AWS re:Invent

ARC335-R

Designing for failure: Architecting resilient systems on AWS

Adrian Cockcroft
VP, Cloud Architecture
Amazon Web Services

Harsha Nippani
Solutions Architect
Amazon Web Services

Vinay Kola
Software Engineer
Snap Inc.

AWS re:Invent

aws

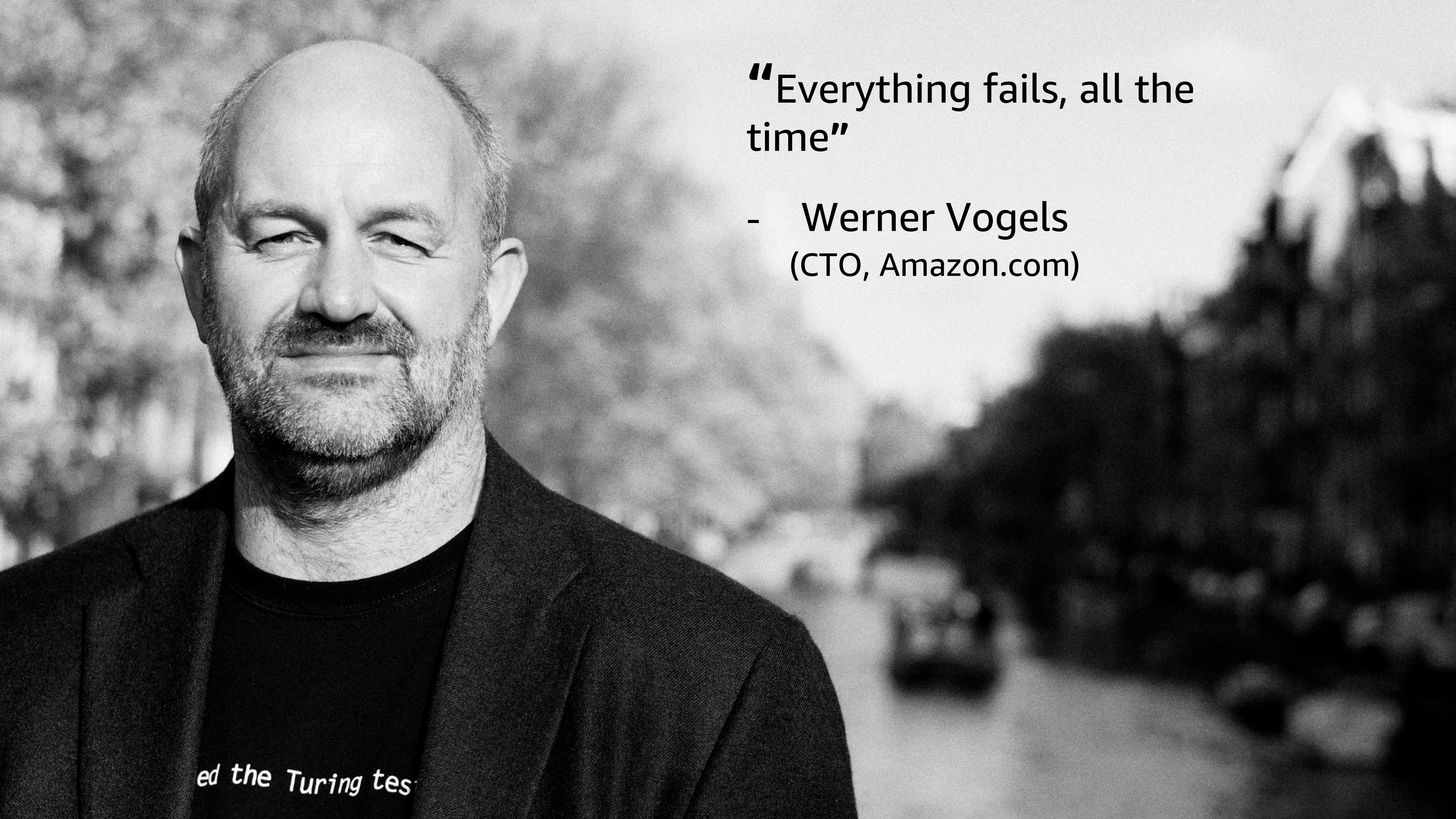
Agenda

- Risk and resilience
- Technical considerations
- Customer use case: Snap
- Continuous resilience
- Related sessions
- AWS whitepaper

Risk and resilience

AWS re:Invent

aws



"Everything fails, all the time"

- Werner Vogels
  (CTO, Amazon.com)

Business continuity

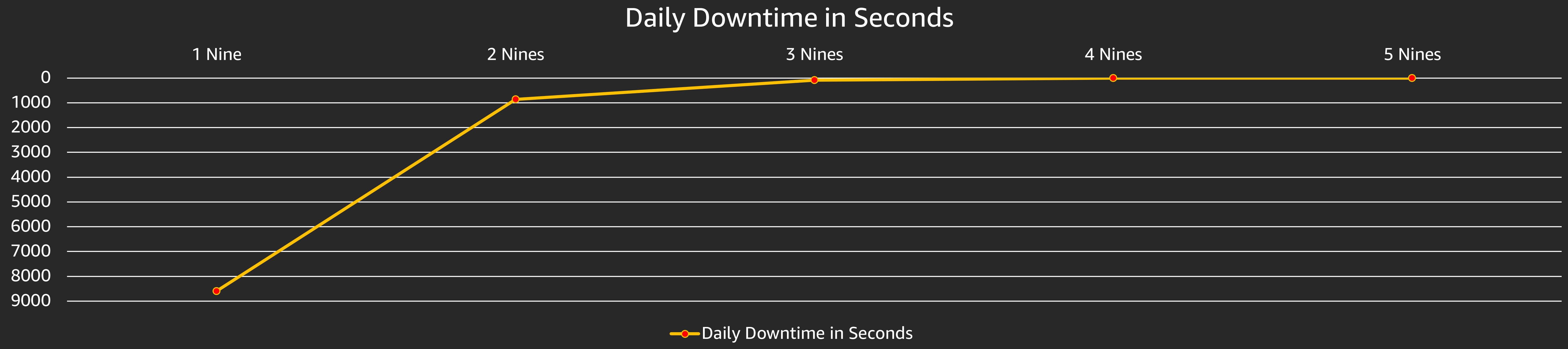
How much data can you afford to recreate or lose?

How quickly must you recover? What is the cost of downtime?

It's *not* about the data, it's about the *mission*

Availability by the numbers

Level of availability	Percent uptime	Downtime per year	Downtime per day
1 Nine	90%	36.5 Days	2.4 Hours
2 Nines	99%	3.65 Days	14 Minutes
3 Nines	99.9%	8.76 Hours	86 Seconds
4 Nines	99.99%	52.6 Minutes	8.6 Seconds
5 Nines	99.999%	5.26 Minutes	0.86 Seconds



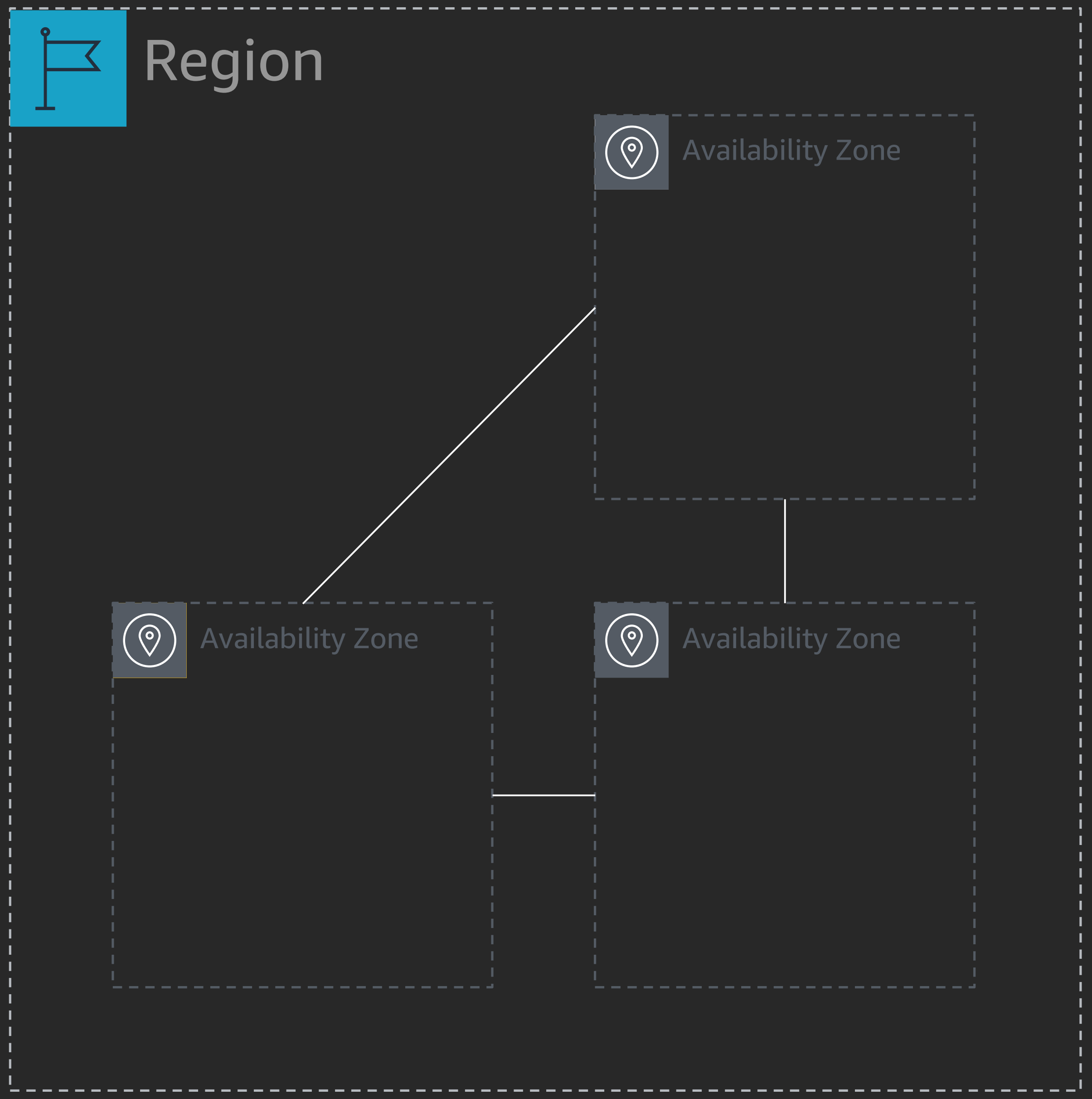
Multi-AZ architecture

- Enables fault-tolerant applications
- AWS Regional services designed to withstand AZ failures
- Leveraged in the Amazon S3 design for 99.999999999% durability

Multi-AZ → Zero blast radius!

Well-Architected Framework

[AWS Shared Responsibility Model]



Resilient AWS infrastructure

Infrastructure

Regions, Availability Zones, Networking

Service design

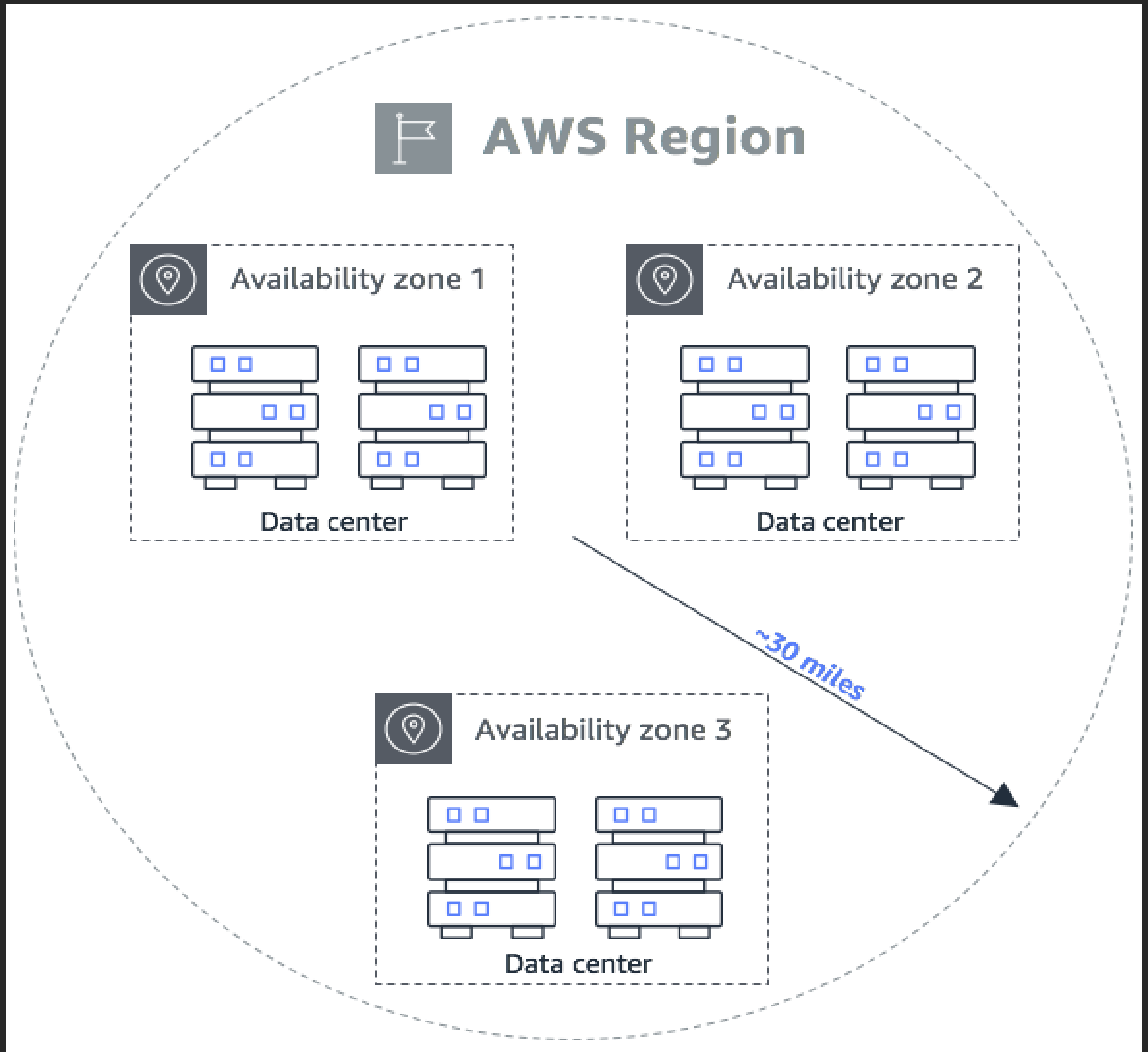
Fault Isolation zones

- Cell-based architecture
- Multi-AZ architecture

Microservice architecture

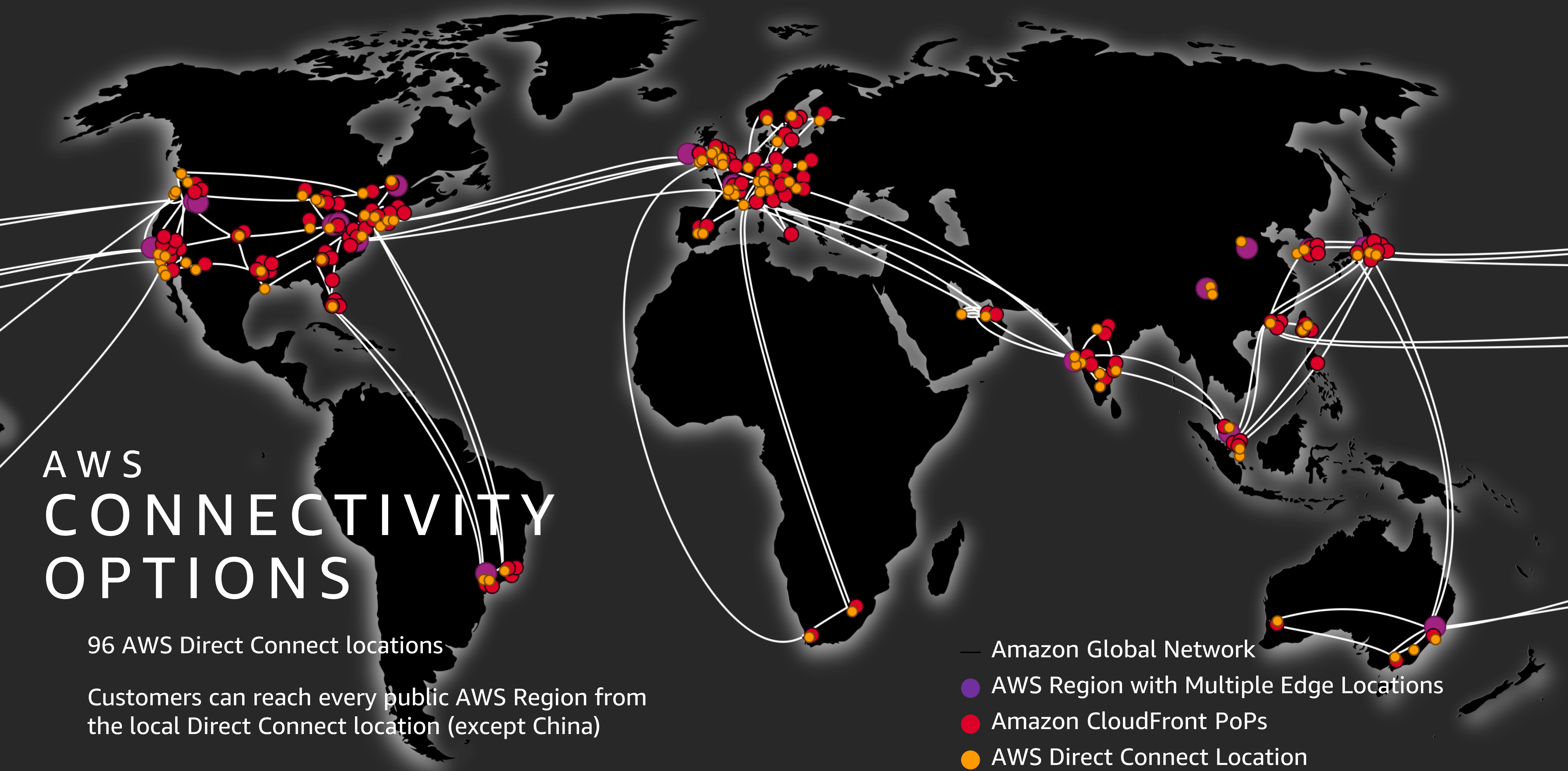
Distributed systems best practices

- Throttling
- Retry with exponential back off
- Circuit breaker



AWS Services scope:
Single-AZ, Regional, Global, Cross-Regional capability

AWS CONNECTIVITY OPTIONS

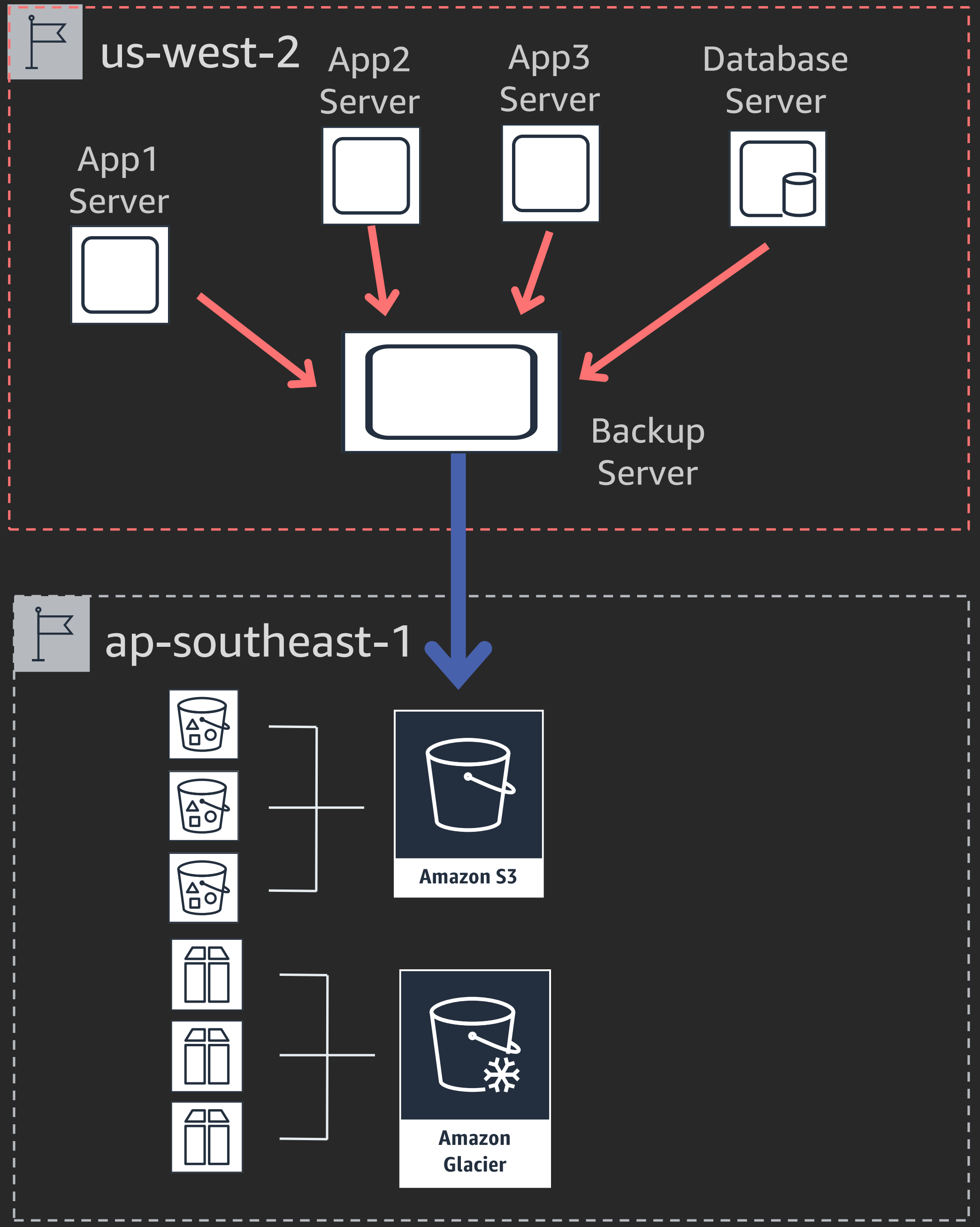


96 AWS Direct Connect locations

Customers can reach every public AWS Region from the local Direct Connect location (except China)

The four strategies for business continuity (multi-region)

Strategy: Backup & restore (multi-region)

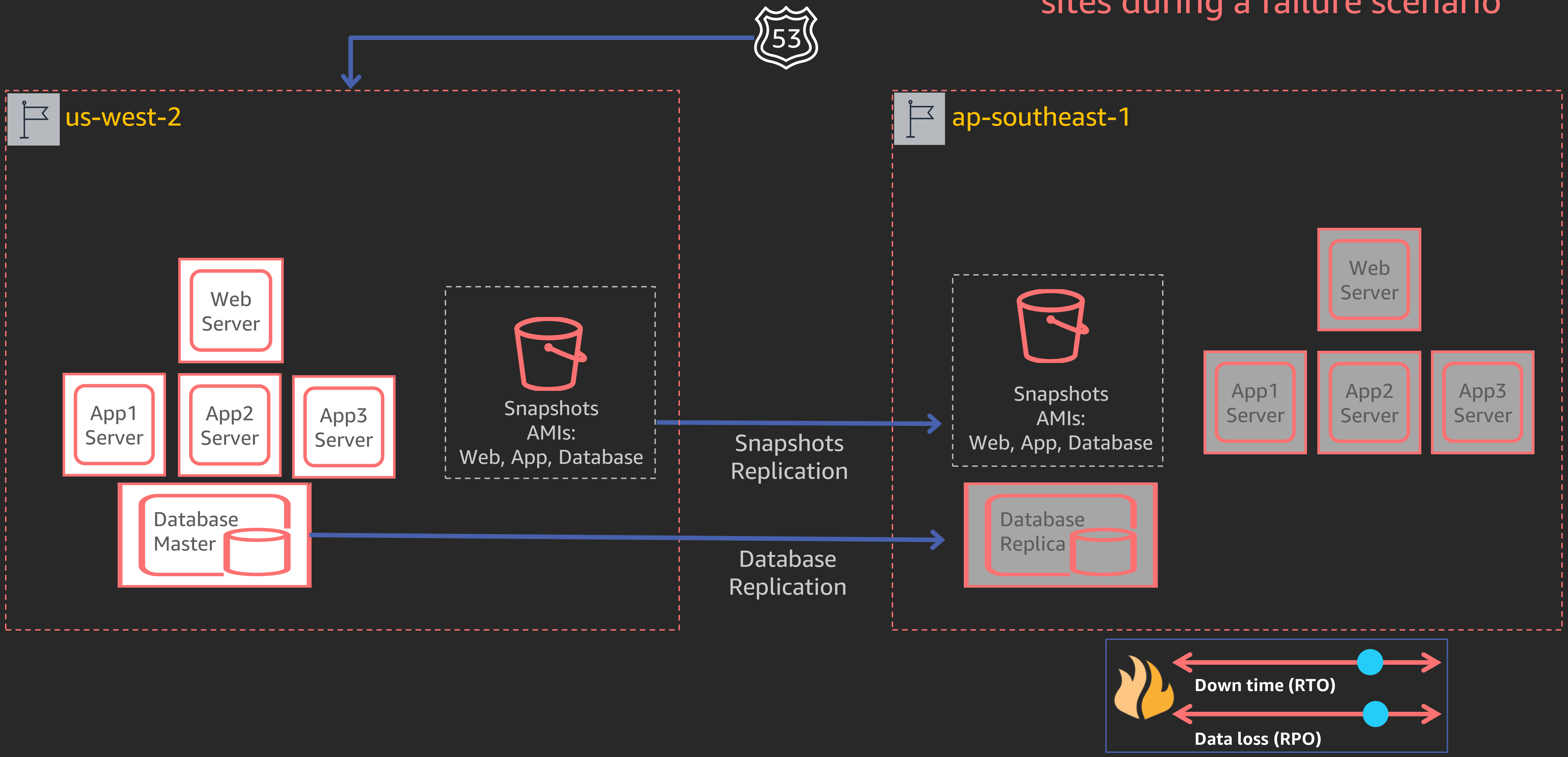


Back up to another Region

- Use managed database services with Amazon S3 (Amazon S3) or Amazon S3 Glacier
- Data stored with high durability in multiple locations

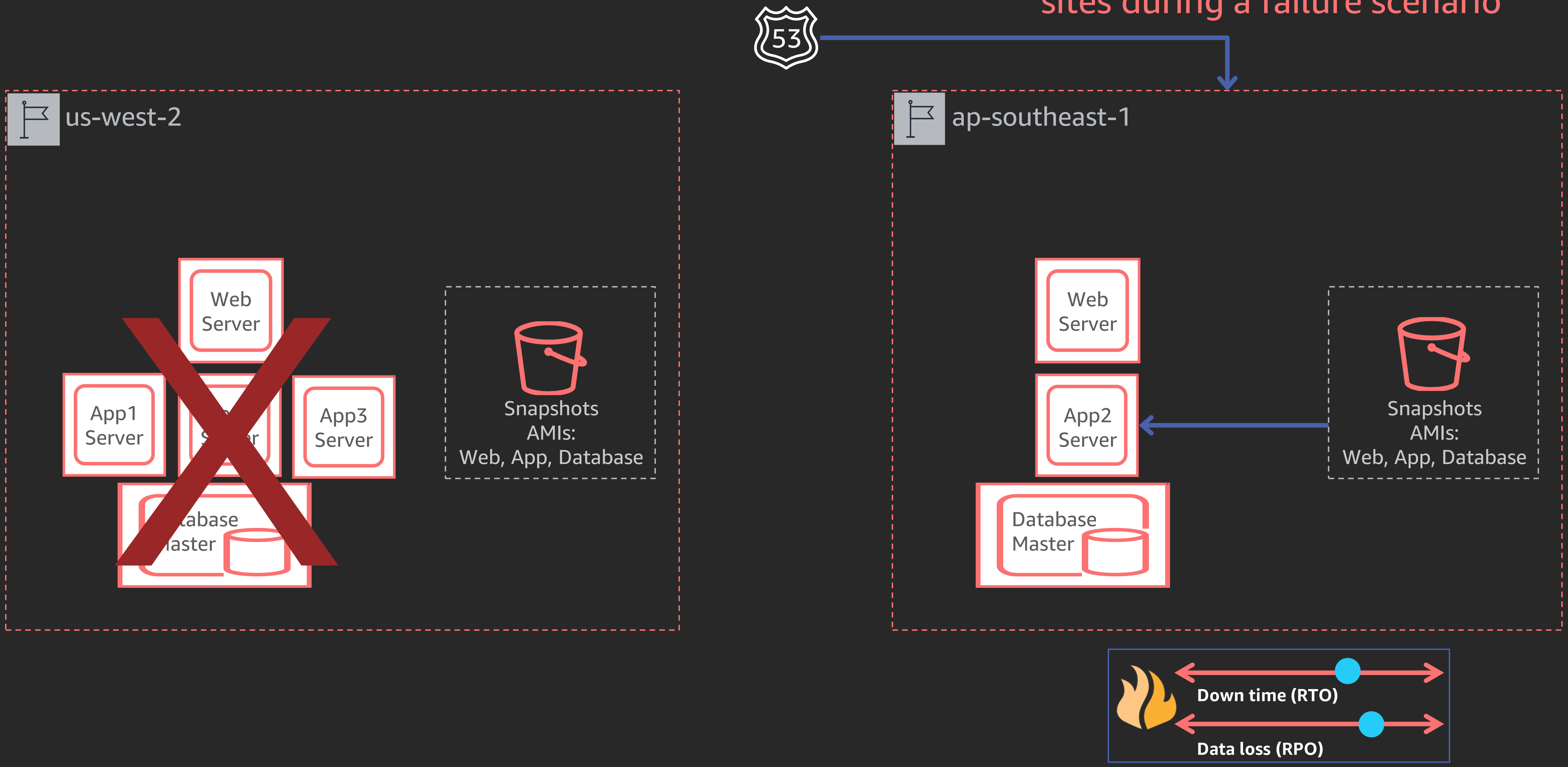
Strategy: Pilot light (multi-region)

Allows the scaling of redundant sites during a failure scenario

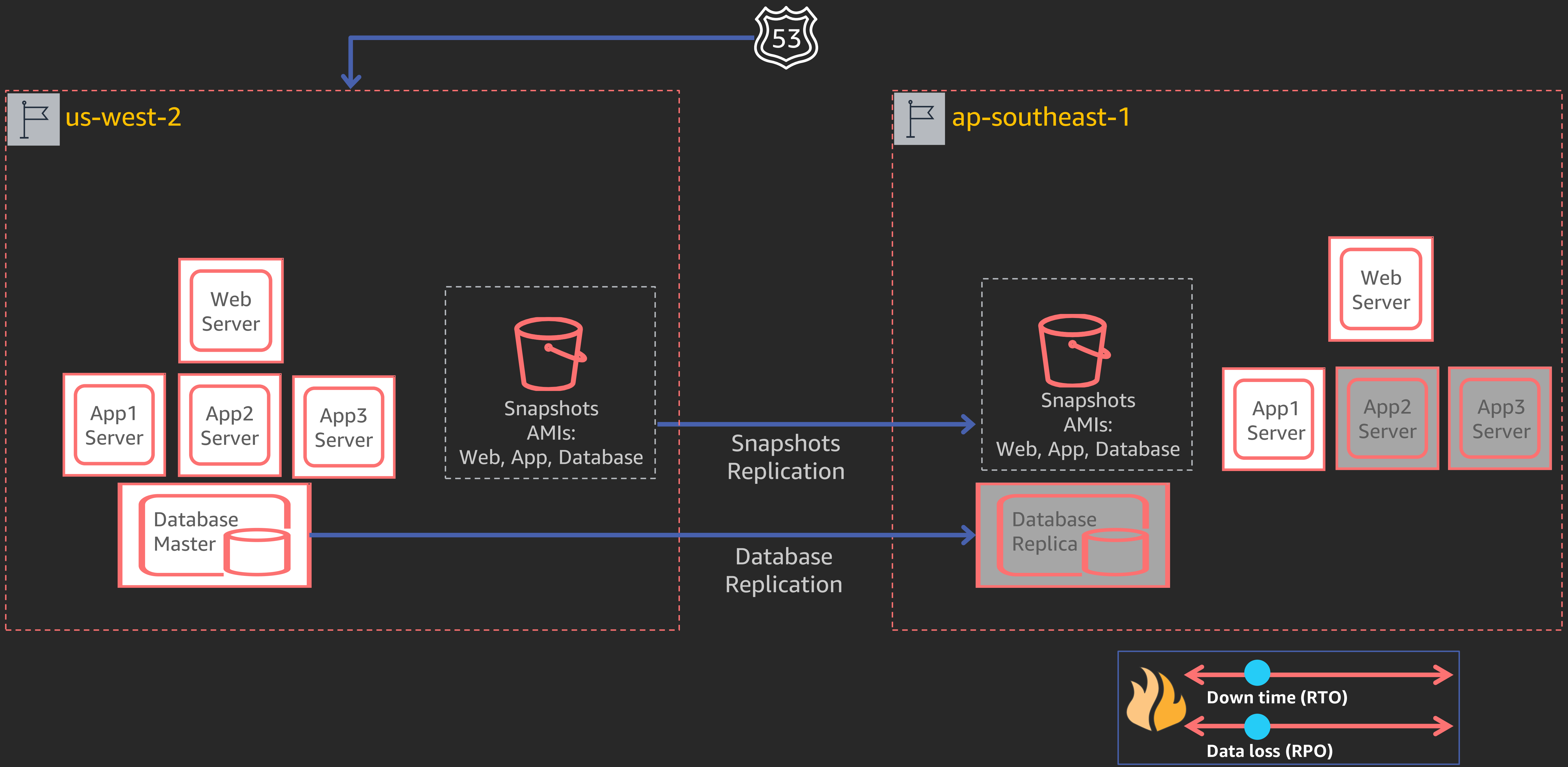


Strategy: Pilot light (multi-region)

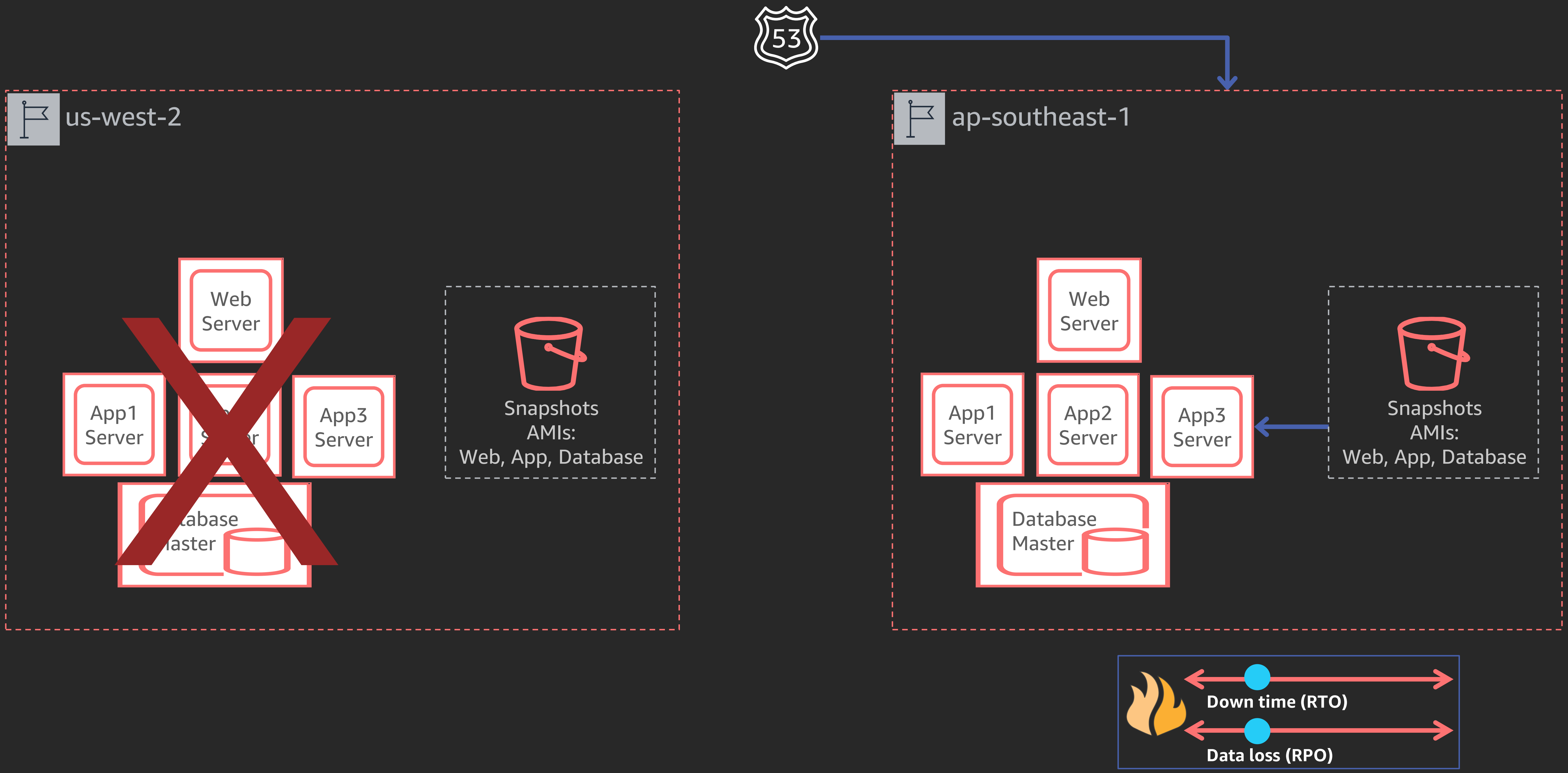
Allows the scaling of redundant sites during a failure scenario



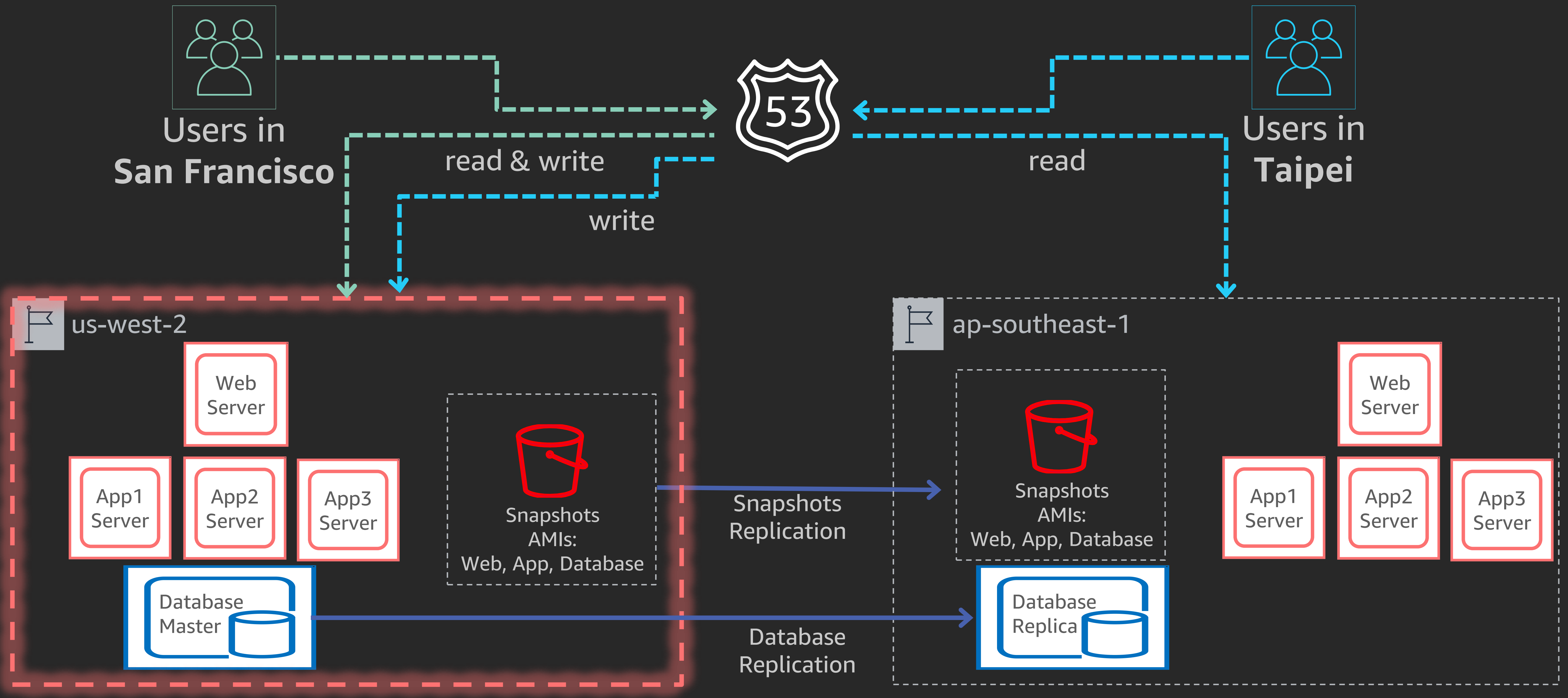
Strategy: Warm standby (multi-region)



Strategy: Warm standby (multi-region)



Strategy: Active-Active (multi-region)



Technical considerations (data and network)

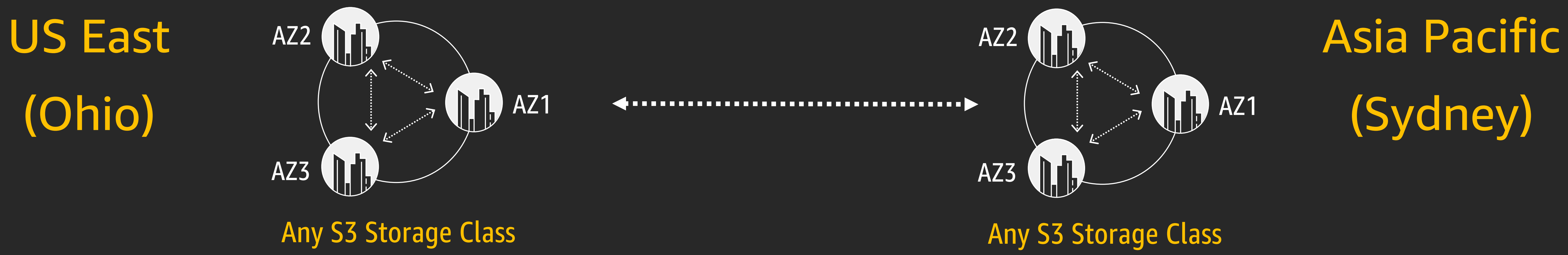
AWS re:Invent

aws

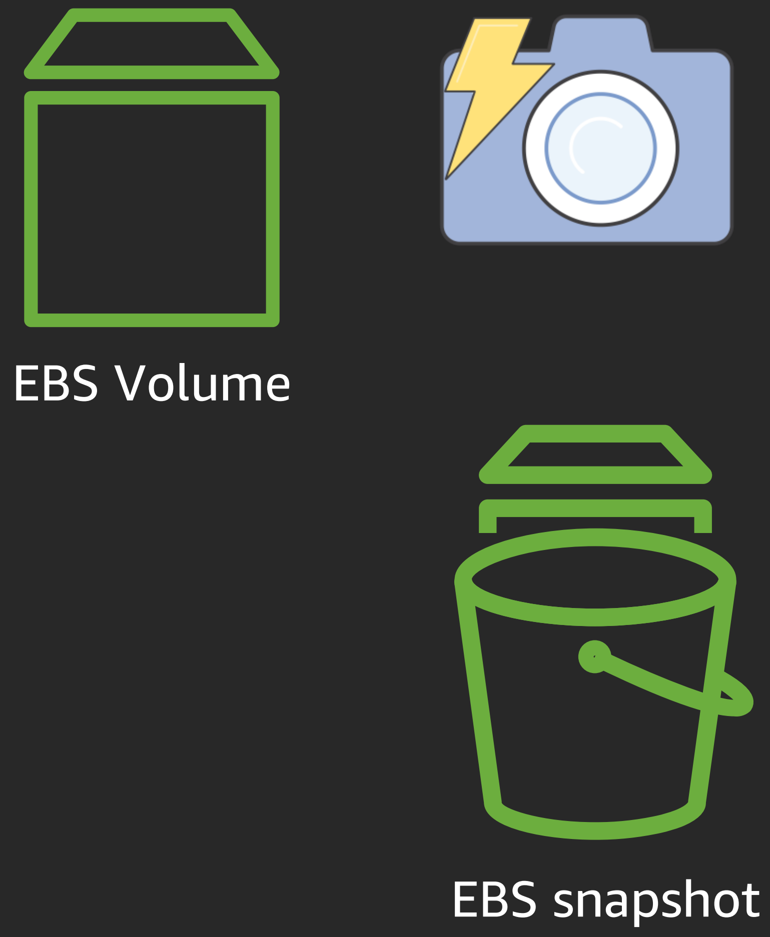
Amazon S3 - Cross-Region Replication

Flexibility to replicate data:

- At the bucket, prefix, or object level
- From any region to any region
- To any storage class
- Across AWS accounts (Change the object owner in the destination region)
- Amazon S3 Replication Time Control (Amazon S3 RTC) **NEW!**



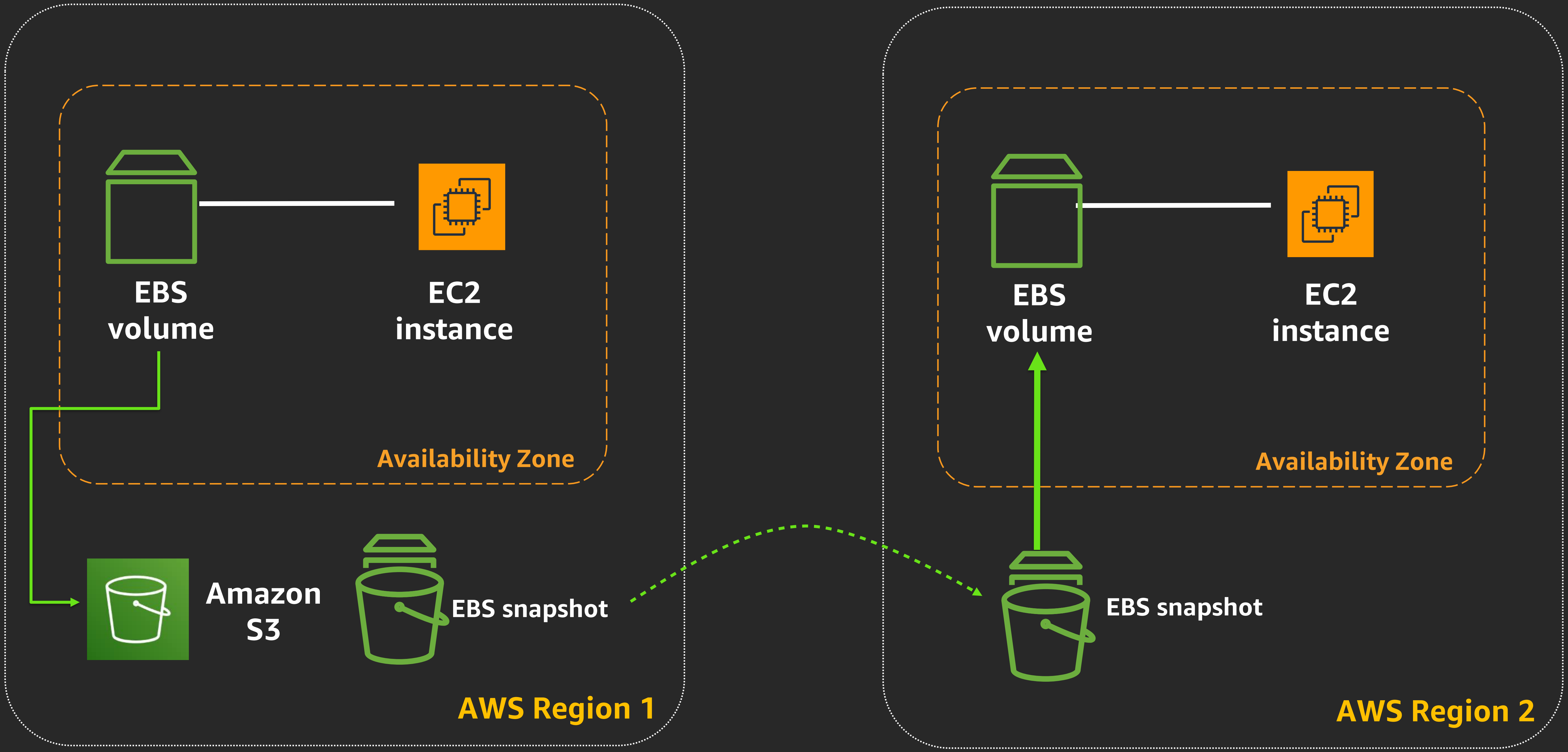
Amazon EBS snapshots



Elastic block storage

- Point-in-time backup of modified volume blocks
- Stored in Amazon S3, accessed via Amazon EBS APIs
- Subsequent snapshots are incremental
- Deleting snapshot will only remove data exclusive to that snapshot
- Copies in same region or cross-region

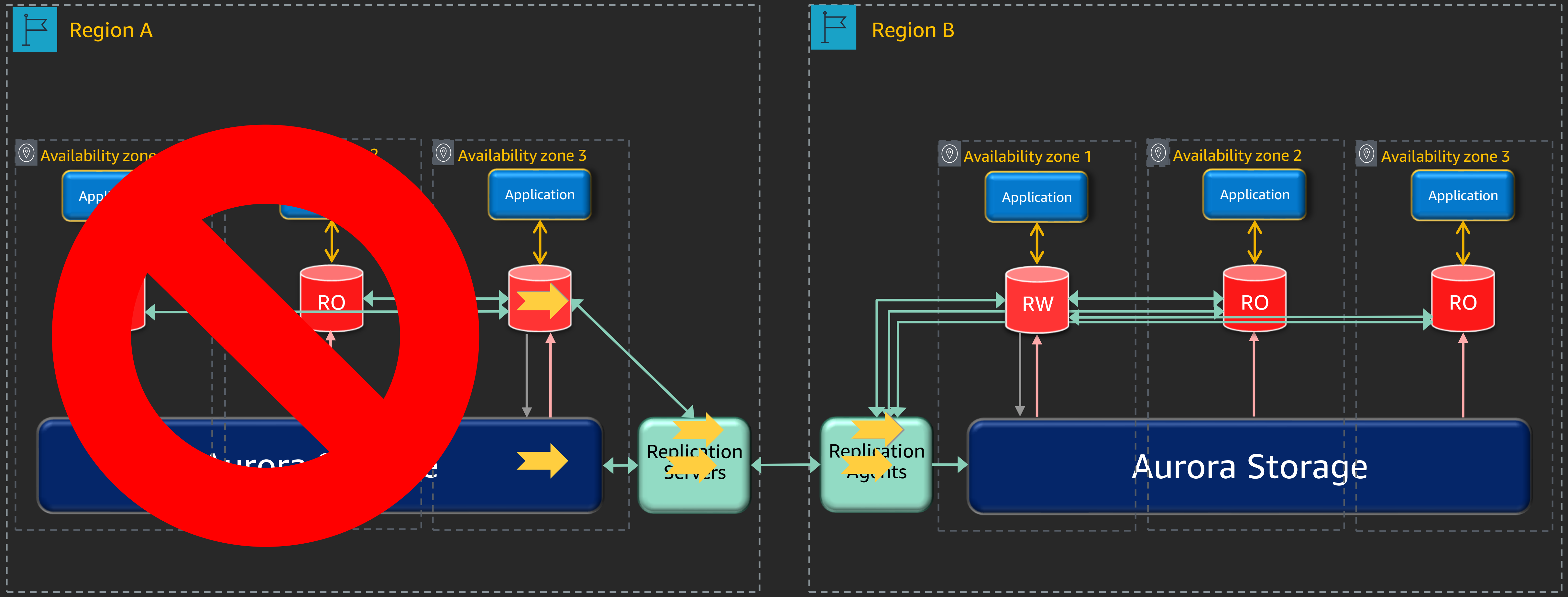
EBS volume – Cross-region snapshot copy



Amazon DynamoDB global tables
Fully managed, multi-master, multi-region database

- Build high performance, globally distributed applications
- Low latency reads & writes to locally available tables
- Disaster proof with multi-region redundancy
- Easy to setup and no application re-writes required

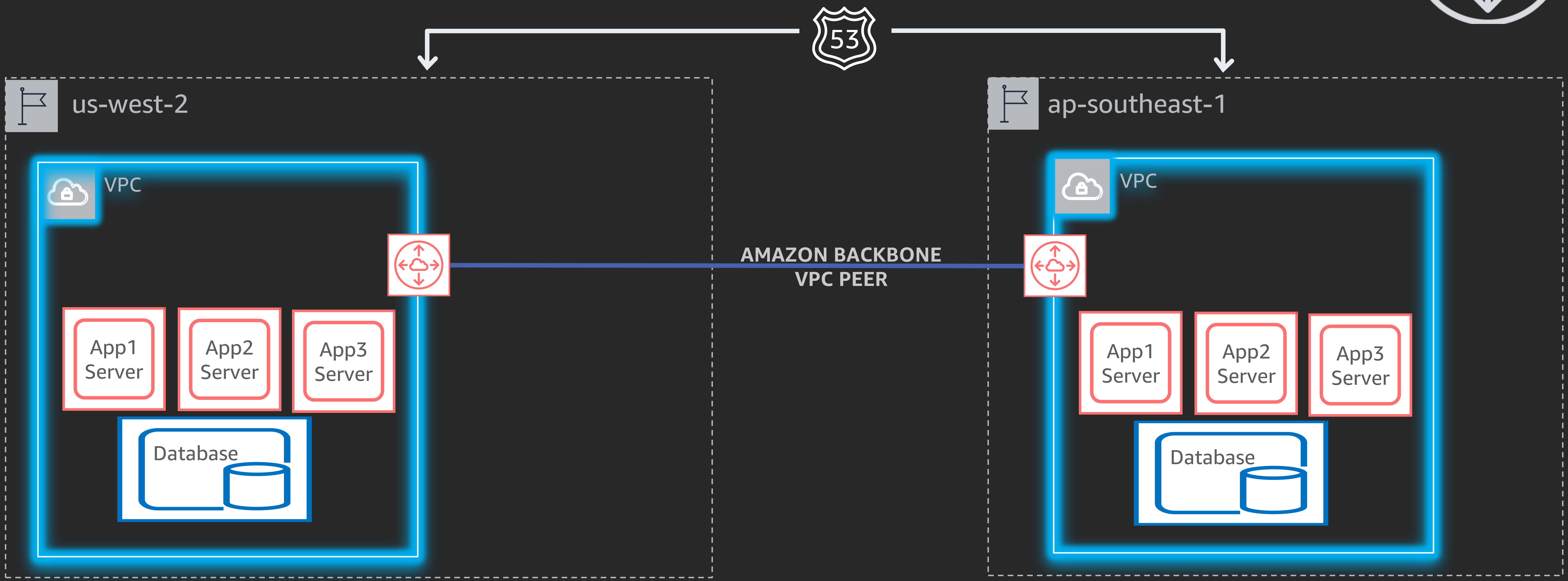
Cross-region read replicas with Amazon RDS and Amazon Aurora



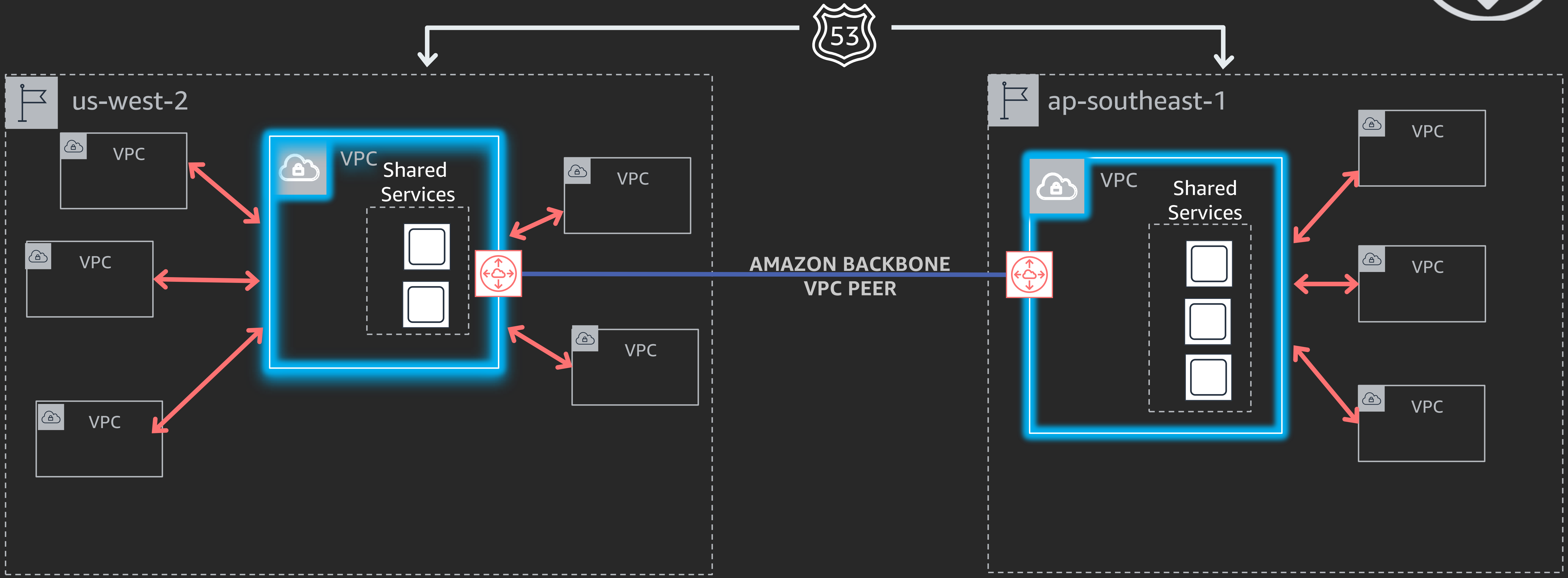
Inter-Region Virtual Private Cloud (VPC) Peering

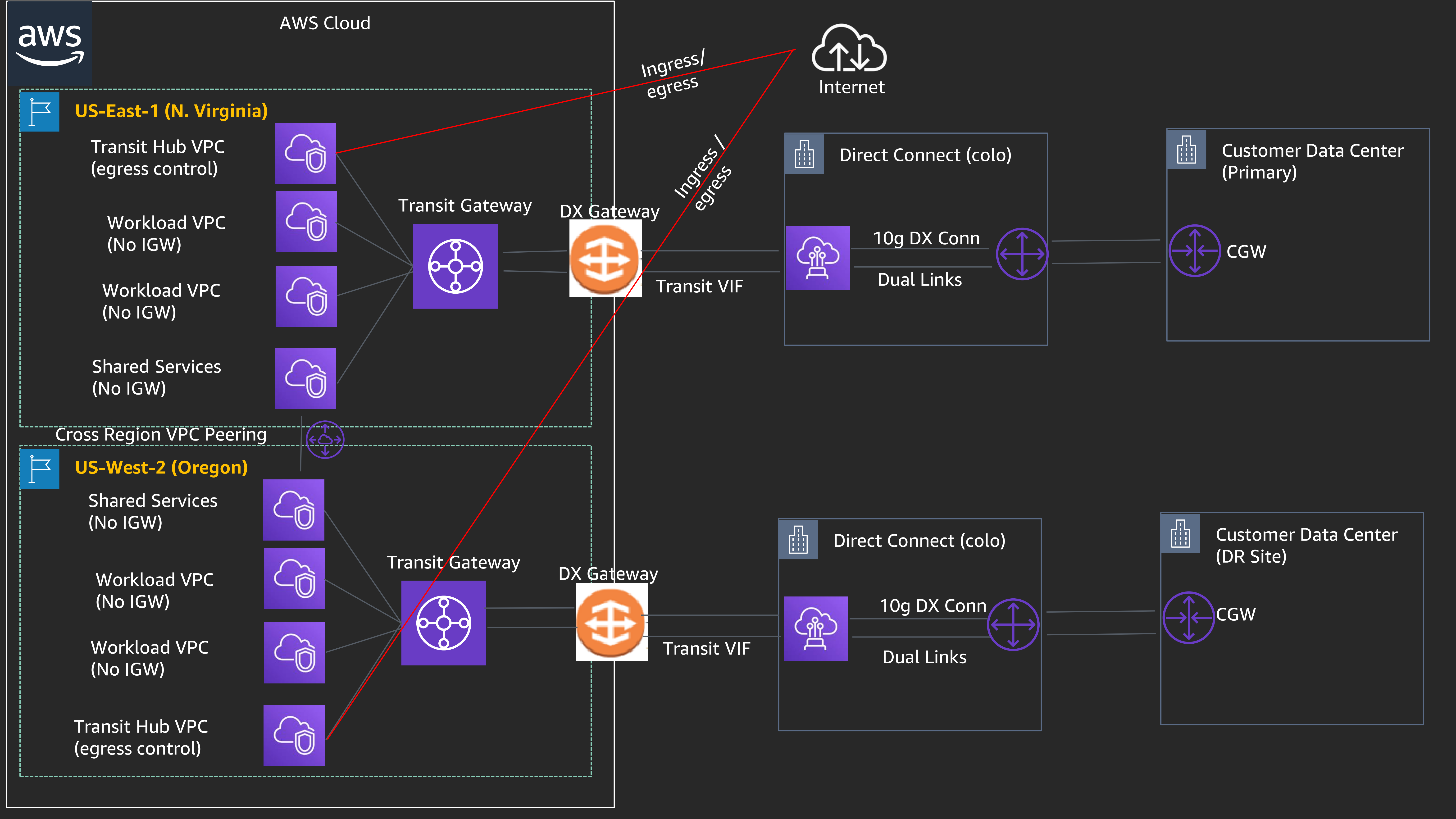
Why is this important to my architecture?

Inter-region VPC peering



Multi-region multi-VPC connectivity



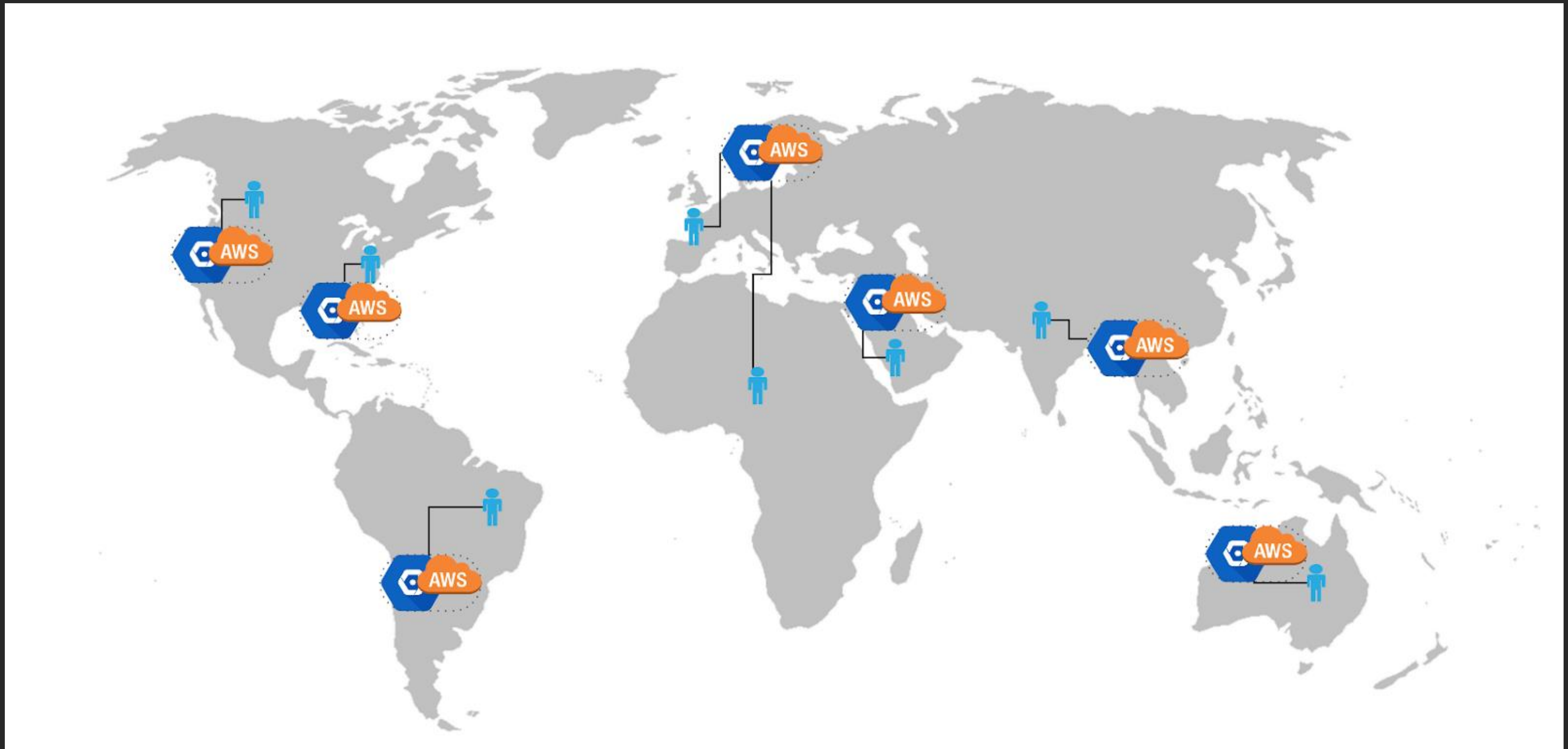

AWS Cloud
US-East-1 (N. Virginia)
Transit Hub VPC
(egress control)
Workload VPC
(No IGW)
Workload VPC
(No IGW)
Shared Services
(No IGW)
Transit Gateway
DX Gateway
Ingress/
egress
Ingress /
egress
Internet
Direct Connect (colo)
10g DX Conn
Dual Links
Transit VIF
Customer Data Center
(Primary)
CGW
Cross Region VPC Peering
US-West-2 (Oregon)
Shared Services
(No IGW)
Workload VPC
(No IGW)
Workload VPC
(No IGW)
Transit Hub VPC
(egress control)
Transit Gateway
DX Gateway
Transit VIF
Direct Connect (colo)
10g DX Conn
Dual Links
Customer Data Center
(DR Site)
CGW

Snap's User and Friend Graph Infrastructure on AWS

What is Snapchat?

Real Friends

Where are our users?



210+
million DAU

(on average)

3.5+ billion
snaps / day

Snap Inc. internal data 2019. See Snap Inc. public filings with the SEC. Snap Inc. internal data Q2 2019.

Why bother with operational resilience?

“Fastest way to share a moment.”

Availability tiers

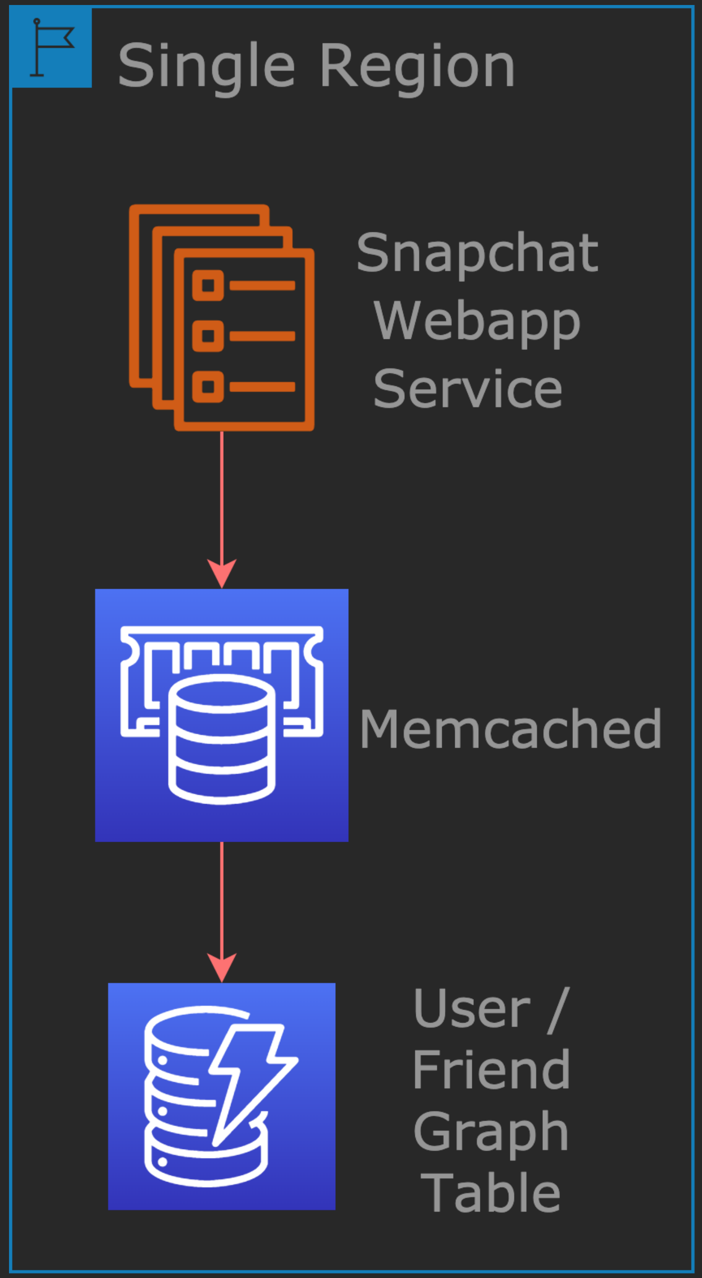
- Tier-0: 99.99% availability (eg: Service Mesh)
- Tier-1: 99.95% availability (eg: Messaging, User/Friend Service)
- Tier-2: 99% availability (eg: Stickers)
- Tier-3: 95% availability (eg: Internal Tools)

Case Study: User & Friend Service

- Data
  - User Profile data (user_id, username, display name, etc)
  - Who their friends are, and corresponding privacy settings (can they send me snaps? can they view my stories?)
  - Tier-1 Service
- Access patterns
  - Online (latency-sensitive, strong vs eventually consistent reads)
  - Offline analytics
  - Near real-time event streams

Legacy architecture

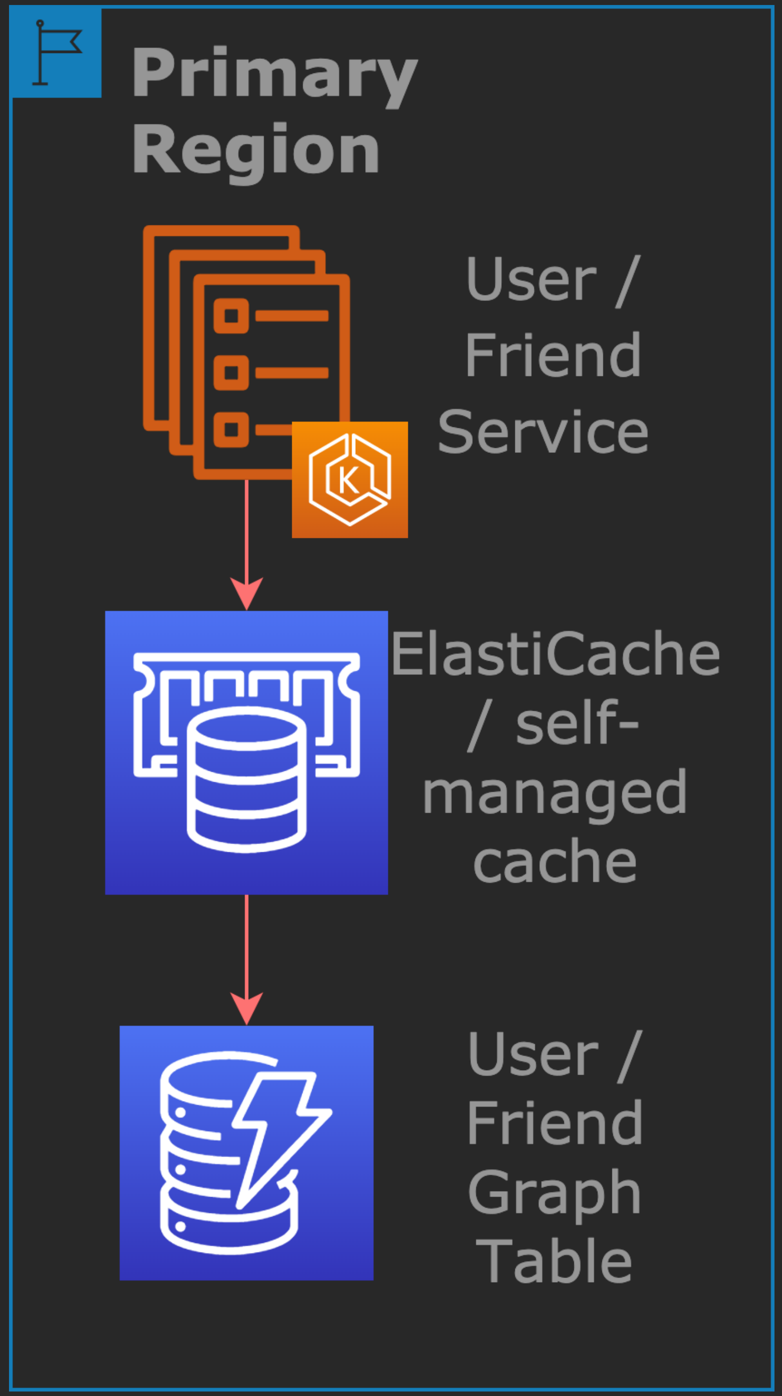
- Monolithic service (large blast radius)
- Single region (performance and availability issues)
- Direct DB access instead of service APIs
  - Unnecessary contention on writes
  - Data corruption
  - Difficult to evolve the data model



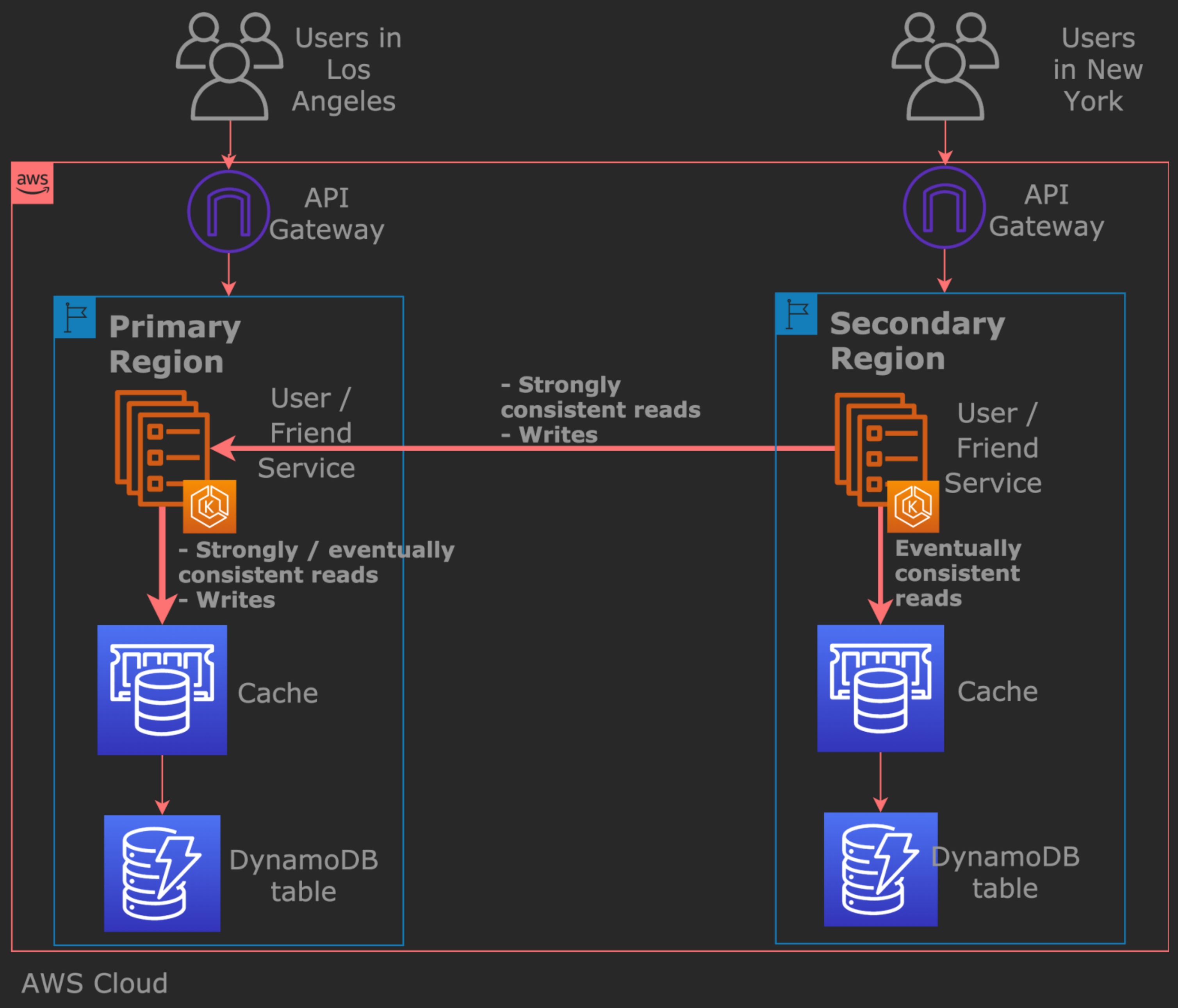
Current architecture

- Service-oriented architecture
- Multi-region active-active
  - Options:
    - **Read local, write global**
    - Read local, write partitioned
    - Read local, write local

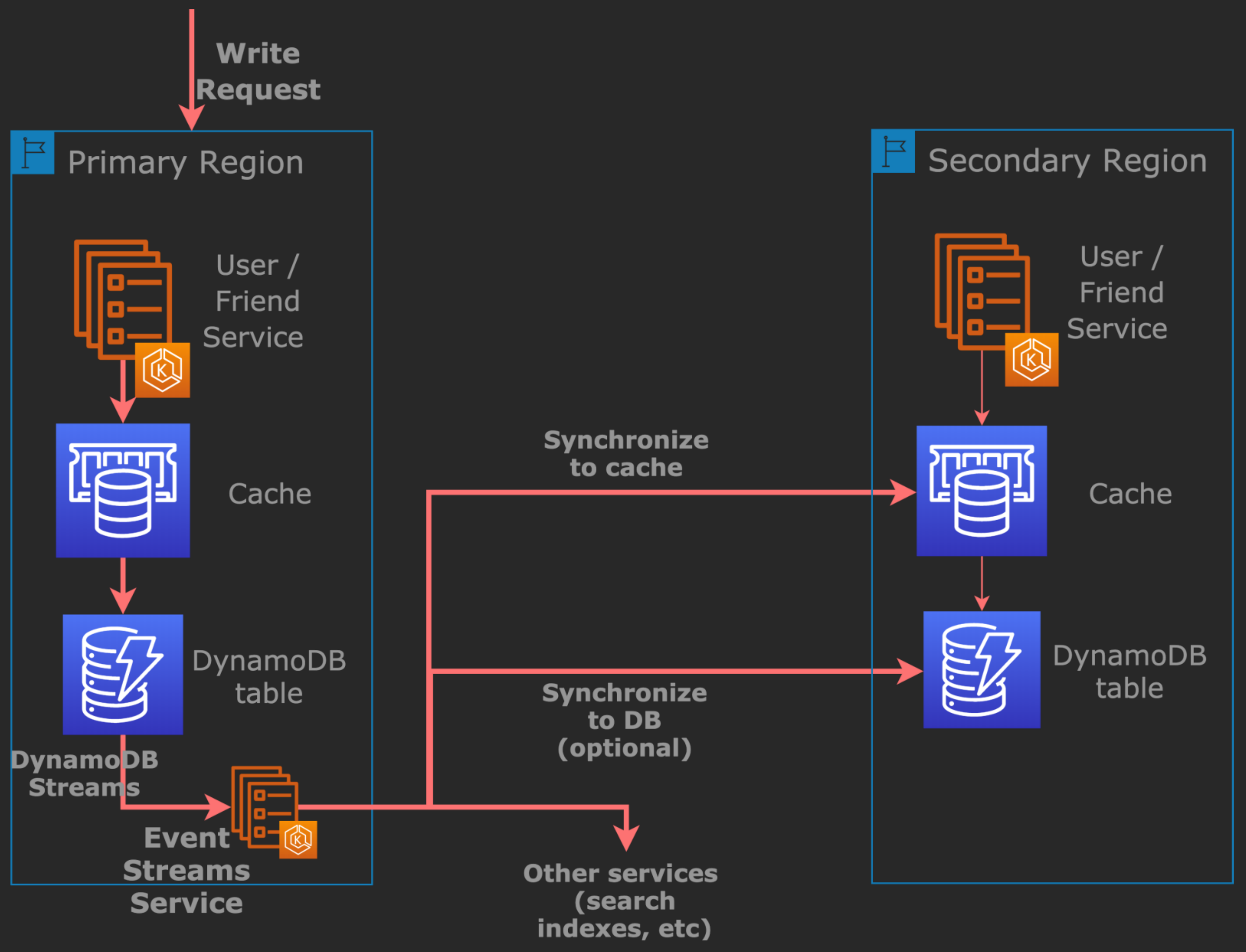
Service architecture (primary region)



Regionalization (multi-region active-active)



Data replication

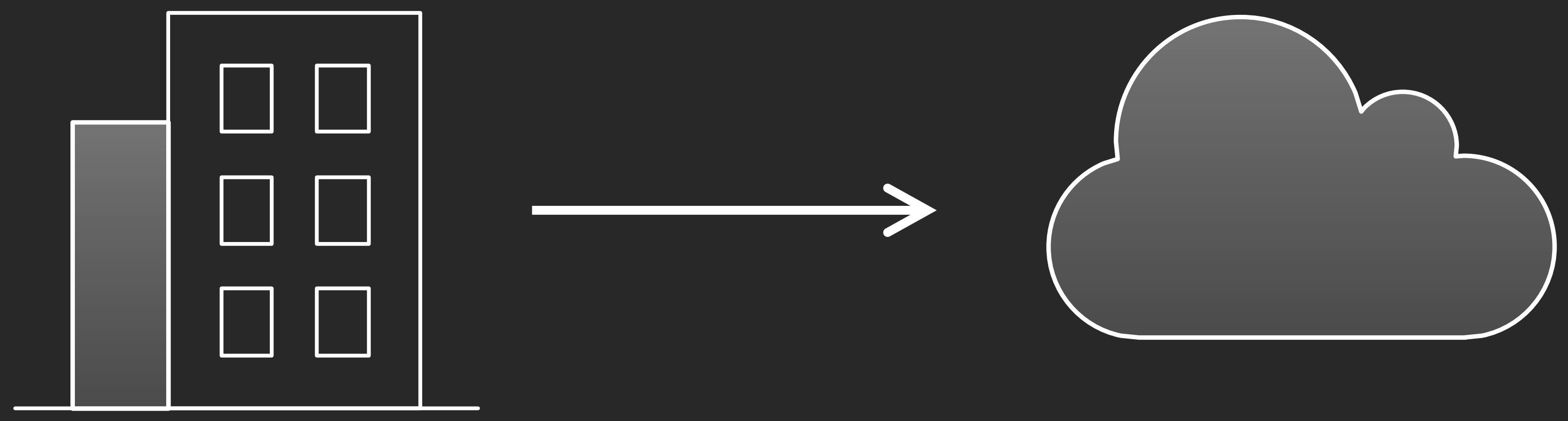


Failure modes & scaling

- Points of failure
  - Compute (Amazon EKS)
  - Cache (Amazon ElastiCache / Amazon EKS)
  - DB (Amazon DynamoDB)
- Modes of failure
  - Server is down
  - Availability Zone is down
  - Region is down

Continuous Resilience

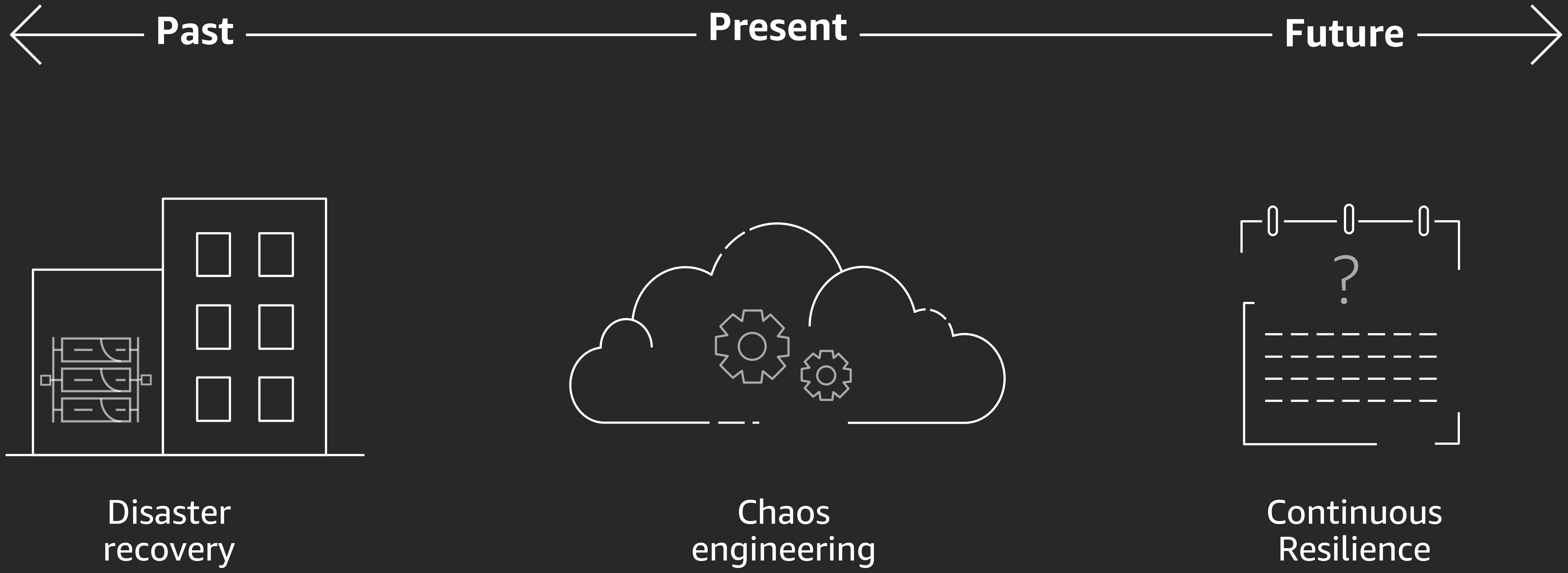
AWS re:Invent



Data center to cloud migrations are under-way for the most business and safety critical workloads

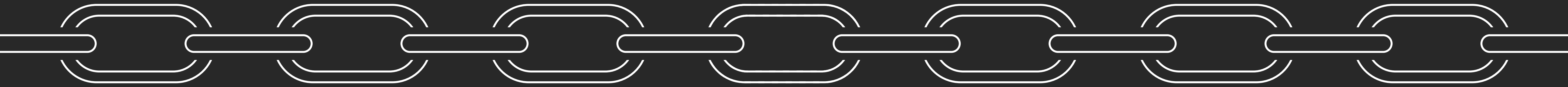
AWS and our partners are developing patterns, solutions and services for customers in all industries including travel, finance, healthcare, manufacturing…

Resilience


Past
Present
Future
Disaster recovery
Chaos engineering
Continuous Resilience

“If we change the name from chaos engineering to continuous resilience, will you let us do it all the time in production?”

Blog post: Failure Modes and Continuous Resilience - medium.com/@adrianco



You can only be as strong as your **weakest link**

Dedicated teams are needed to find weaknesses before they take you out!

Availability, safety, and security have similar characteristics

Hard to measure near misses

Hard to model complex dependencies

Catastrophic failure modes

Availability, safety, and security have similar mitigations

Layered defense in depth

Bulkheads to contain blast radius

Minimize dependencies/privilege

Availability, Safety, and security break each other

Security breaks availability

Availability breaks safety

Etc.

- Stop?
- Carry on with reduced functionality?
- Collapse horribly?

Permissive failure, what's the real cost of continuing?

See ***Memories, Guesses, and Apologies*** by Pat Helland

How often do you failover apps to it?

How often do you failover the **whole data center** at once?

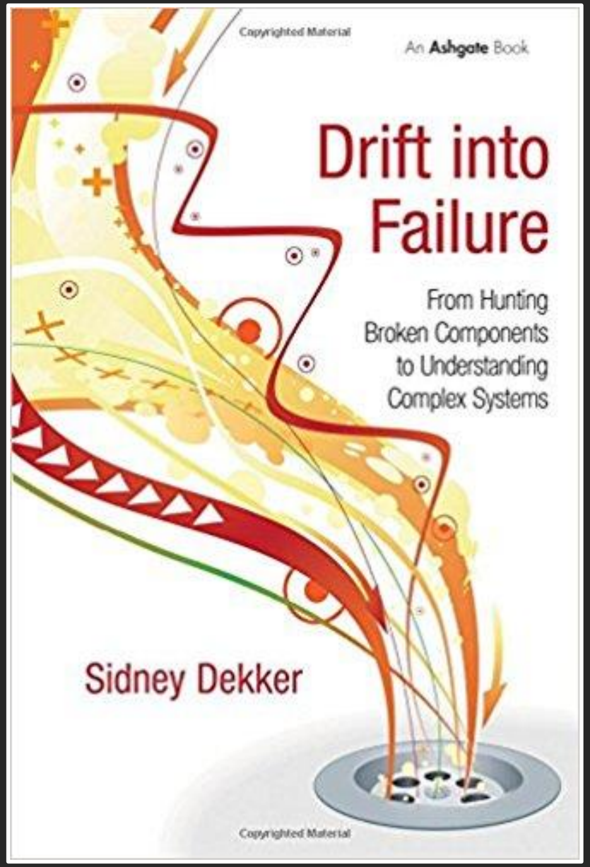
"Availability Theater"

A fairy tale…

Once upon a time, in theory, if everything works perfectly, we have a plan to survive the disasters we thought of in advance.

How did that work out?

Forgot to renew domain name…	SaaS vendor
Didn’t update security certificate and it expired…	Entertainment site
Data center flooded in hurricane Sandy…	Finance company, Jersey City
Whoops!	**YOU, tomorrow**



Drift into Failure

Sydney Dekker

Everyone can locally optimize for the right outcome at every step, and you may still get a catastrophic failure as a result…

We need to capture and learn from near misses, test and measure the safety margins, before things go wrong.

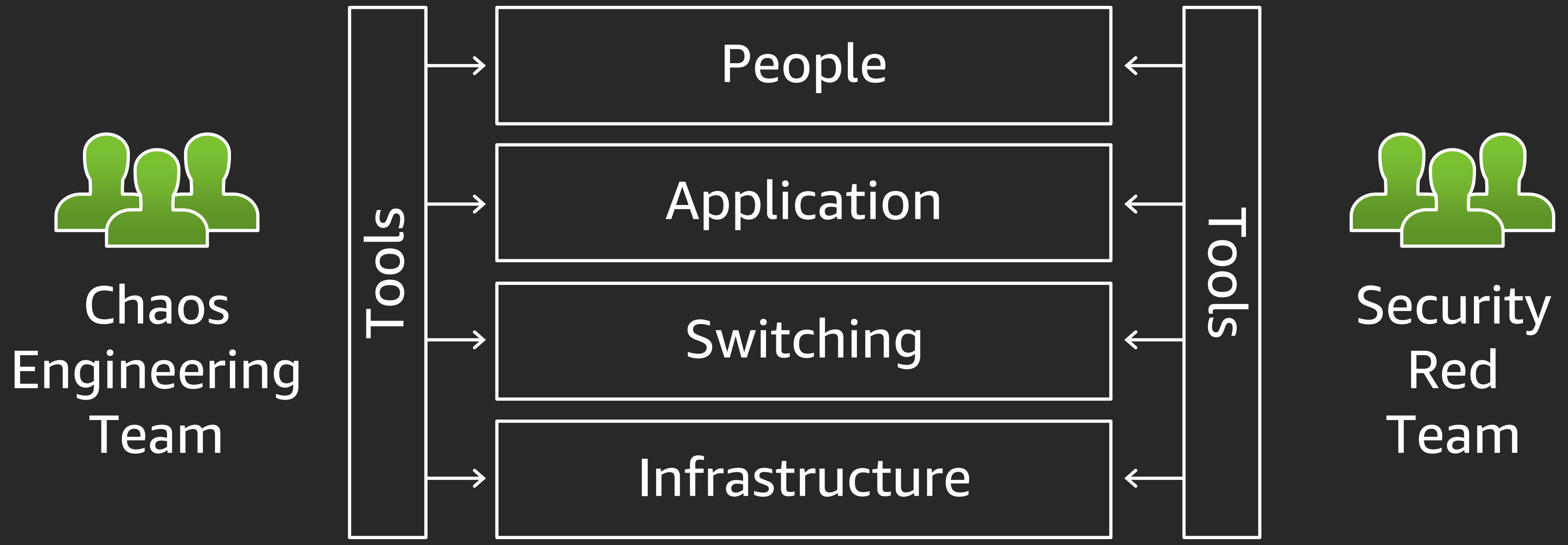
Chaos architecture

Four layers

Two teams

An attitude—

Find the weakest link



Defense in depth

Experienced staff

Robust applications

Dependable switching fabric

Redundant service foundation

“You can’t legislate against failure. Focus on fast detection and response.”

—Chris Pinkham

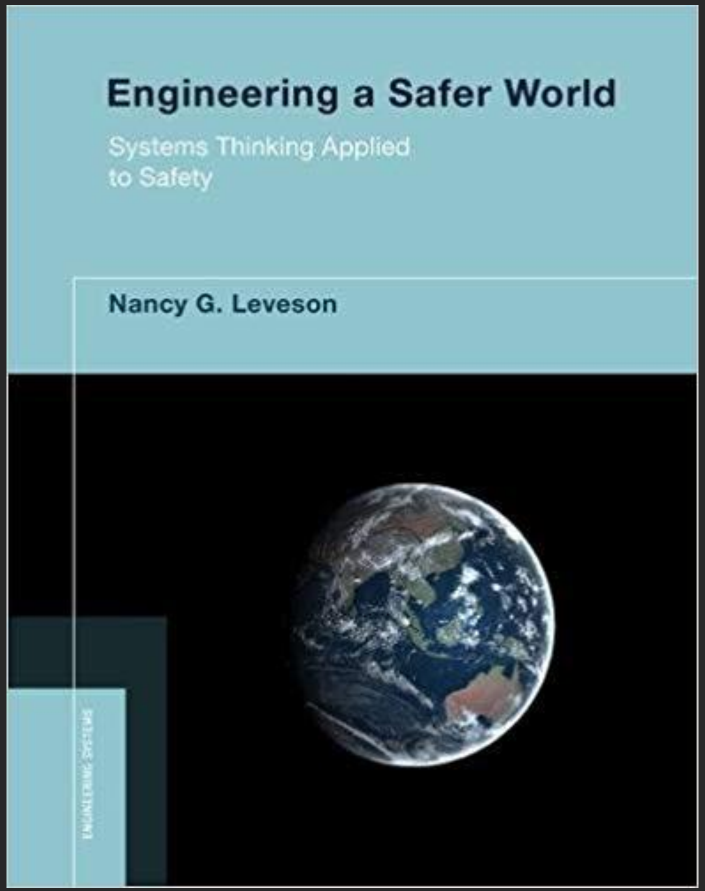
Observability

Kalman, 1961 paper

On the general theory of control systems

A system is observable If the behavior of the entire system can be determined by only looking at its inputs and outputs

Physical and software control systems are based on models, remember all models are wrong, but some models are useful…



Engineering a Safer World

Systems Thinking Applied to Safety

Nancy G. Leveson

STPA – Systems Theoretic Process Analysis

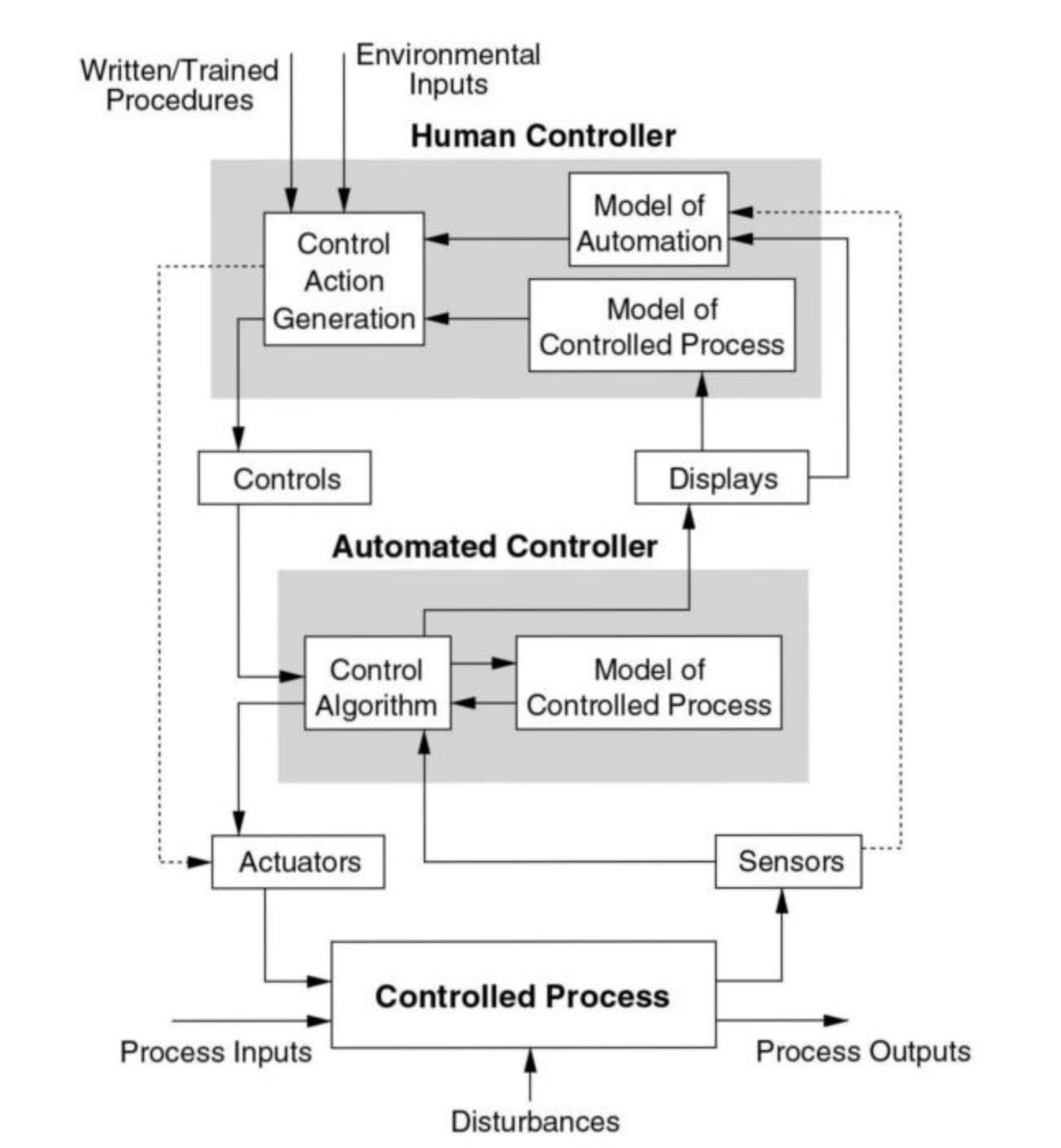
STAMP – Systems Theoretic Accident Model & Processes

http://psas.scripts.mit.edu for handbook and talks

Observability

STPA Model

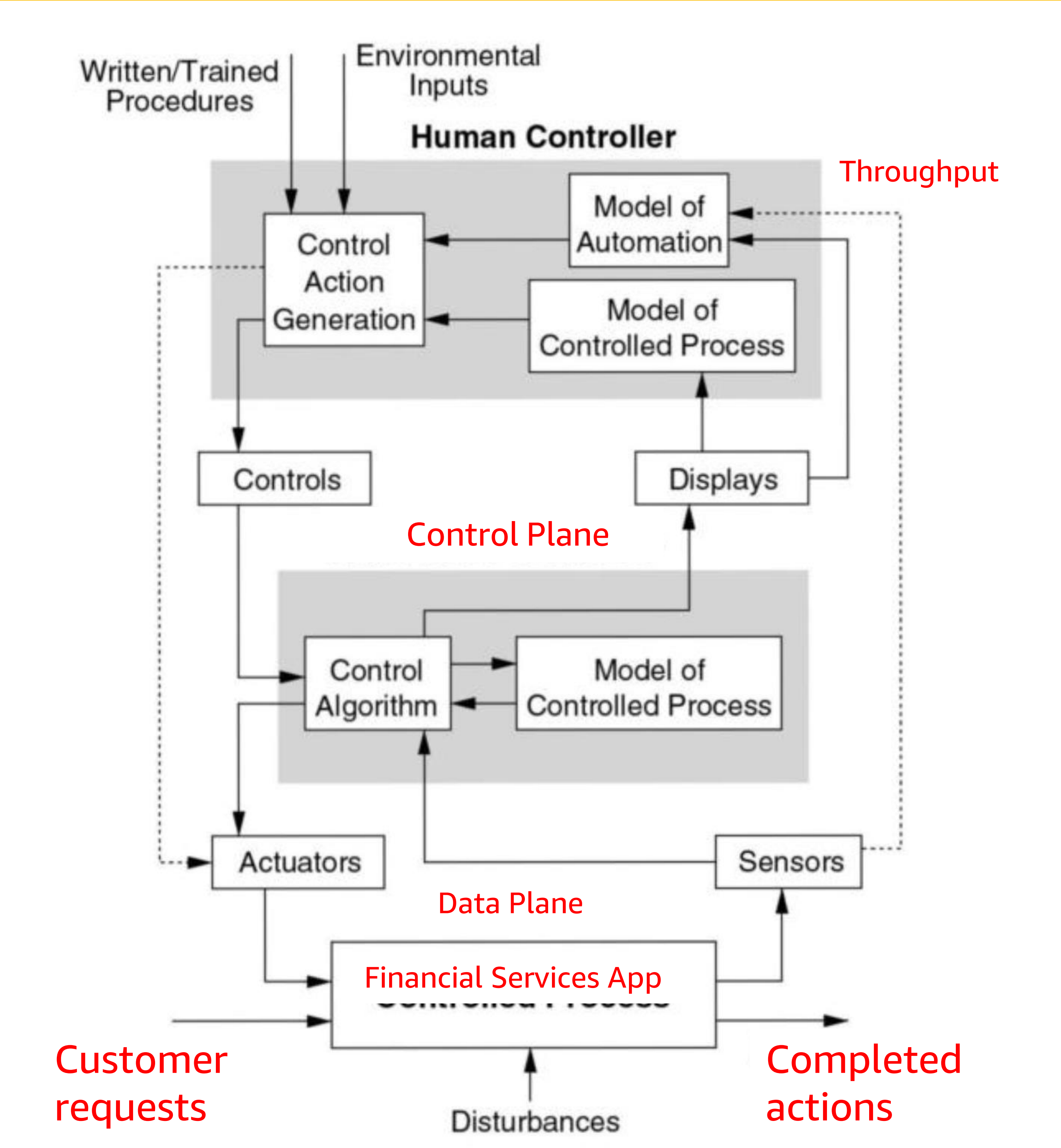
(System Theoretic Process Analysis)



Observability

STPA Model

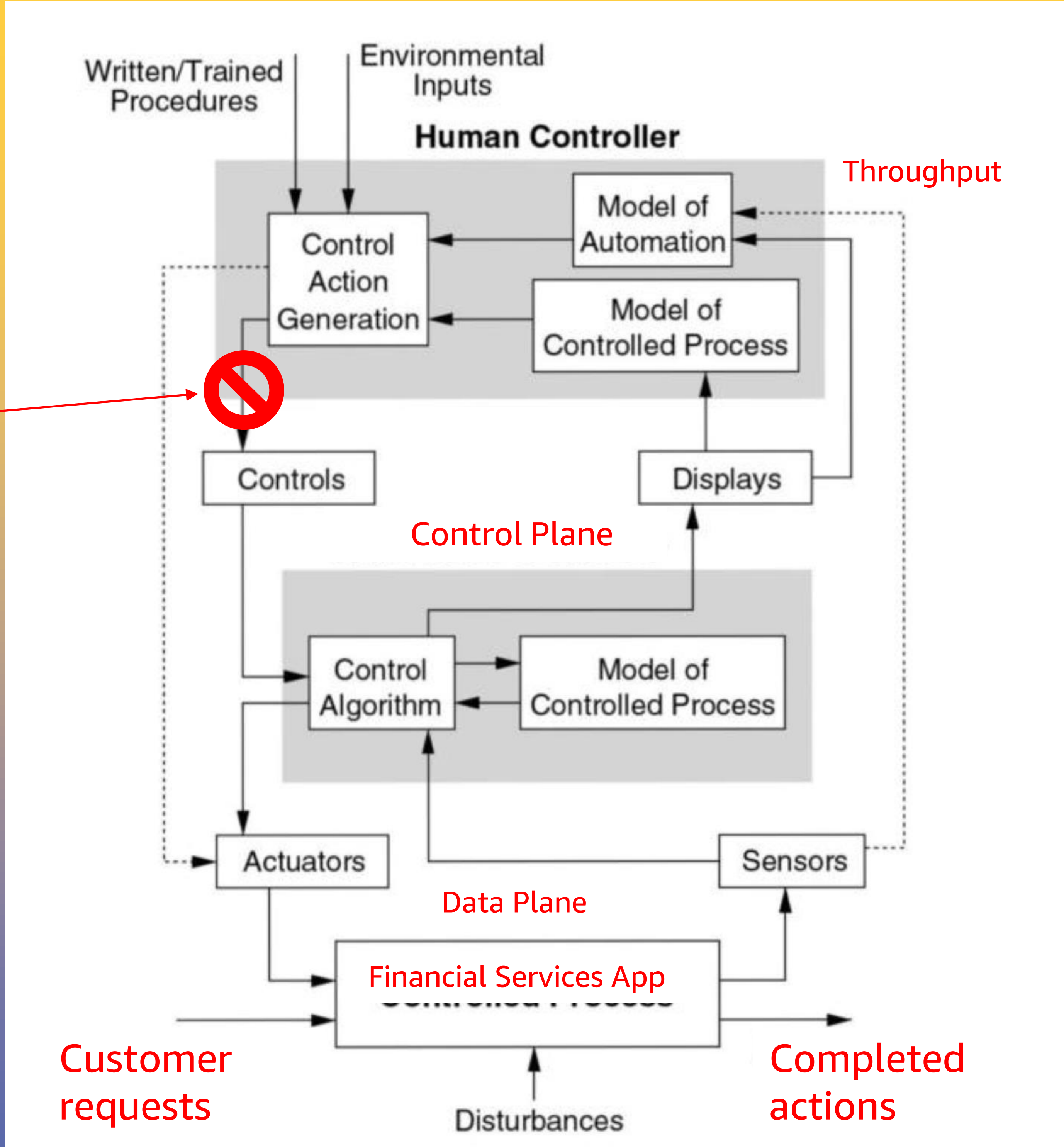
Understand Hazards that could disrupt successful application processing



STPA Hazards

Human Control Action:

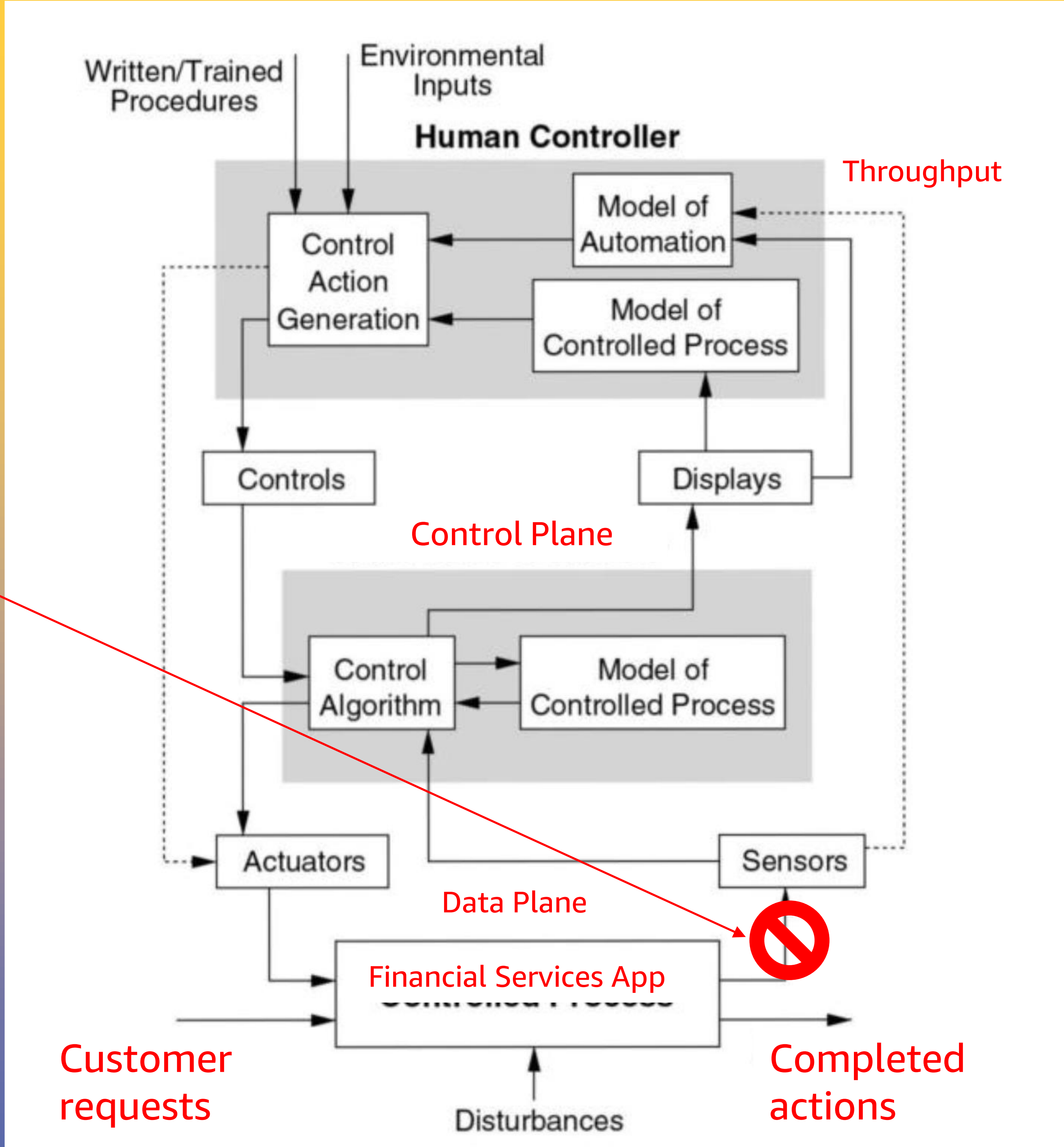
Not provided
Unsafe action
Safe but too early
Safe but too late
Wrong sequence
Stopped too soon
Applied too long
Conflicts



STPA Hazards

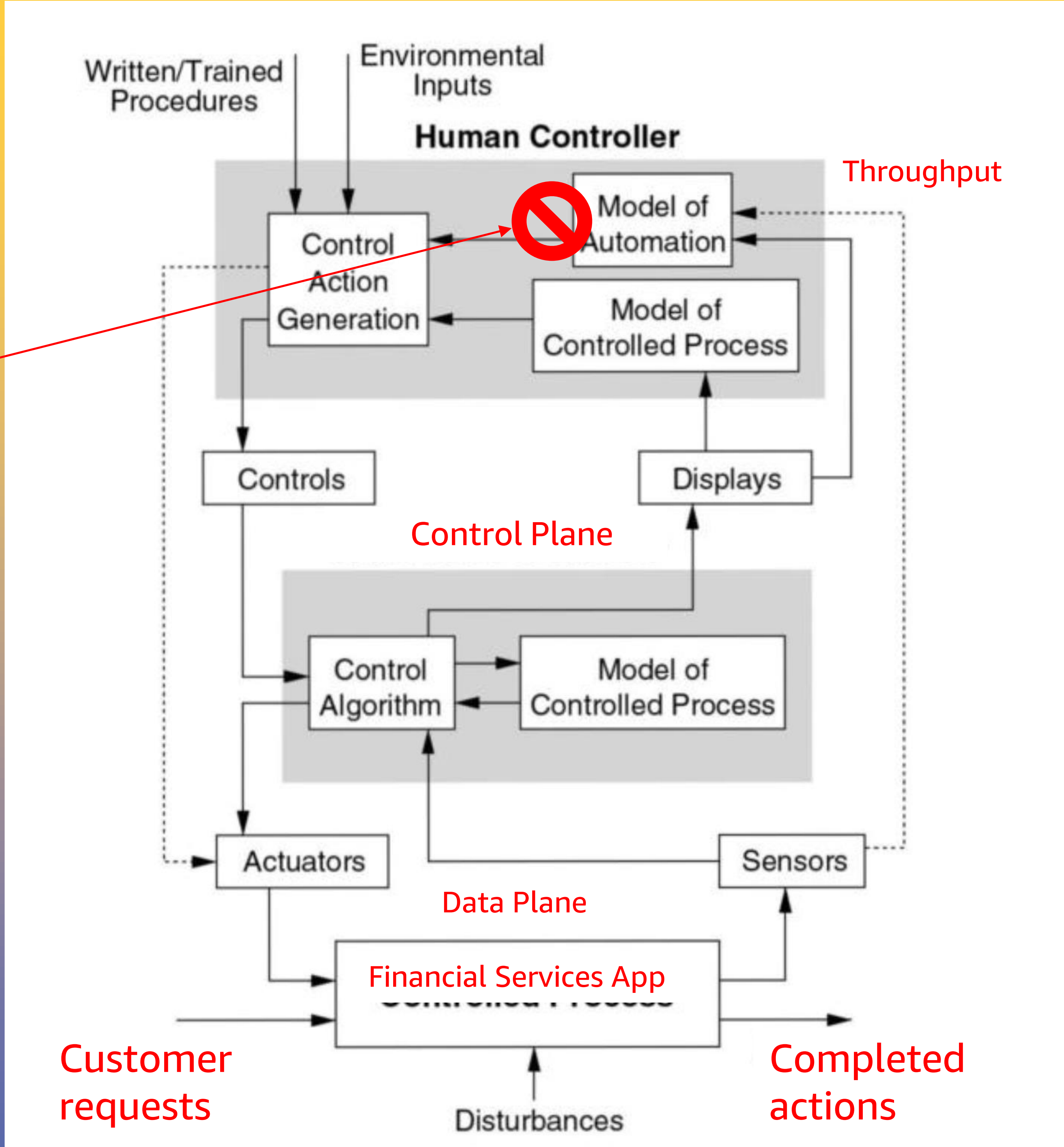
Sensor Metrics:

Missing updates
Zeroed
Overflowed
Corrupted
Out of order
Updates too rapid
Updates infrequent
Updates delayed
Coordination



STPA Hazards

Model problems:
Model mismatch
Missing inputs
Missing updates
Updates too rapid
Updates infrequent
Updates delayed
Coordination problems
Degradation over



How do we usually calculate risk?

Severity * Probability = Risk

Assumes that we can determine severity and probability

Assumes we always detect the failure when it occurs

Basic model for financial and economic risk analysis

Failure Modes and Effects Analysis (FMEA)

Engineering-oriented risk analysis

Severity * Probability * **Detectability** = Risk

Add observability to mitigate silent failures

Discuss and record component level failure modes

Prioritize mitigation work where it will do most good

FMEA for Web Services - Layered Responsibility

Product Managers and Developers – unique business logic

Software Platform Team – standard components and services

Infrastructure Platform Team – resources, regions, and networks

Resilience Engineering – observability and incident management

FMEA Spreadsheets: github.com/adrianco/slides

FMEA Severity Mapped to Infrastructure

Effect	SEVERITY of Effect	Ranking
Hazardous without warning	Earthquake or meteorite destroys data center building, no warning, people injured	**10**
Hazardous with warning	Hurricane or tornado destroys data center building, several days warning, people injured	**9**
Very High	Data center flooded, compute, and storage systems destroyed, building ok	**8**
High	Fire in data center, suppression system saves building, partial permanent compute and storage loss	**7**
Moderate	Hardware failure, CPU, disk, or power supply needs replacement. Often occurs after power or cooling failures.	**6**
Low	Power cut, cooling failure or network partition. Compute and storage returns when power, cooling and network are restored	**5**
Very Low	System operable with significant degradation of performance	**4**
Minor	System operable with some degradation of performance	**3**
Very Minor	System operable with minimal interference	**2**
None	No effect	**1**

FMEA Probability Per Service Request

Guess to start with, then measure in production

PROBABILITY of Failure	Failure Prob	Ranking
Very High: Failure is almost inevitable	>1 in 2	**10**
	1 in 3	**9**
High: Repeated failures	1 in 8	**8**
	1 in 20	**7**
Moderate: Occasional failures	1 in 80	**6**
	1 in 400	**5**
	1 in 2,000	**4**
Low: Relatively few failures	1 in 15,000	**3**
	1 in 150,000	**2**
Remote: Failure is unlikely	<1 in 1,500,000	**1**

FMEA Detectability

Needs an observable monitoring alert to detect a failure

Detection	Likelihood of DETECTION by Design Control	Ranking
Absolute Uncertainty	Design control cannot detect potential cause/mechanism and subsequent failure mode	**10**
Very Remote	**Very remote chance the design control will detect potential cause/mechanism and subsequent failure mode**	**9**
Remote	**Remote chance the design control will detect potential cause/mechanism and subsequent failure mode**	**8**
Very Low	**Very low chance the design control will detect potential cause/mechanism and subsequent failure mode**	**7**
Low	**Low chance the design control will detect potential cause/mechanism and subsequent failure mode**	**6**
Moderate	**Moderate chance the design control will detect potential cause/mechanism and subsequent failure mode**	**5**
Moderately High	**Moderately High chance the design control will detect potential cause/mechanism and subsequent failure mode**	**4**
High	**High chance the design control will detect potential cause/mechanism and subsequent failure mode**	**3**
Very High	**Very high chance the design control will detect potential cause/mechanism and subsequent failure mode**	**2**
Almost Certain	Design control will detect potential cause/mechanism and subsequent failure mode	**1**

FMEA Example – Application Level

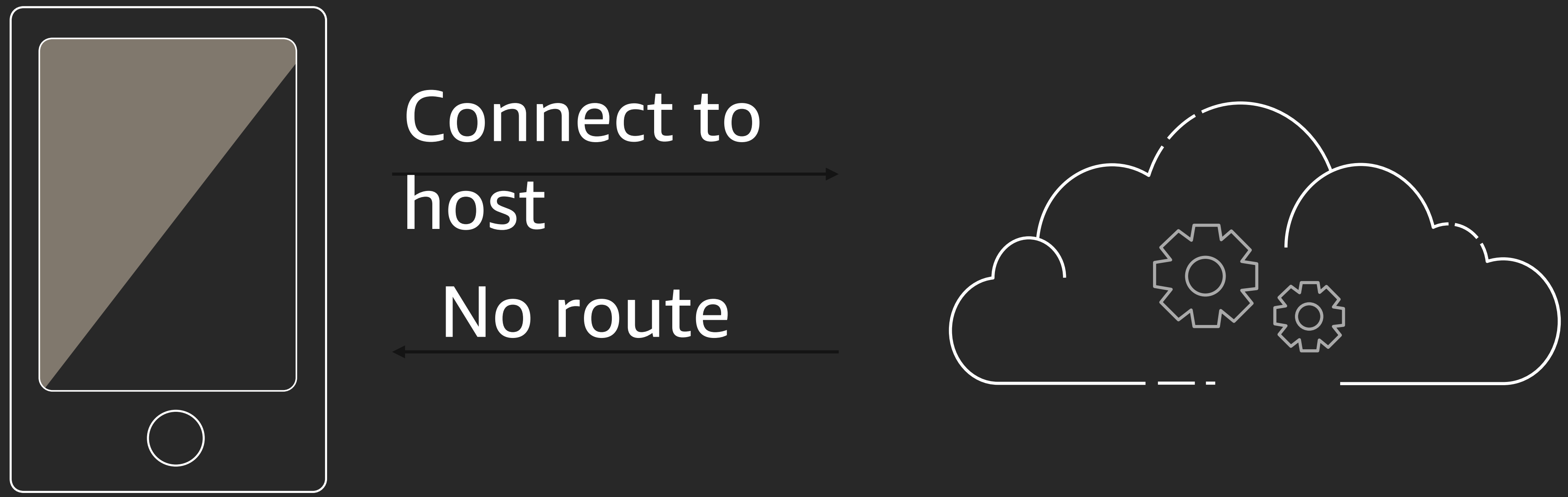
Customer is trying to obtain an IP address for a service

what could go wrong?

FMEA Example – Application Level

Customer is trying to make a request to a service

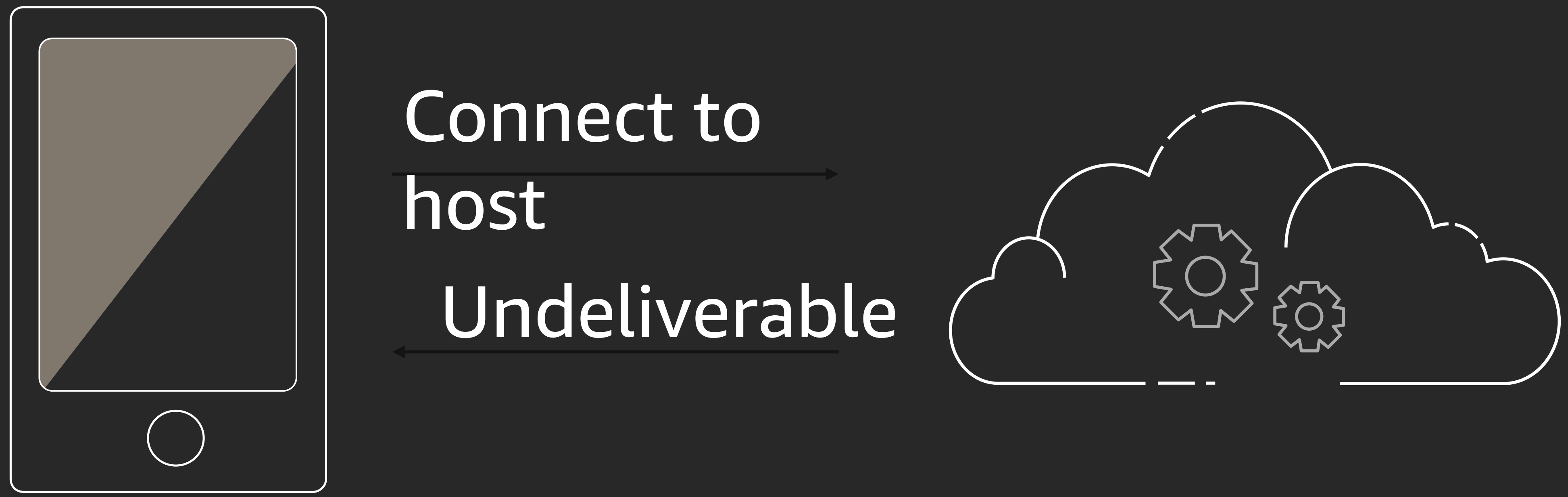
what could go wrong?



FMEA Example – Application Level

Customer is trying to make a request to a service

what could go wrong?



FMEA Example – Application Level

Customer is trying to make a request to a service

what could go wrong?

FMEA Example – Application Level

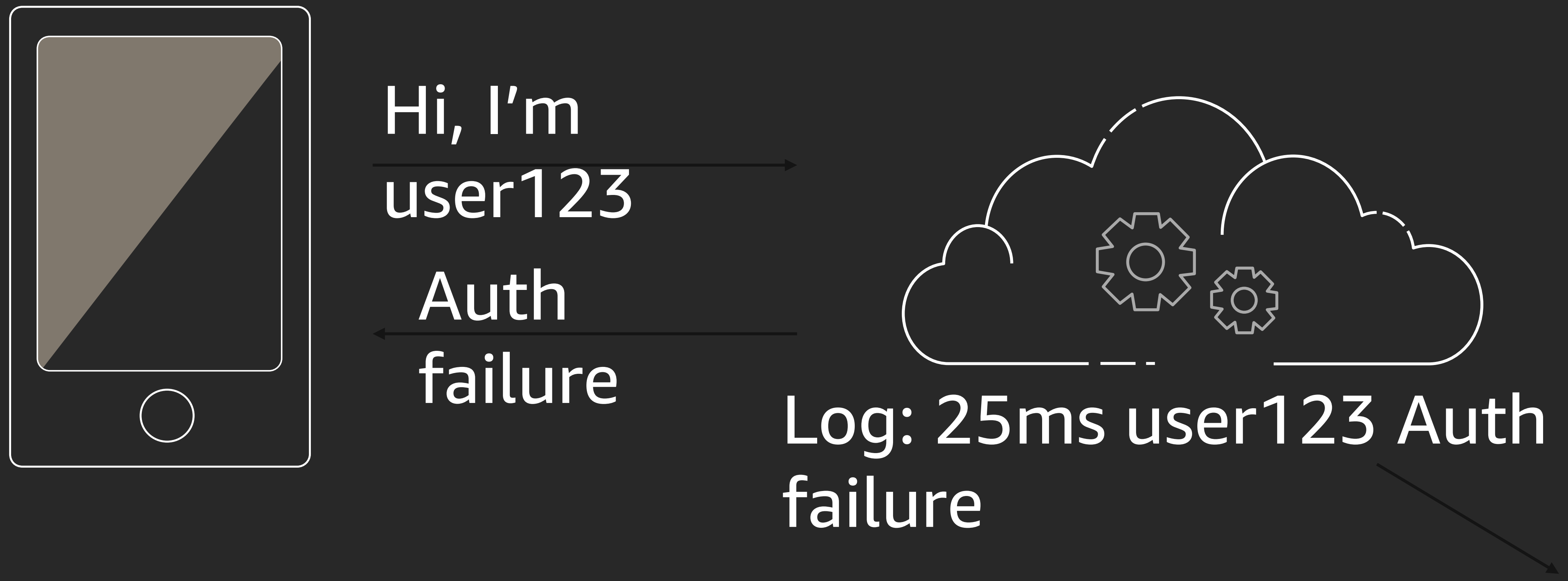
See full spreadsheets github.com/adrianco/slides for more failure modes

								0	
Client Request to API Endpoint	Service unknown, address un-resolvable	Delay while discovery or DNS times out, slow fallback response	5	DNS configuration error, denial of service attack, or provider failure	1	Customer eventually complains via call center	10	**50**	Dual redundant DNS, fallback to local cache, hardcoded IP addresses. Endpoint monitoring and alerts
	Service unreachable, request undeliverable	Fast fail, no response	4	Network route down or no service instances running	1	Autoscaler maintains a number of healthy instances	1	**4**	Endpoint monitoring and alerts
	Service reachable, request undeliverable	Connect timeout, slow fail, no response	4	Service frozen/not accepting connection	1	Retry request on different instance. Healthcheck failed instances removed. Log and alert.	2	**8**	
	Request delivered, no response - stall	Application request timeout, slow fail, no response	4	Broken service code, overloaded CPU or slow dependencies	1	Retry request on different instance. Healthcheck failed instances removed. Log and alert.	2	**8**	

FMEA Example – Application Level

Customer is trying to make a request to a service

what could go wrong?



FMEA Example – Application Level

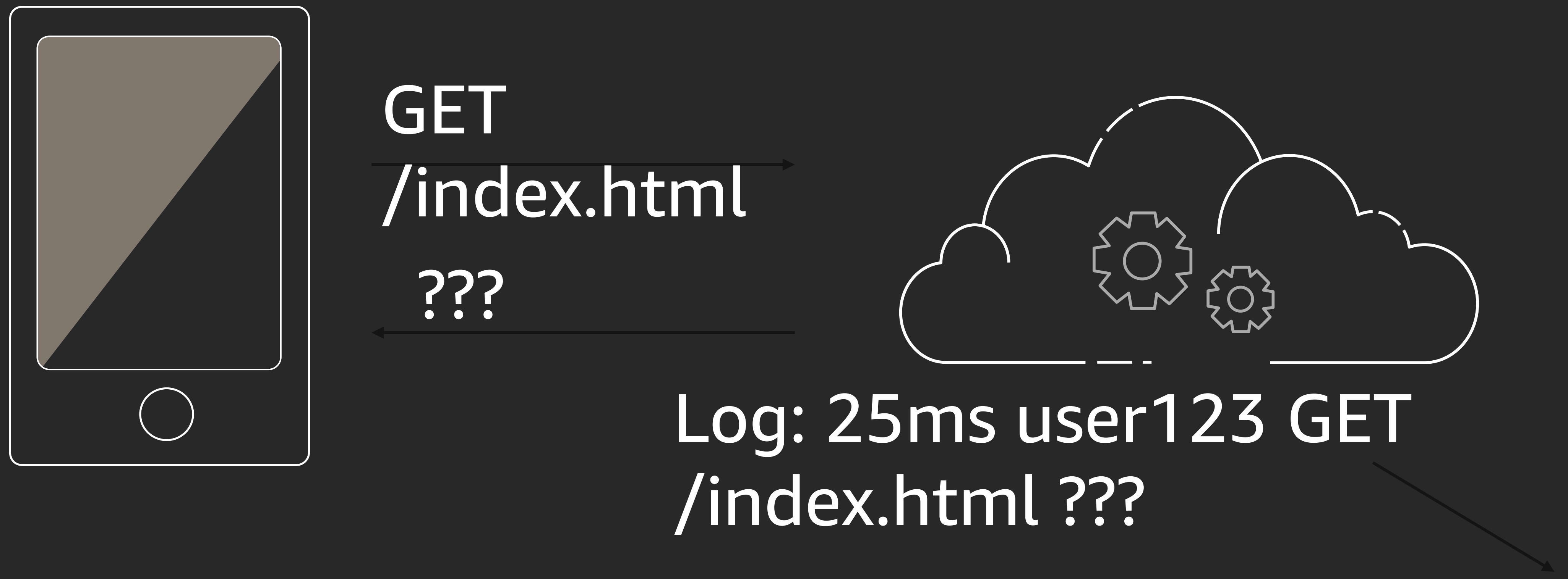
Authentication Failures

Item / Function	Potential Failure Mode(s)	Potential Effect(s) of Failure	Sev	Potential Cause(s)/ Mechanism(s) of Failure	Prob	Current Design Controls	Det	RPN	Recommended Action(s)
Authentication	Client can't authenticate	Can't connect application	5	Certificate timeout, version mismatch, account not setup, credential changed	3	Log and alert on authentication failures	3	**45**	
	Slow or unreliable authentication	Slow start for application	4	Auth service overloaded, high error and retry rate	3	Log and alert on high authentication latency and errors	4	**48**	

FMEA Example – Application Level

Customer is trying to make a request to a service

what else could go wrong?



FMEA Example – Application Level

Application Failures

Time Bombs	Internal application counter wraparound								Test long running operations of code base
	Memory leak								Monitor process sizes and garbage collection intervals over time
Date Bombs	Leap year, leap second, epoch wrap around, "Y2K"								Test across date boundaries
Content Bombs	Incoming data that crashes the app								Fuzz the input with generated random and structured data to show it doesn't crash.
Configuration Errors	Configuration file syntax errors or incorrect values								Canary test deployments incrementally. Chaos testing.
Versioning Errors	Incompatible interface versions								Canary test deployments incrementally
Retry Storms	Too many retries, too large timeout values								Chaos testing applications under stress
Excessive Logging	Cascading overload								Chaos testing applications under stress

STPA – Top down focus on control hazards

FMEA – Bottom up focus on prioritizing failure modes

STPA tends to have better failure coverage than FMEA, especially for human controller/user experience issues

Both are useful…

Cloud provides the automation
that leads to chaos engineering

Good Cloud Resilience Practices

Rule of 3 – three ways for critical operations to succeed

- Synchronous data replication over three zones in a region
- DR failover from primary region to either of two secondary regions
- Active-Active-Active workloads across three regions

Good Cloud Resilience Practices

Fail up - DR failover between regions

- From smaller capacity region to larger capacity region
- From distant region to closer (lower latency) region

Good Cloud Resilience Practices

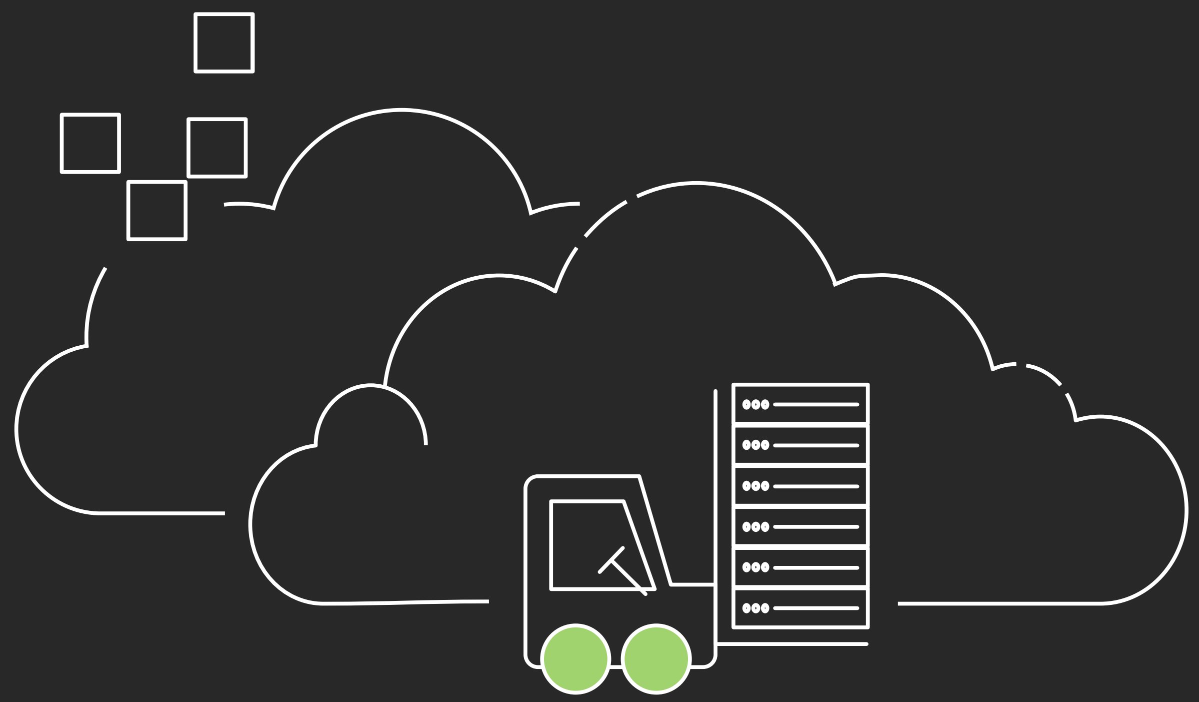
Chaos first

- Build your resilience environment *before* introducing apps to it
- Automated continuous zone and region failover testing
- Make it a “badge of honor” to have an app pass the chaos test

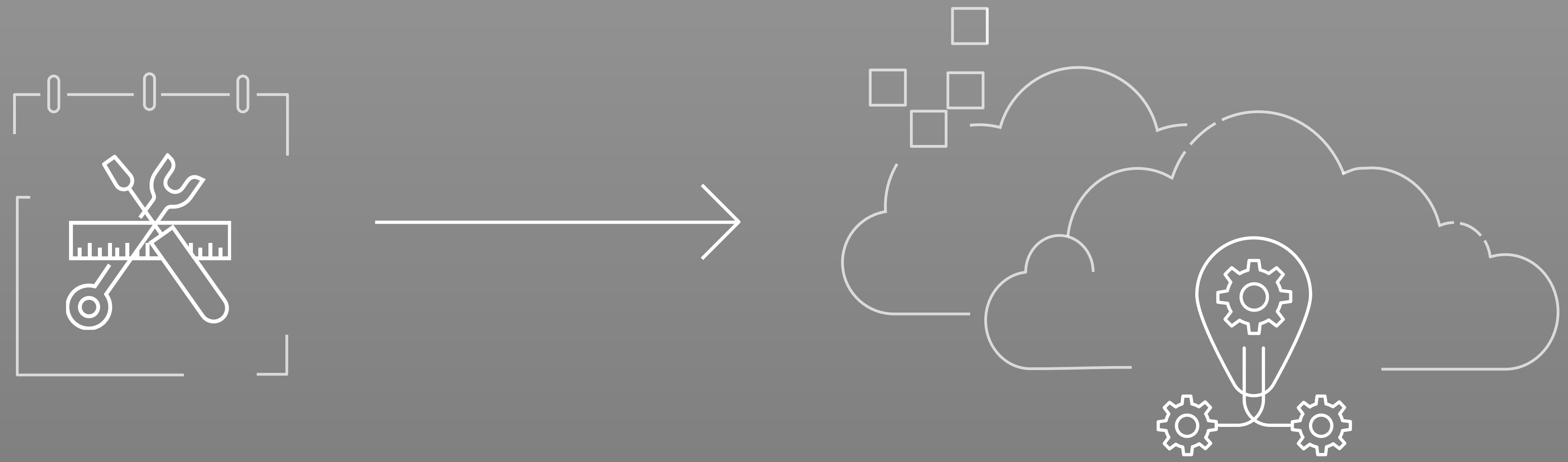
Good Cloud Resilience Practices

Continuous Resilience

- Continuous Delivery needs Test Driven Development and Canaries
- Continuous Resilience needs automation in both test and production
- Make failure mitigation into a well tested code path and process
- Call it Chaos Engineering if you like, it’s the same thing…



As data centers migrate to cloud, fragile and manual disaster recovery processes can be standardized and automated



Testing failure mitigation will move from a scary annual experience to automated **continuous resilience**

References

AWS Whitepaper: Building Mission Critical Financial Services Applications on AWS
- By Pawan Agnihotri with contributions by Adrian Cockcroft

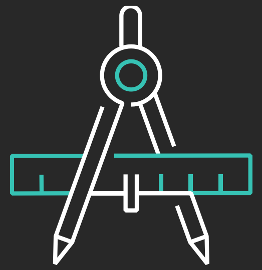
Blog Post (Failure Modes and Continuous Resilience): http://bit.ly/continuous-resilience
- By Adrian Cockcroft

Related sessions

Session ID	Title	Type	Level
ARC303-R2	[REPEAT 2] Failing successfully: The AWS approach to resilient design	Chalk Talk	300
ARC342-R	[REPEAT] Cell-based architectures for global, well-architected apps	Chalk Talk	300
ARC306-R1	[REPEAT 1] Reliability of the cloud: How AWS achieves high availability	Chalk Talk	300
ARC309-R1	[REPEAT 1] Hands-on: Building a multi-region active-active solution	Workshop	300
ARC317-R	[REPEAT] Building global applications that align to BC/DR objectives	Workshop	300
ARC404-R1	[REPEAT 1] Resiliency testing: Verifying your system is as reliable as you think	Workshop	400
ARC411-R1	[REPEAT] Reducing blast radius with cell-based architectures	Session	400
ARC349-R1	[REPEAT 1] Beyond five 9s: Lessons from our highest available data planes	Session	400

Learn to architect with AWS Training and Certification

Resources created by the experts at AWS to propel your organization and career forward



Free foundational to advanced digital courses cover AWS services and teach architecting best practices

Classroom offerings, including Architecting on AWS, feature AWS expert instructors and hands-on labs

Validate expertise with the **AWS Certified Solutions Architect - Associate** or **AWS Certification Solutions Architect - Professional** exams

Visit aws.amazon.com/training/path-architecting/

aws training and certification

Thank you!

Please complete the session survey in the mobile app.