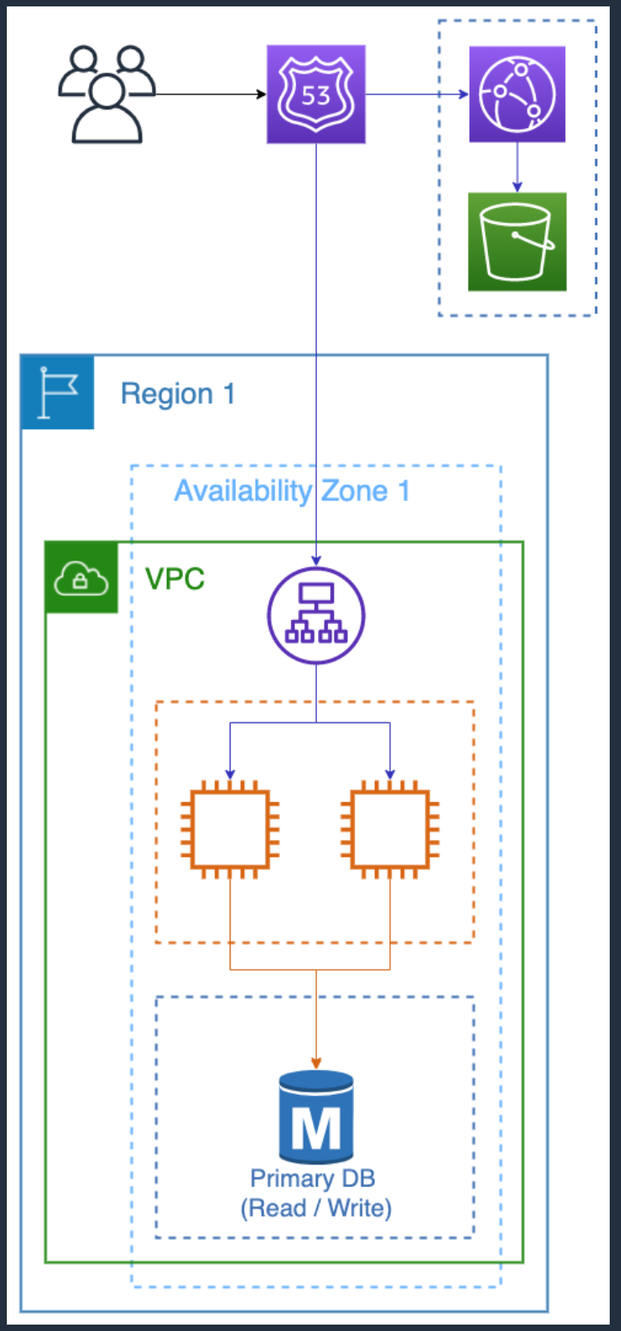
Pros

- Simplicity in design, implementation, and operations.
- Some services offer self-healing features
- It is difficult to achieve this scenario since most services offers AZ resilience by default

Cons

- Slow recovery
- Higher RPO, RTO

Examples: Some MVP’s, prototypes, internal applications



aws

Multi AZ

Start here before adopting more complex architecture

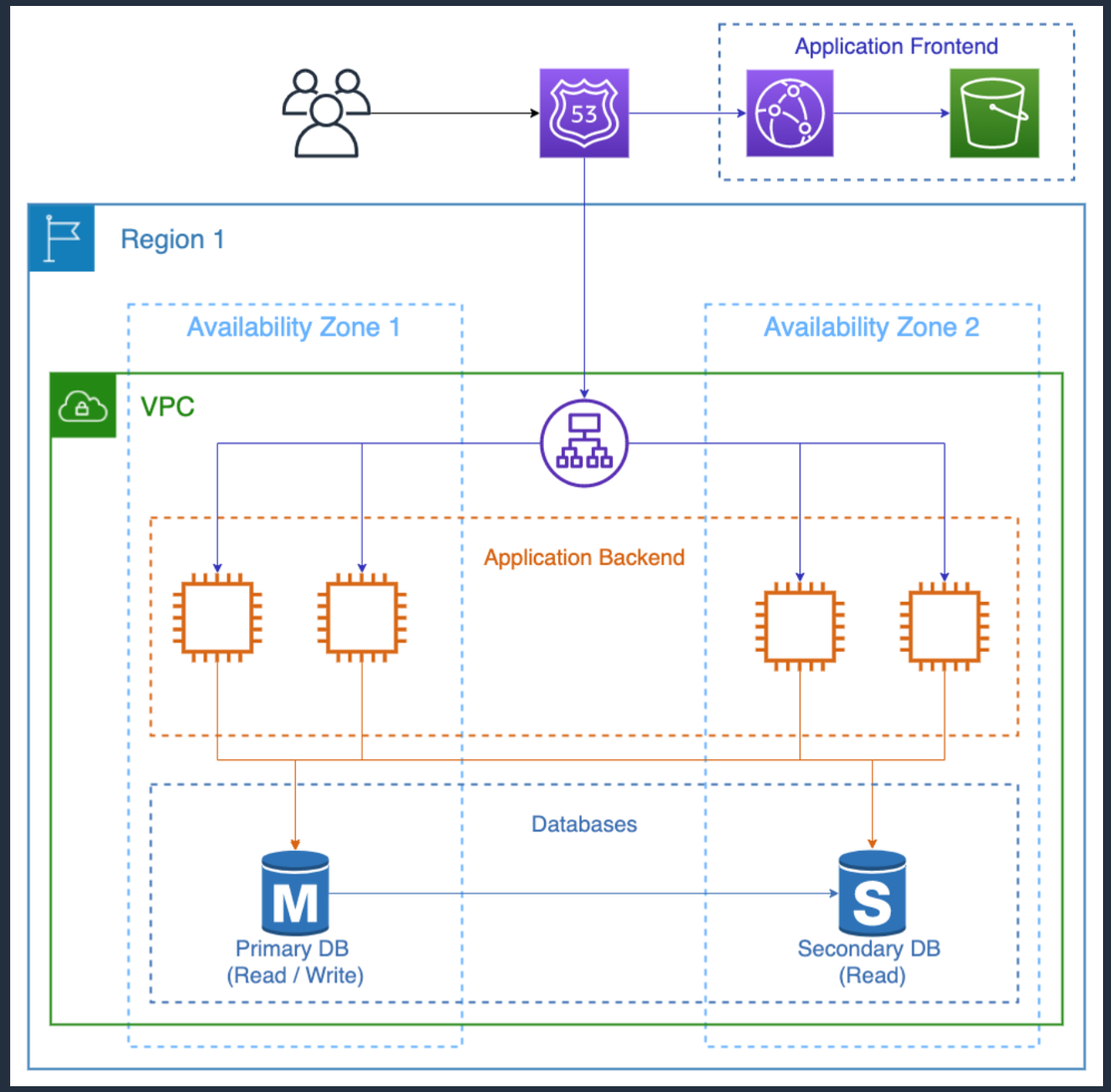
Only consider multi-region if requirements dictate

Pros

- Availability of AWS region-wide services include Amazon S3, Amazon DynamoDB, Amazon EFS, Amazon SQS, Amazon Kinesis
- Much less complexity in design, implementation, and operations.

Cons

- If you need >99.9% availability, consider multi-region.
- May not meet needs of regulators



aws

Multi-Region: Active-Standby

Traditional DR Pattern

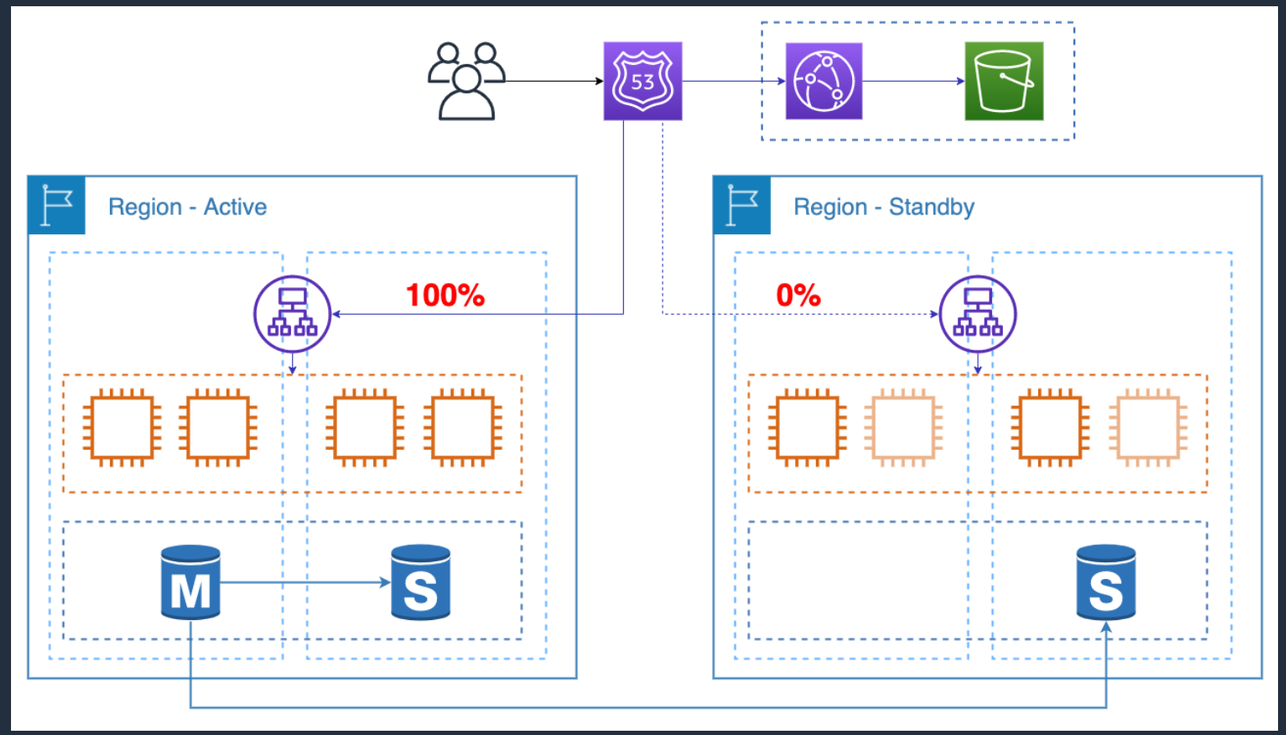
Backup region used in event of failure only

Pros

- For Apps which cannot use native AWS features
- Least # changes to the application

Cons

- RPO limited by replication lag
- RTO, delays while Standby becomes Active



aws

Multi-Region: Active-active

Both stacks active, traffic distributed

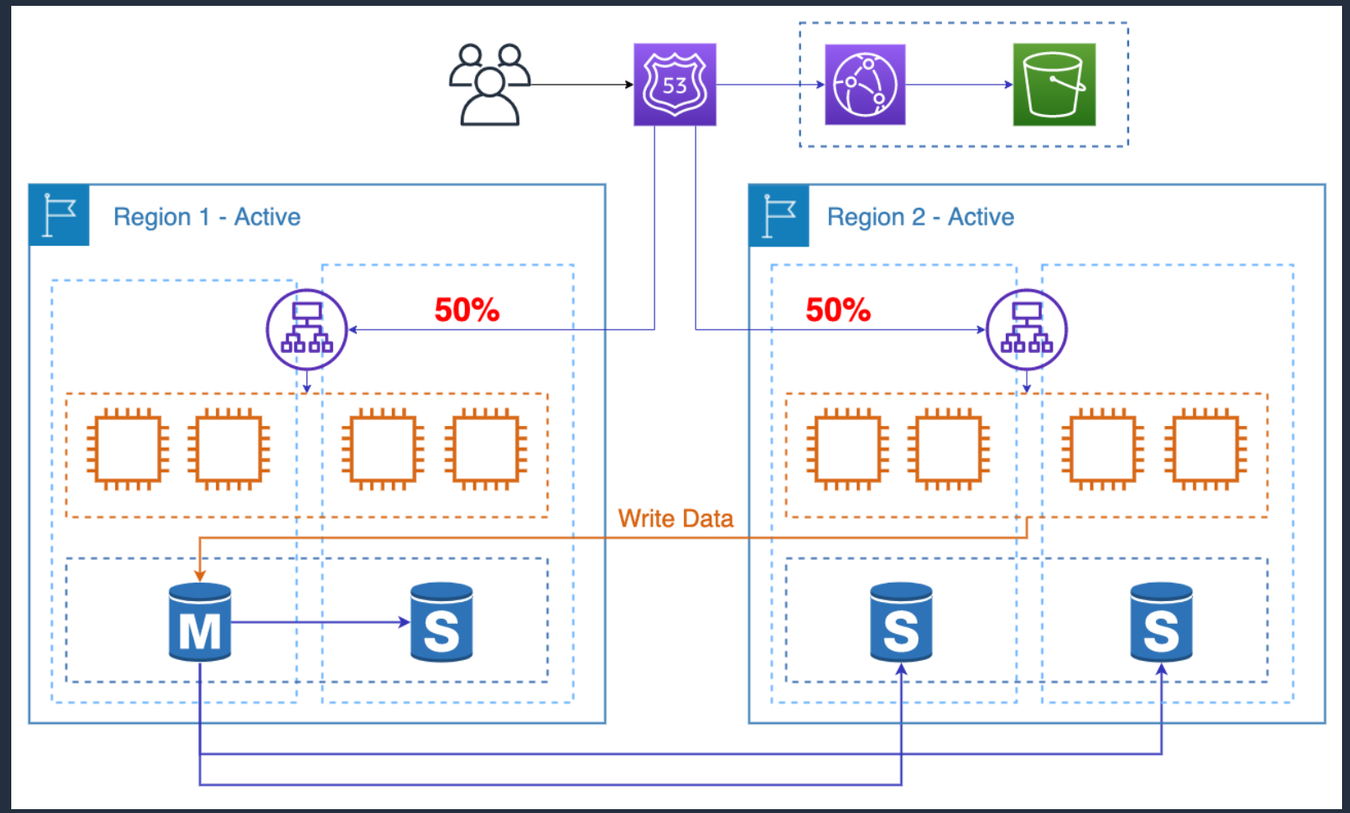
Data replication critical, must consider latency impacts

Pros

- Zero RTO
- Works well for apps that can partition users

Cons

- Data replication must be handled by Applications



aws

Multi-Region: Dual-write

Shared nothing architecture

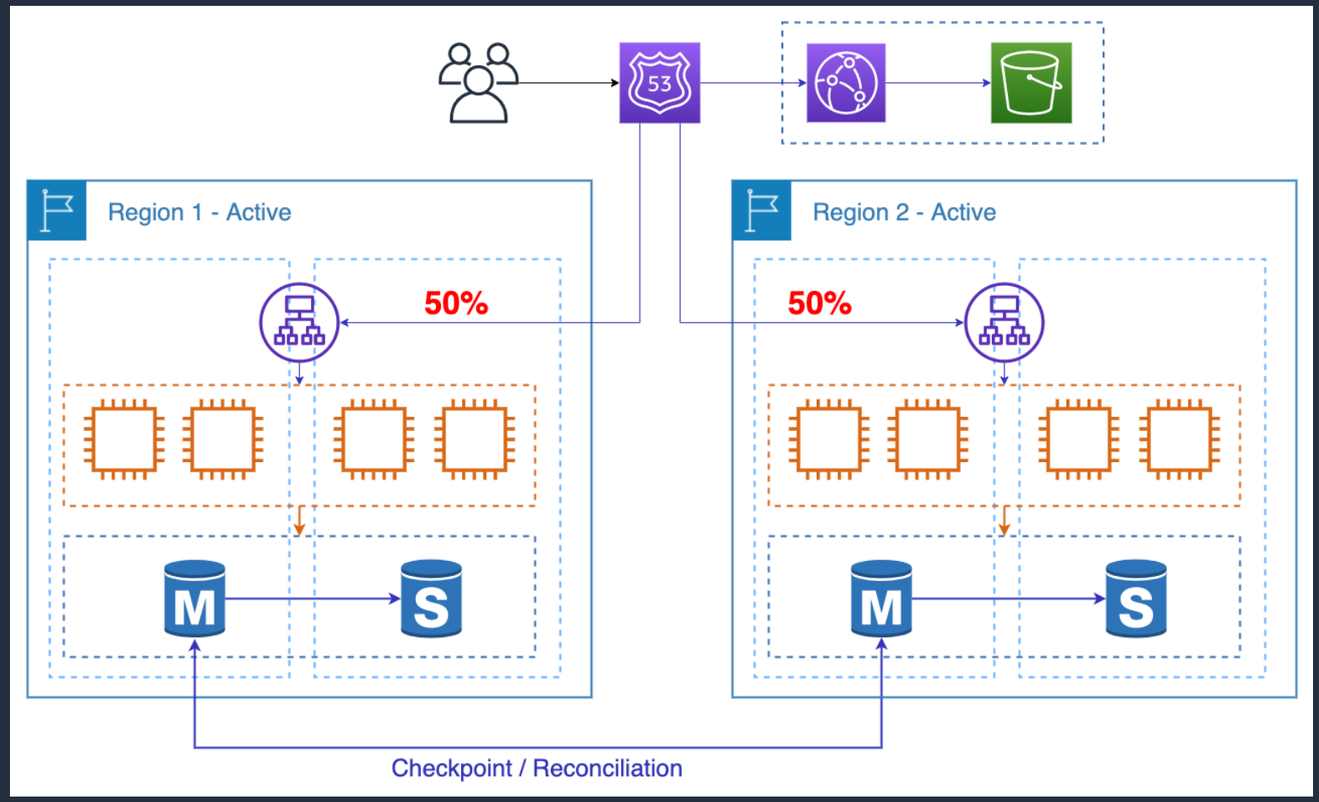
Good for legacy applications

Pros

- Zero RPO
- Zero RTO
- Little/No change to apps in each region

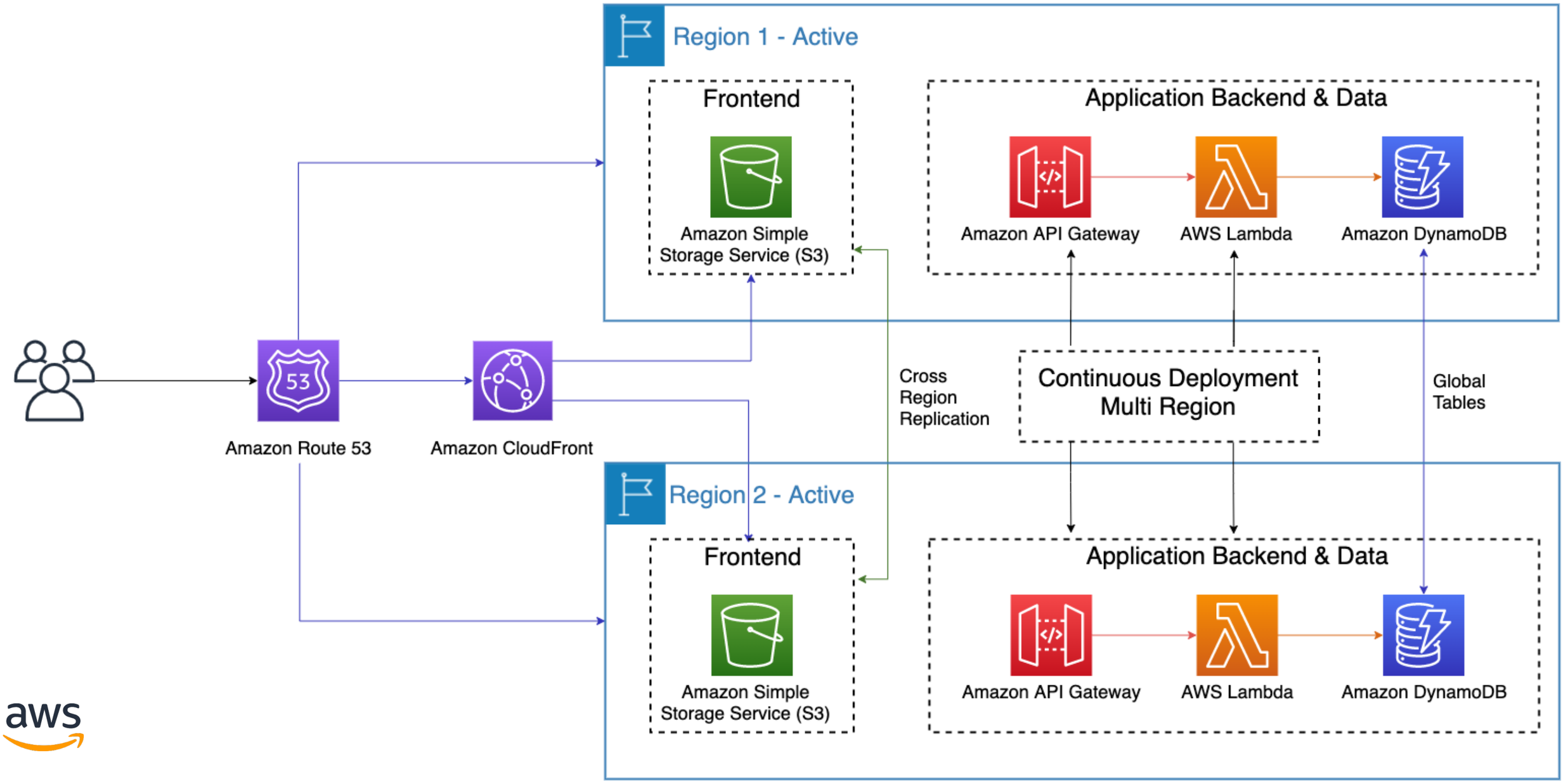
Cons

- Requires checkpointing
- Reconciliation jobs to ensure sites in sync

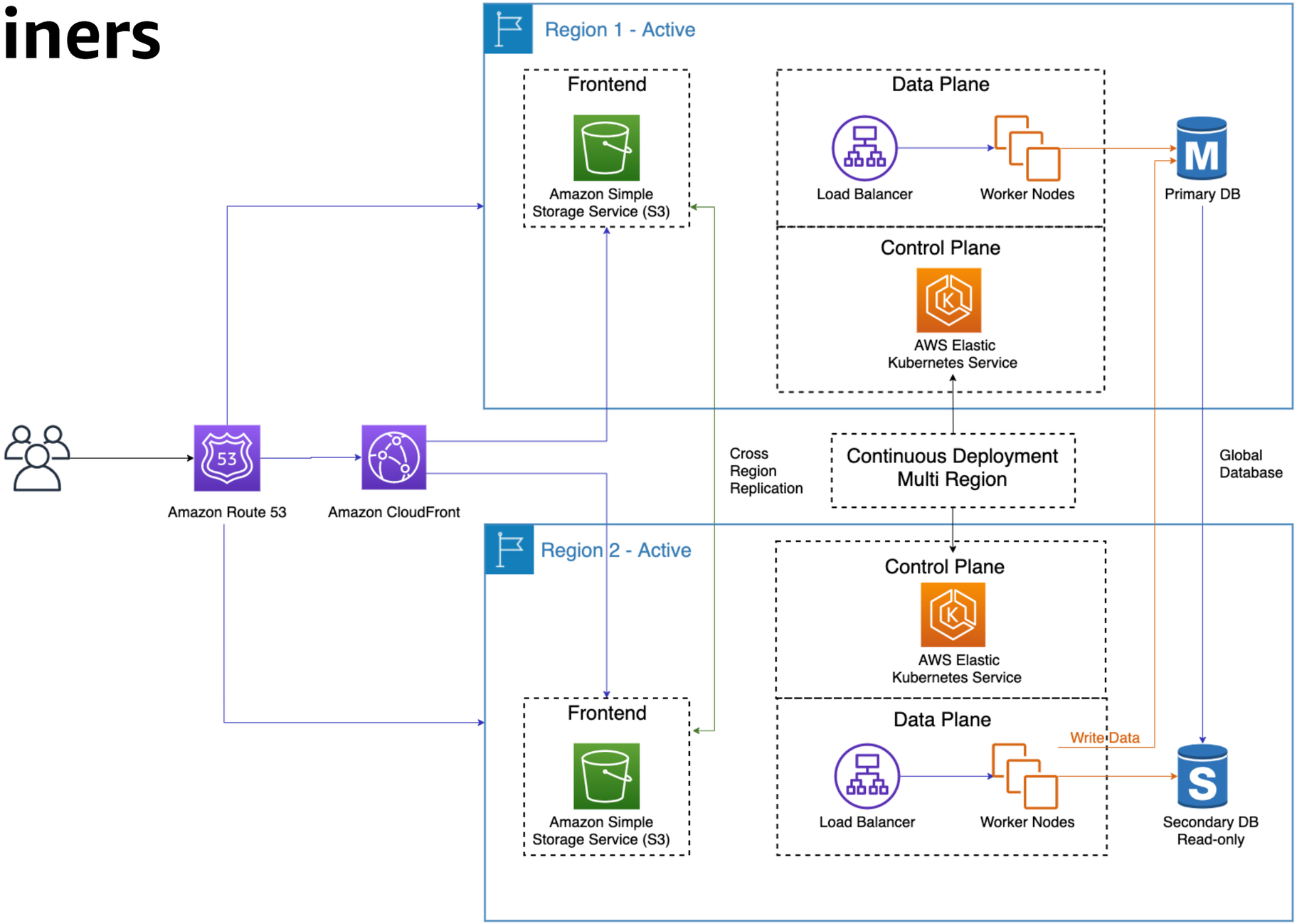


aws

Serverless


Region 1 - Active
Frontend
Amazon Simple Storage Service (S3)
Application Backend & Data
Amazon API Gateway
AWS Lambda
Amazon DynamoDB
Amazon Route 53
Amazon CloudFront
Cross Region Replication
Continuous Deployment Multi Region
Global Tables
Region 2 - Active
Frontend
Amazon Simple Storage Service (S3)
Application Backend & Data
Amazon API Gateway
AWS Lambda
Amazon DynamoDB

Containers


Region 1 - Active
Frontend
Amazon Simple Storage Service (S3)
Data Plane
Load Balancer
Worker Nodes
Primary DB
Control Plane
AWS Elastic Kubernetes Service
Amazon Route 53
Amazon CloudFront
Cross Region Replication
Continuous Deployment Multi Region
Global Database
Region 2 - Active
Control Plane
AWS Elastic Kubernetes Service
Frontend
Amazon Simple Storage Service (S3)
Data Plane
Write Data
Load Balancer
Worker Nodes
Secondary DB Read-only

Anti-Patterns

- Replicate existing problems & patterns to the cloud
- Use of Non-redundant architectures to meet schedules
- Single datacenter (Availability Zones) architectures
- Reusing manual processes
  - Data retention practices, Failover & Scaling
  - Responding to monitoring alerts and metrics (vs self-healing, auto scaling)
- Assuming data is safe in your data center

Don't sacrifice long-term value for short-term results

aws

Continuous Testing of Infrastructure

Regularly execute tests in stable, production & production-like test environments.

- Load Testing

Treat Infrastructure as Code

- CI/CD Test in Infrastructure Build Pipeline
- Testing of infrastructure during Integration Test
- Zero Touch

Monitoring

Chaos Engineering

- *“Breaking things to make them better”*

aws

Chaos engineering

Cloud has ushered in new method of testing

Principles of Chaos Engineering – “Chaos Engineering can be thought of as the facilitation of experiments to uncover systemic weaknesses.” https://principlesofchaos.org/

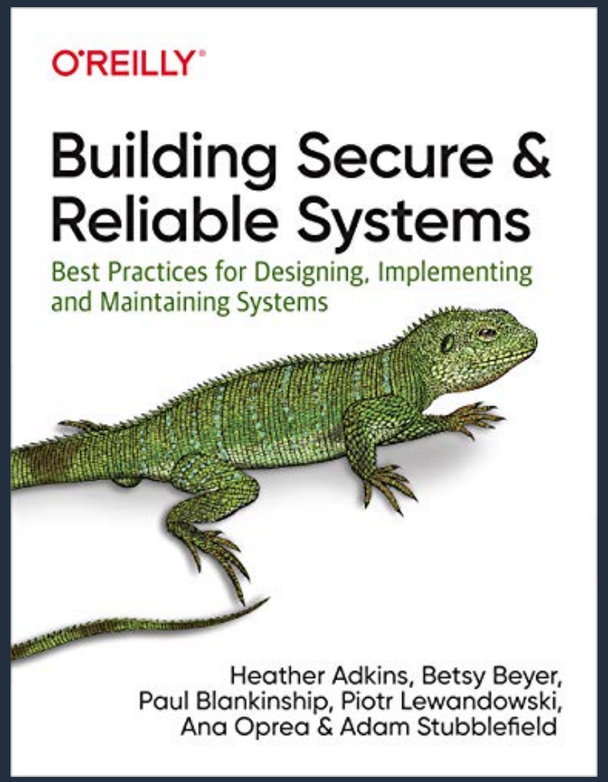
Principles

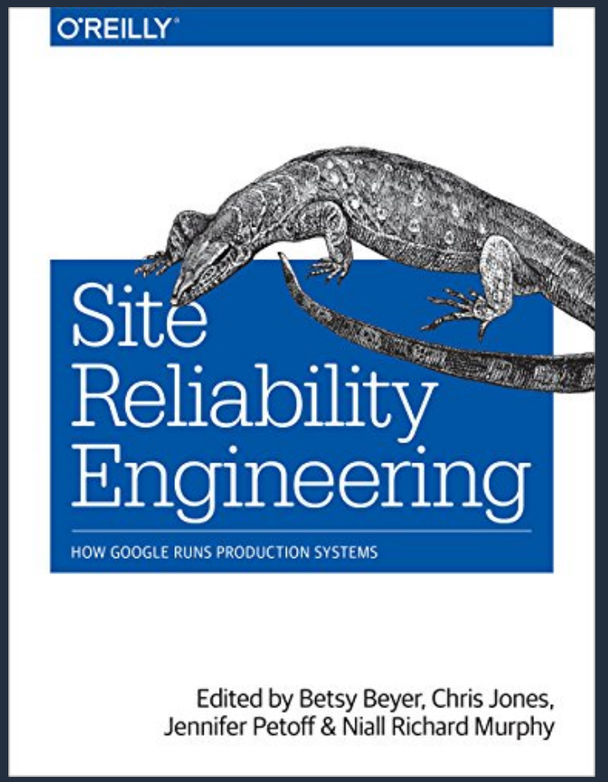
- Building a hypothesis around steady state behavior
- Applying variations to simulate real world events
- Run experiments in production
- Automate the experiments to run continuously
- Minimize blast radius of failures

aws


We are here!
Cascading failures
Backpressure and Exponential Backoff
Timeouts and Circuit Breaker
Shared nothing and Cell-Architecture
Partitions and Bulkheads
Self provisioning and Fast replacement
Fitness Functions & SLA’s
Crisis Response and Post mortem
Automatic Responses
Quarantine & Debugging

Some books…





aws

Thank You!

Luiz Yanai, Solutions Architect - AWS
Leonardo Piedade, Solutions Architect - AWS