

AWS re:Invent

NOV. 28 – DEC. 2, 2022 | LAS VEGAS, NV

ARC401-R

Achieving extreme resiliency through recovery-oriented architectures

James Lamanna (he/him)

Senior Principal Engineer
Amazon Web Services

Steven Hooper (he/him)

Principal Solutions Architect
Amazon Web Services



Agenda

1. Designing for failure and automated recovery (15 mins)
2. Amazon Route 53 application recovery controller (5 mins)
3. Q&A (35 mins)

Recovery-oriented design

Build on the principle that failure will happen

“Everything fails, all the time.”

Dr. Werner Vogels, VP and CTO, Amazon

Recovery-oriented design

Build on the principle that failure will happen

“Everything fails, all the time.”

Werner Vogels, VP and CTO, Amazon

Centers on limiting and coping with failure

Recovery-oriented design

Build on the principle that failure will happen

"Everything fails, all the time."

Werner Vogels, VP and CTO, Amazon

Centers on limiting and coping with failure

Focus on recovery readiness and control

Recovery-oriented design

Insulating from a failure is faster than fixing it

Recovery-oriented design

Insulating from a failure is faster than fixing it

Recovery actions are more broadly applicable

Recovery-oriented design

Insulating from a failure is faster than fixing it

Recovery actions are more broadly applicable

Recovery can be automated to reduce time and errors

Recovery can be tested to build confidence

Recovery design goals

Reliable

Recovery design goals

Reliable

Safe

Recovery design goals

Reliable

Safe

Fast

Recovery design goals

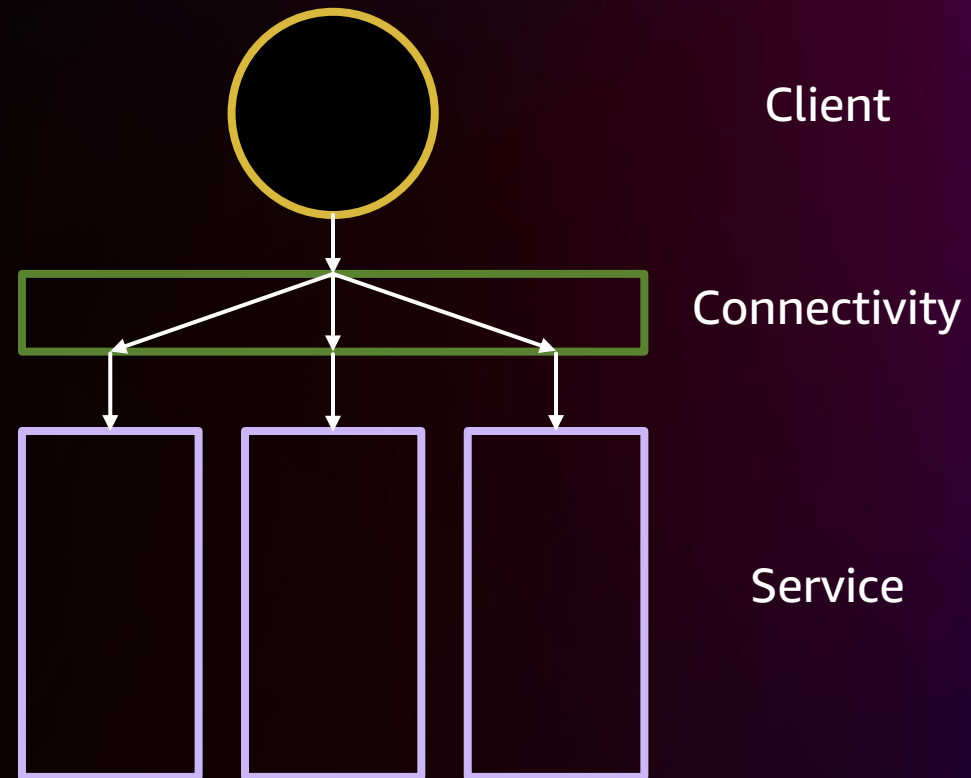
Reliable

Safe

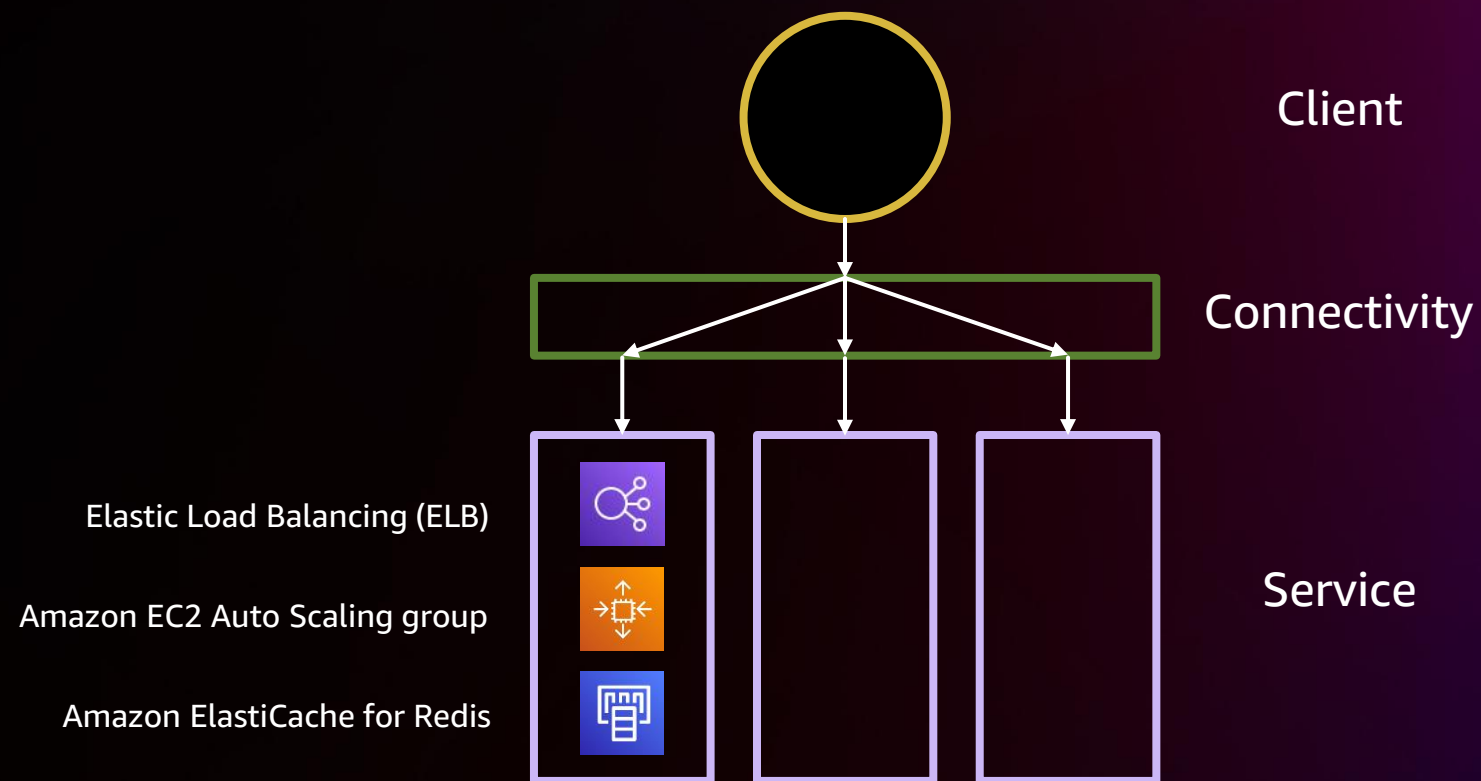
Fast

Categorical

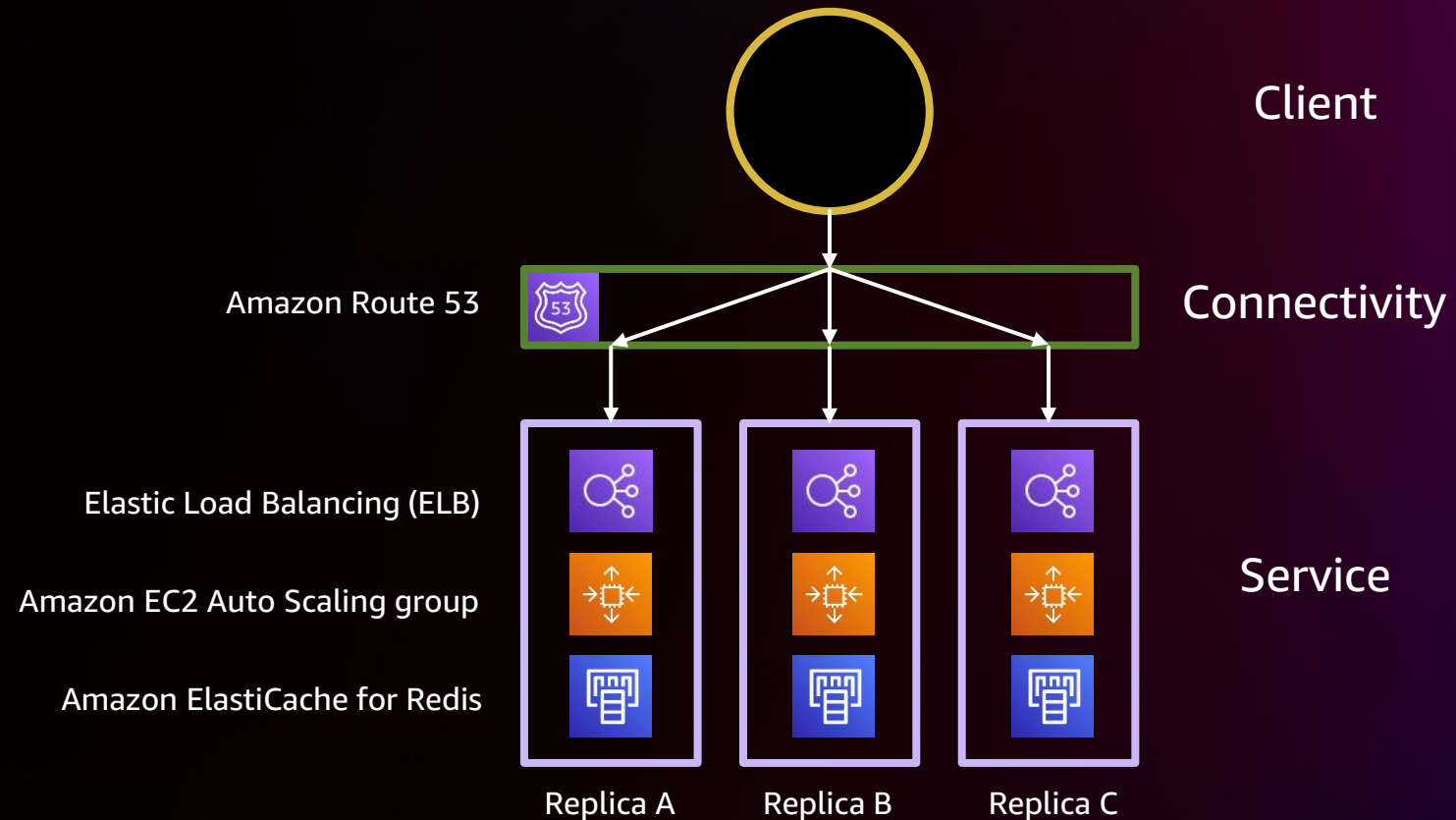
Recovery-oriented architecture



Recovery-oriented architecture



Recovery-oriented architecture



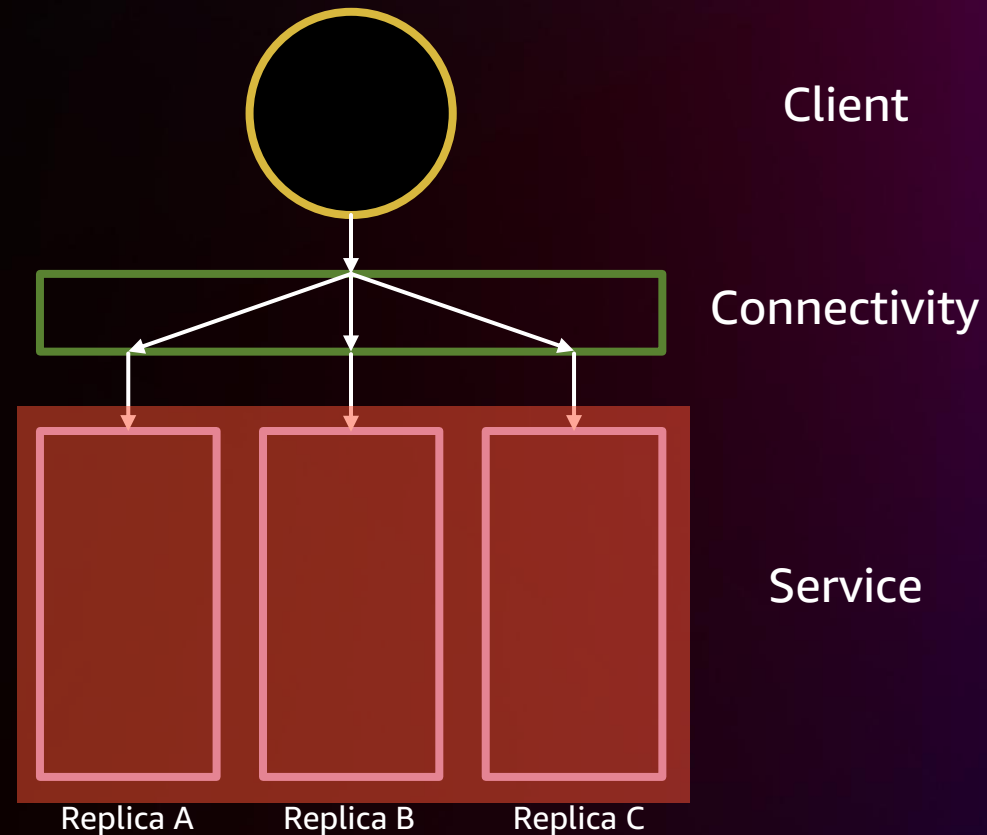
Design for failure

How can we manage failure?

Design for failure

Categories of failure

Code and configuration

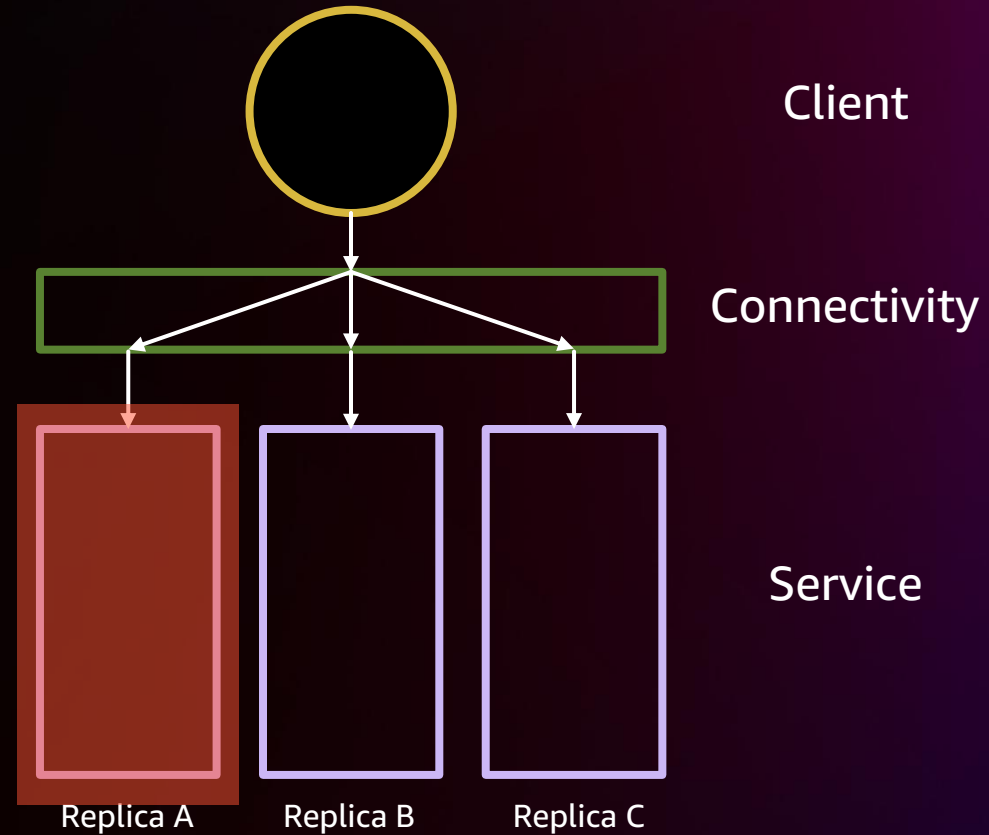


Design for failure

DEPLOY TO ONE REPLICAS AT A TIME

Categories of failure

Code and configuration



Design for failure

DEPLOY TO ONE REPLICA AT A TIME

Categories of failure

Code and configuration

Code and configuration failures:

- Bad code deployments
- Application misconfiguration
- Security group misconfiguration
- Routing policy misconfiguration
- TLS certificates misconfiguration
- Cert and credential expiration

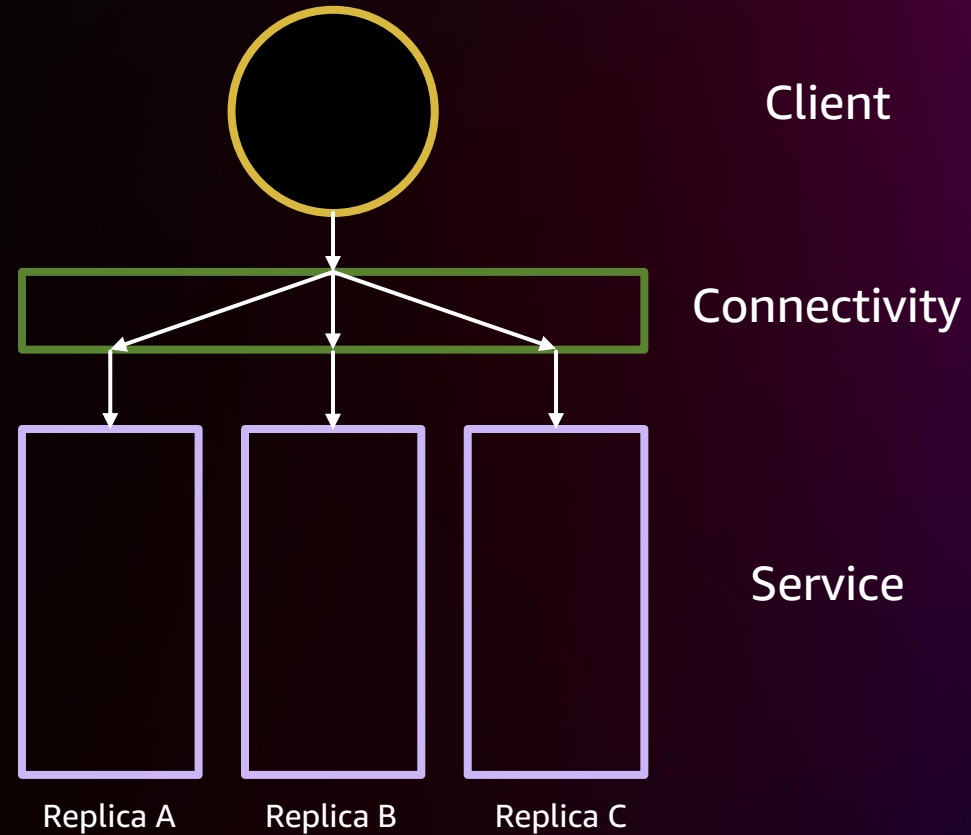
Certification and credential expiration

Design for failure

Categories of failure

Code and configuration

Infrastructure

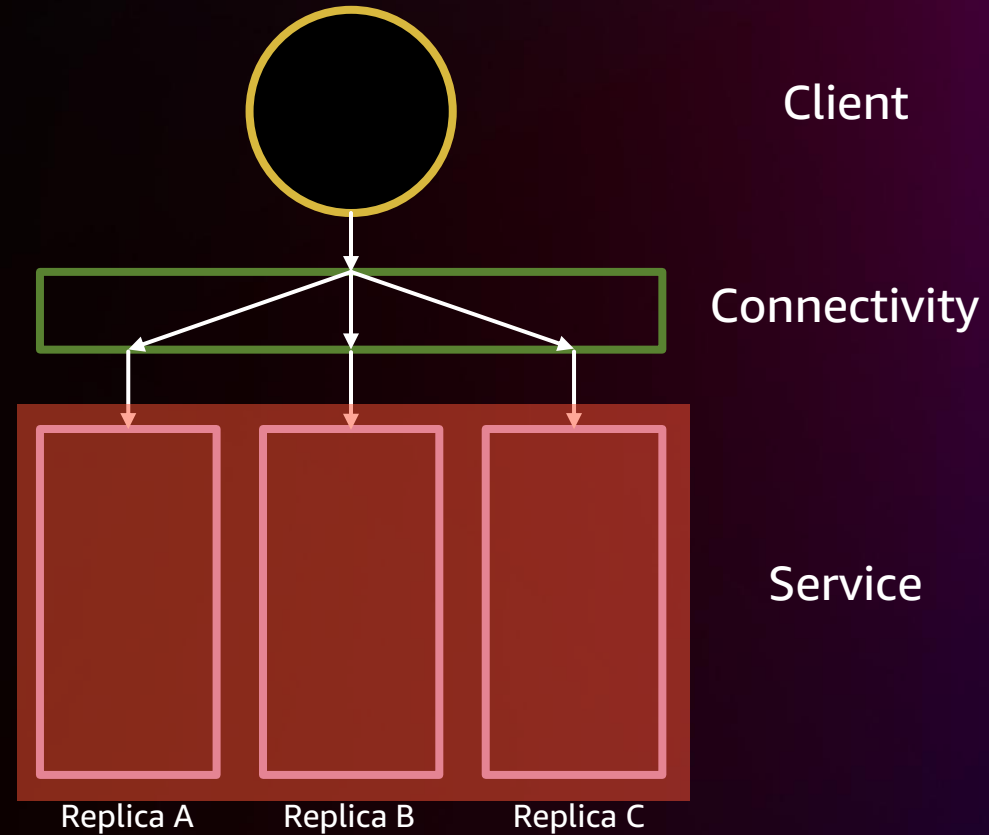
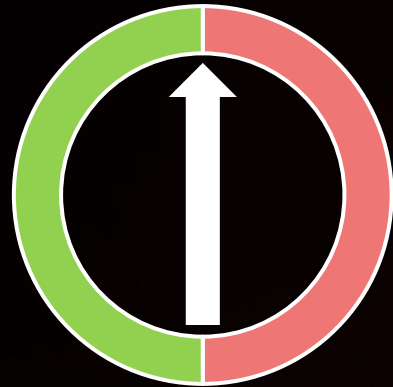


Design for failure

Categories of failure

Code and configuration

Infrastructure



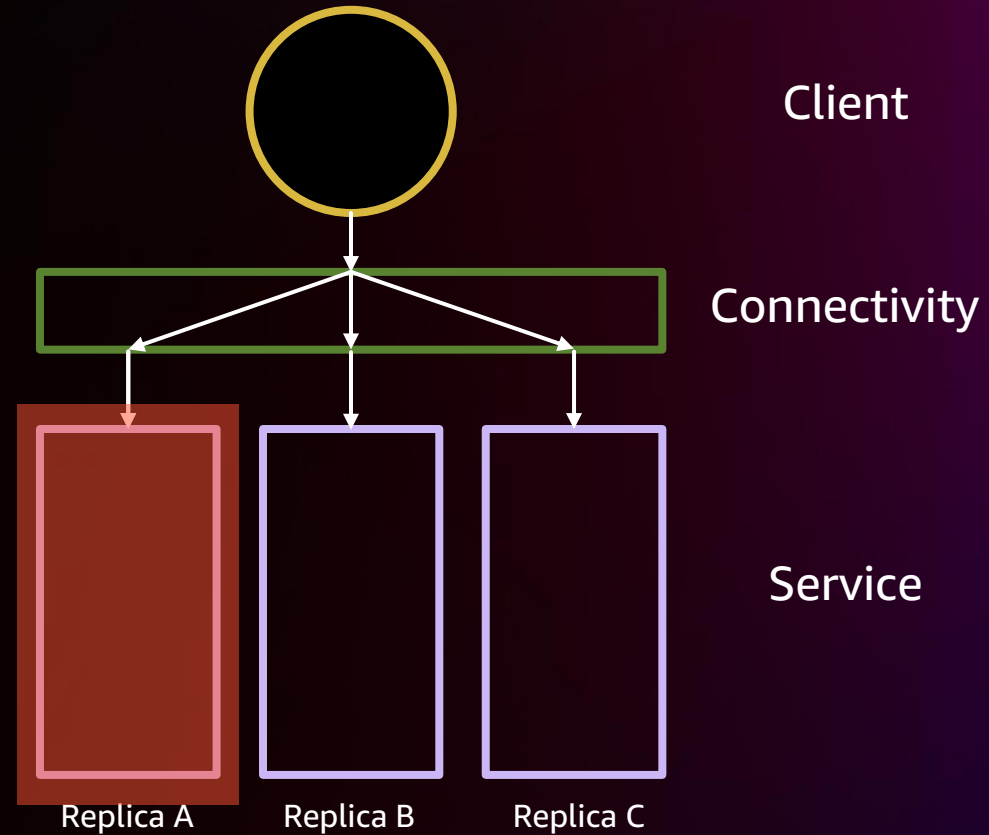
Design for failure

PLACE REPLICAS ON INDEPENDENT INFRASTRUCTURE

Categories of failure

Code and configuration

Infrastructure



Design for failure

PLACE REPLICAS ON INDEPENDENT INFRASTRUCTURE

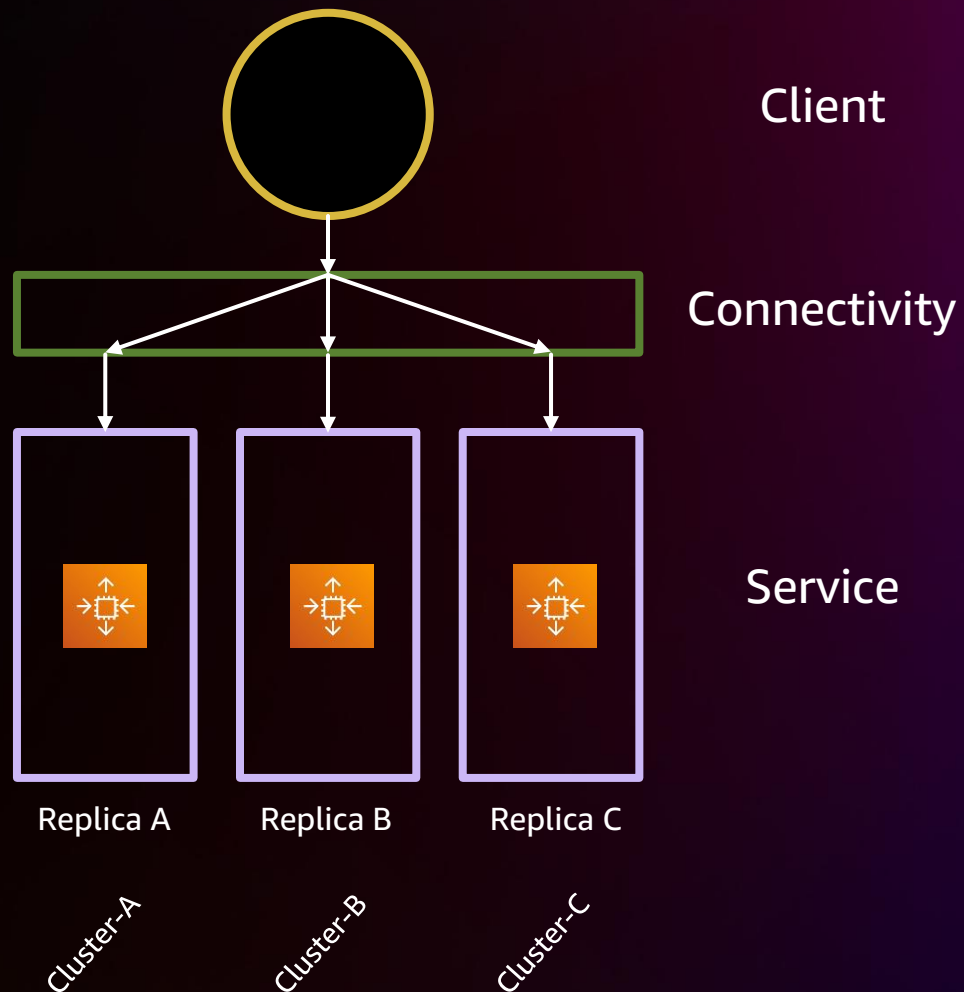
EC2 placement groups

Availability Zones

Regions

Amazon EC2 Auto Scaling group

EC2 placement group "cluster"



Design for failure

PLACE REPLICAS ON INDEPENDENT INFRASTRUCTURE

EC2 placement groups

Availability Zones

Regions

Elastic Load Balancing (ELB)

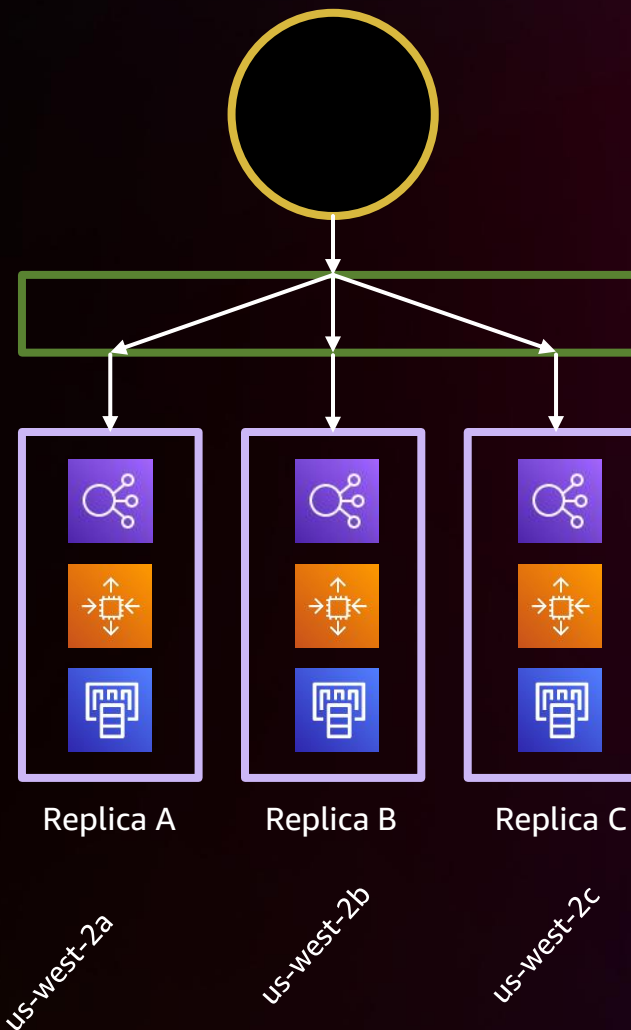
Amazon EC2 Auto Scaling group

Amazon ElastiCache for Redis

Client

Connectivity

Service



Design for failure

PLACE REPLICAS ON INDEPENDENT INFRASTRUCTURE

EC2 placement groups

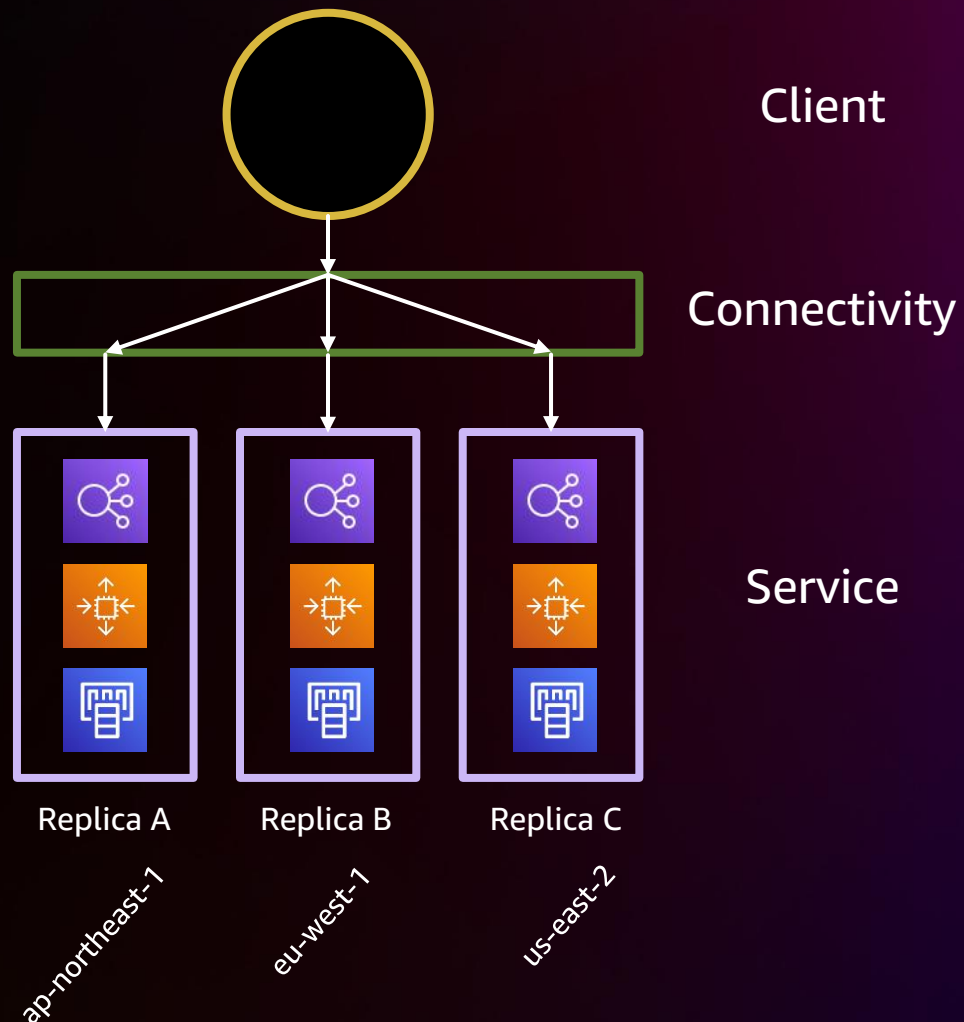
Availability Zones

Regions

Elastic Load Balancing (ELB)

Amazon EC2 Auto Scaling group

Amazon ElastiCache for Redis



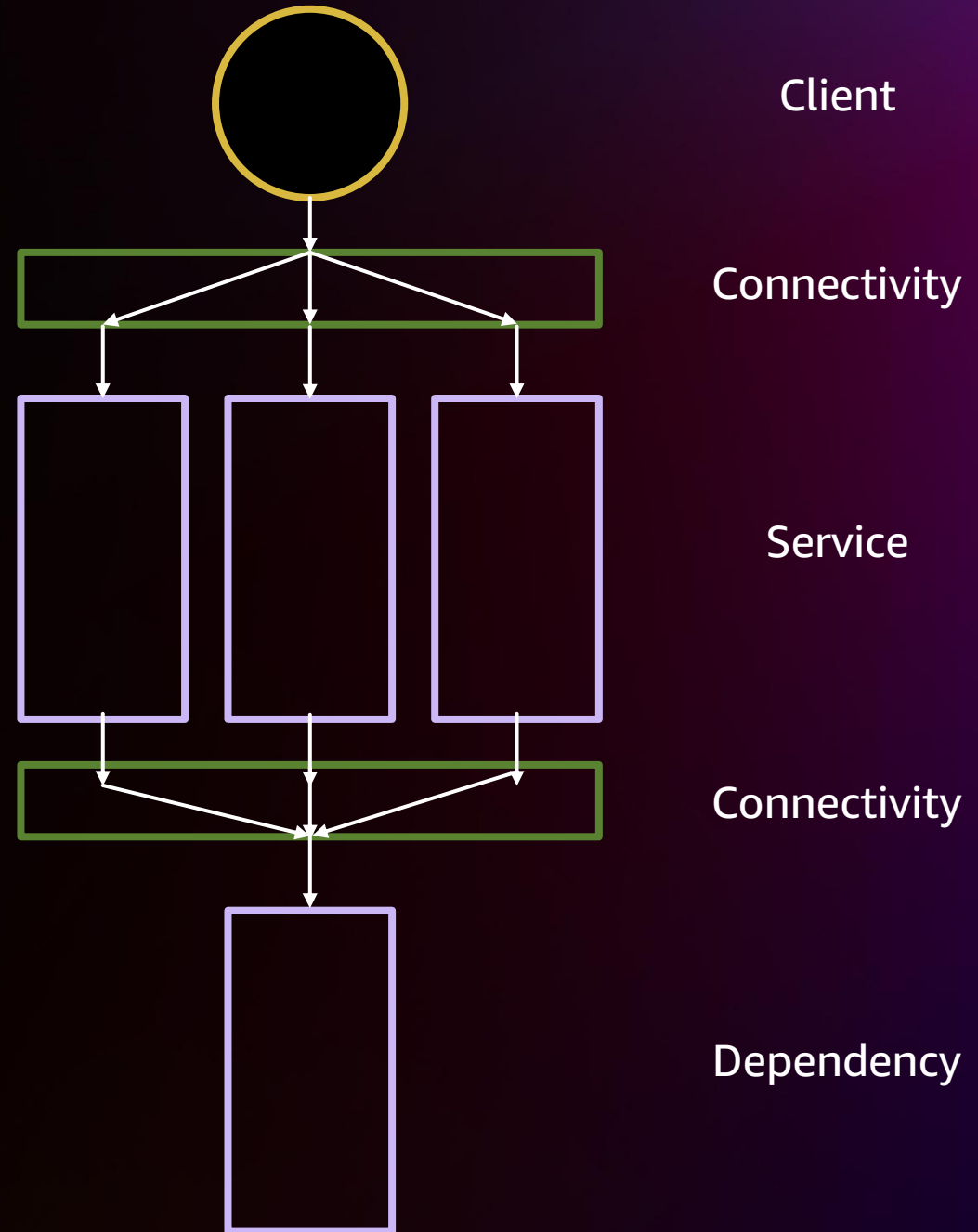
Design for failure

Categories of failure

Code and configuration

Infrastructure

Dependencies



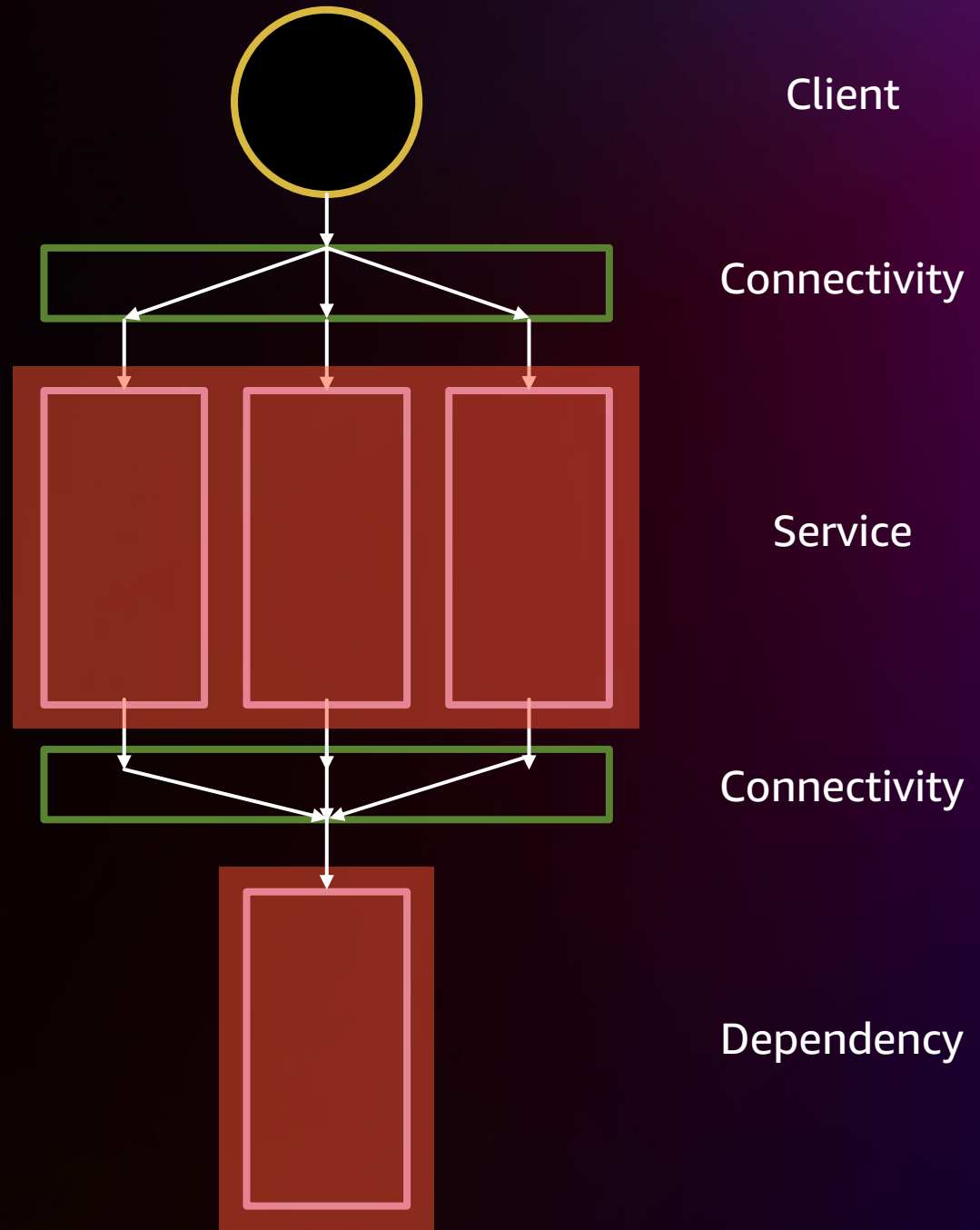
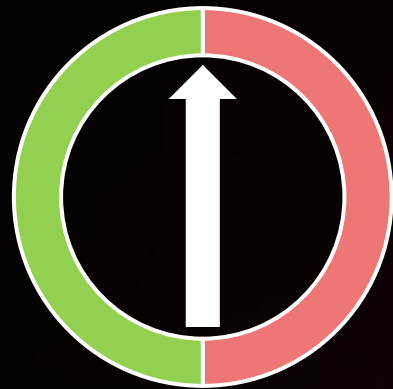
Design for failure

Categories of failure

Code and configuration

Infrastructure

Dependencies



Design for failure

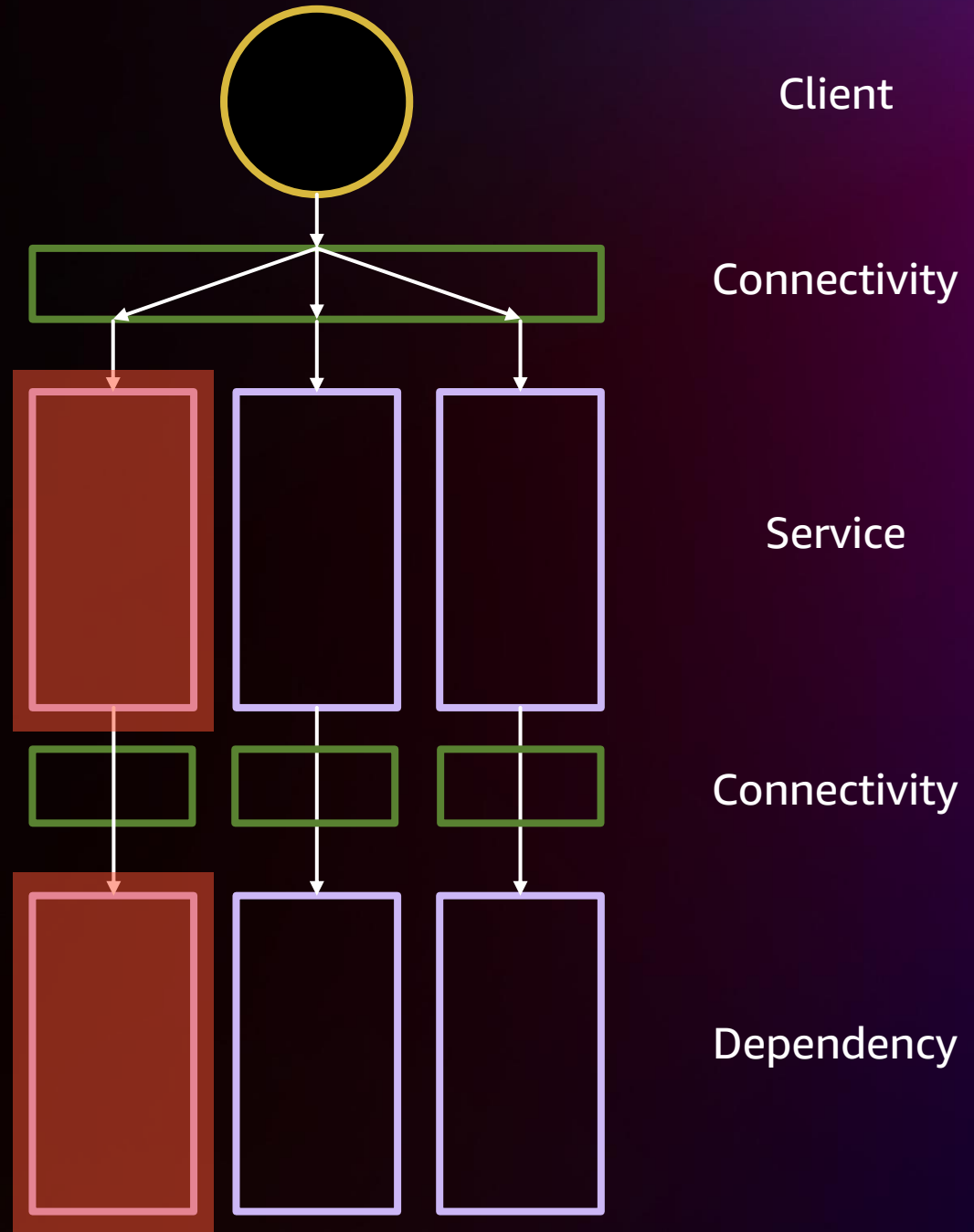
USE INDEPENDENT DEPENDENCIES

Categories of failure

Code and configuration

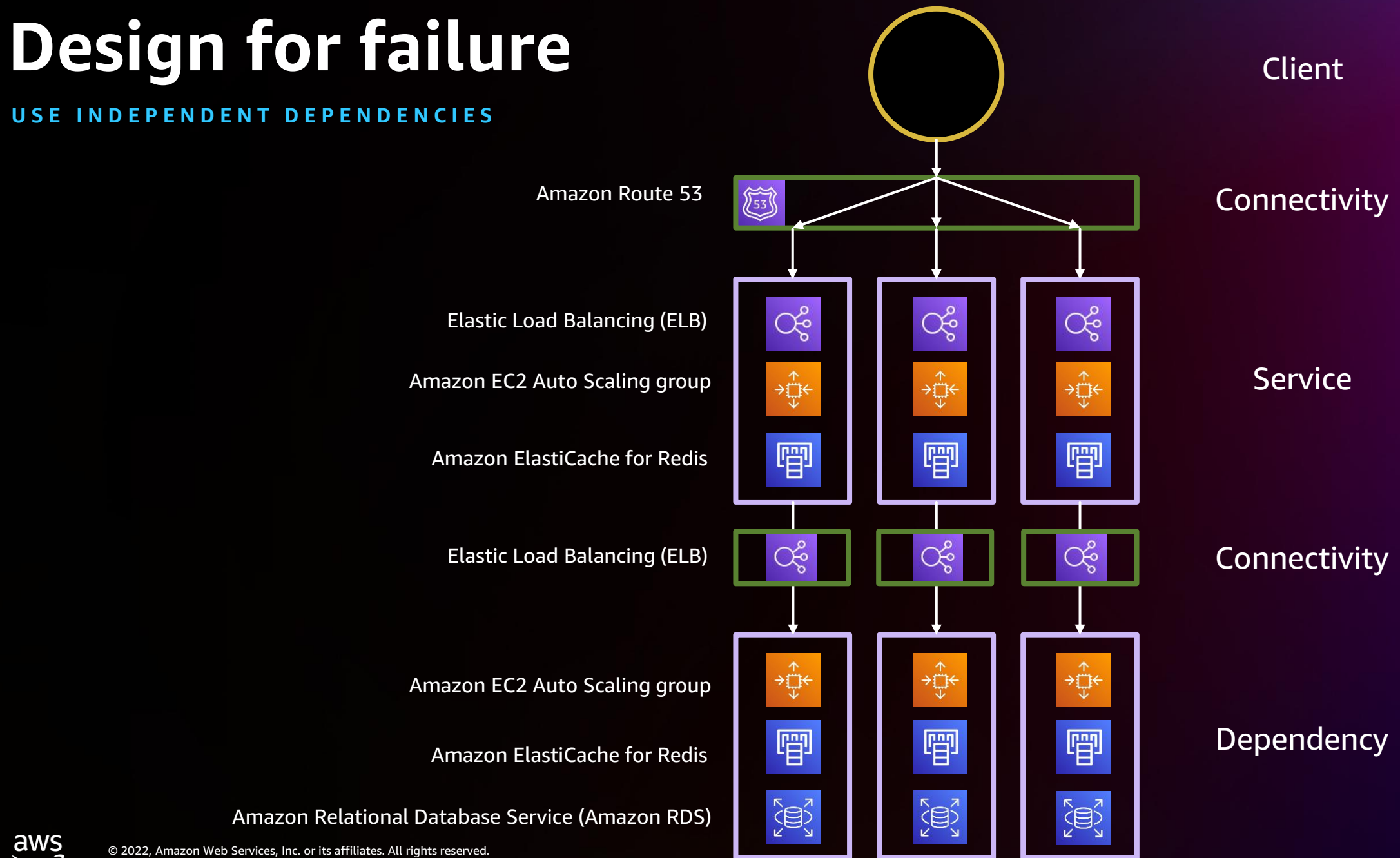
Infrastructure

Dependencies



Design for failure

USE INDEPENDENT DEPENDENCIES



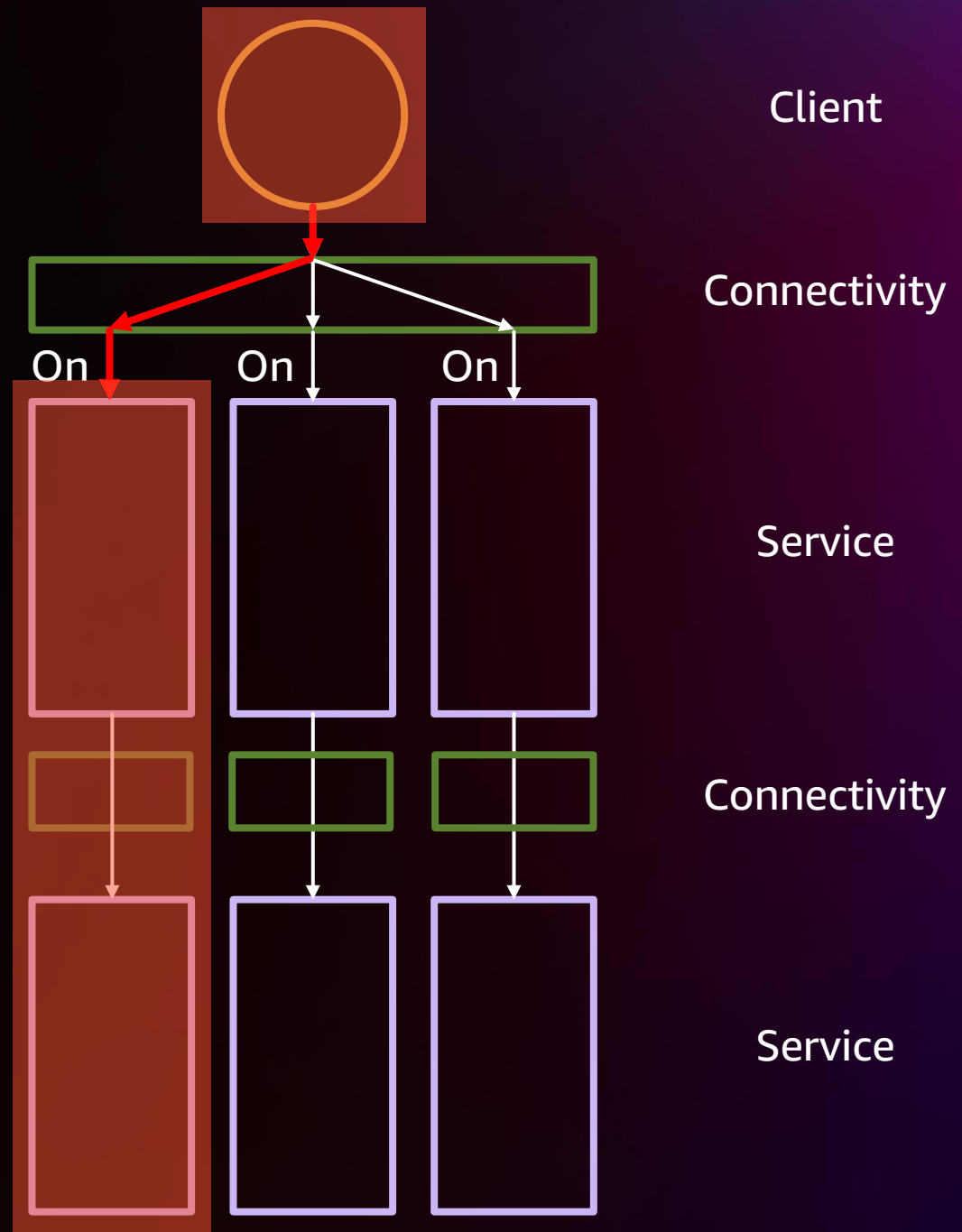
Design for failure

Categories of failure

Code and configuration

Infrastructure

Dependencies



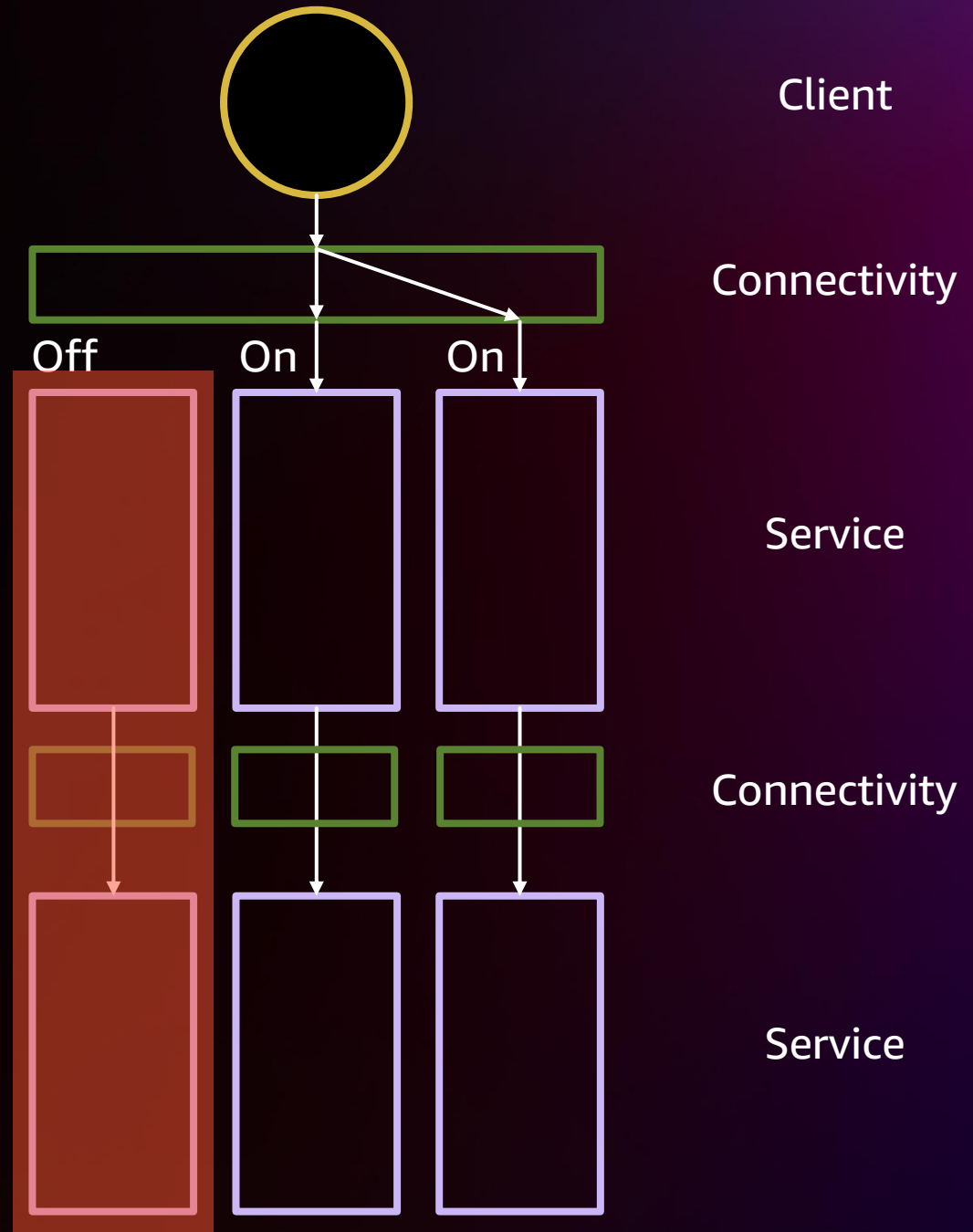
Design for failure

Categories of failure

Code and configuration

Infrastructure

Dependencies



Design for failure

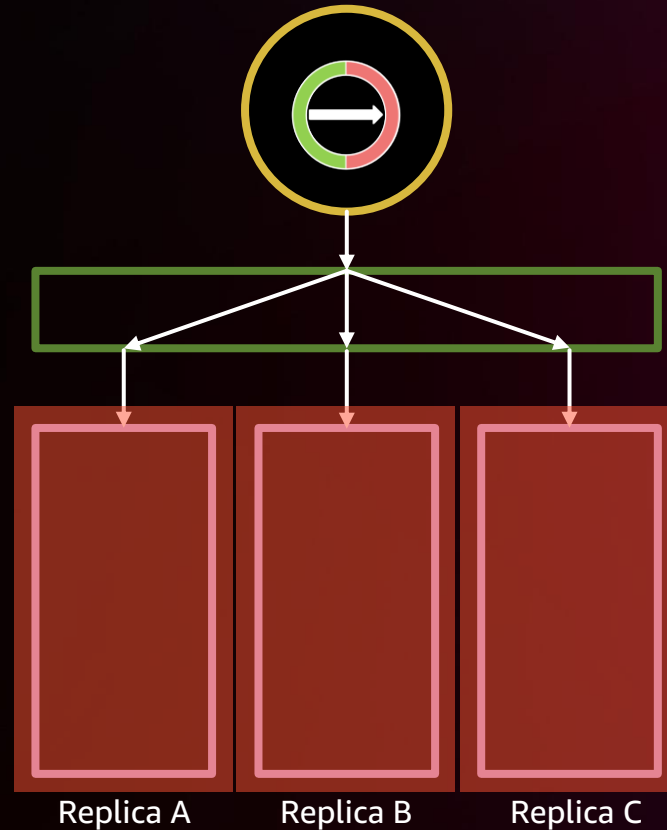
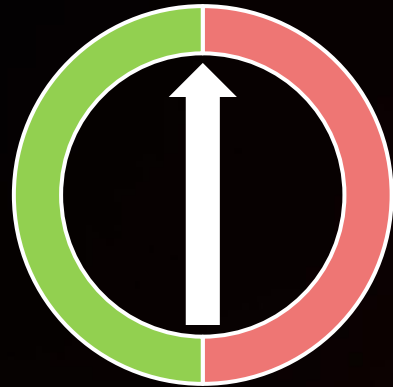
Categories of failure

Code and configuration

Infrastructure

Dependencies

Data and State



Client

Connectivity

Service

Design for failure

SHARD WORKLOADS AND DATA

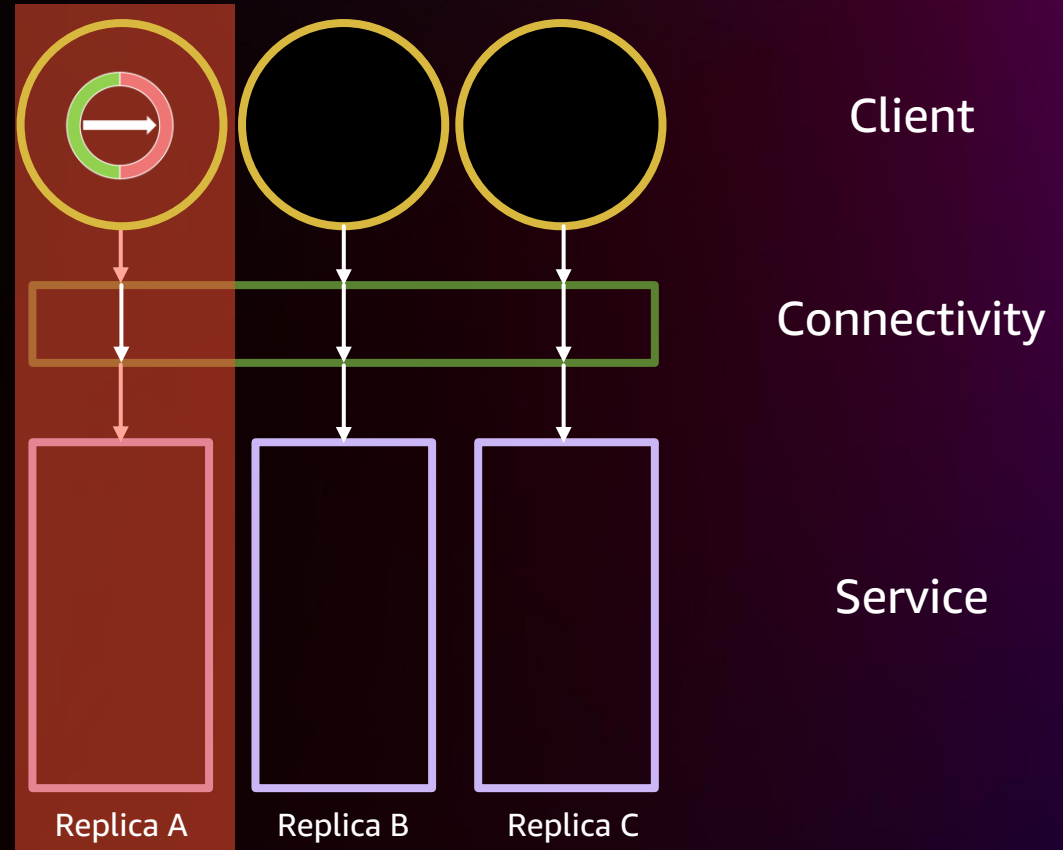
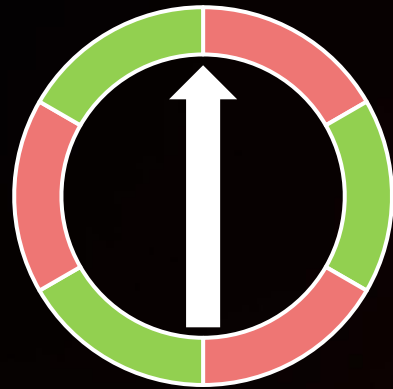
Categories of failure

Code and configuration

Infrastructure

Dependencies

Data and State



Design for failure

SHARD WORKLOADS AND DATA

- **Amazon Builders' Library: Workload isolation using shuffle-sharding**
- **AWS re:Invent 2018: How AWS Minimizes the Blast Radius of Failures (ARC338)**

Design for failure

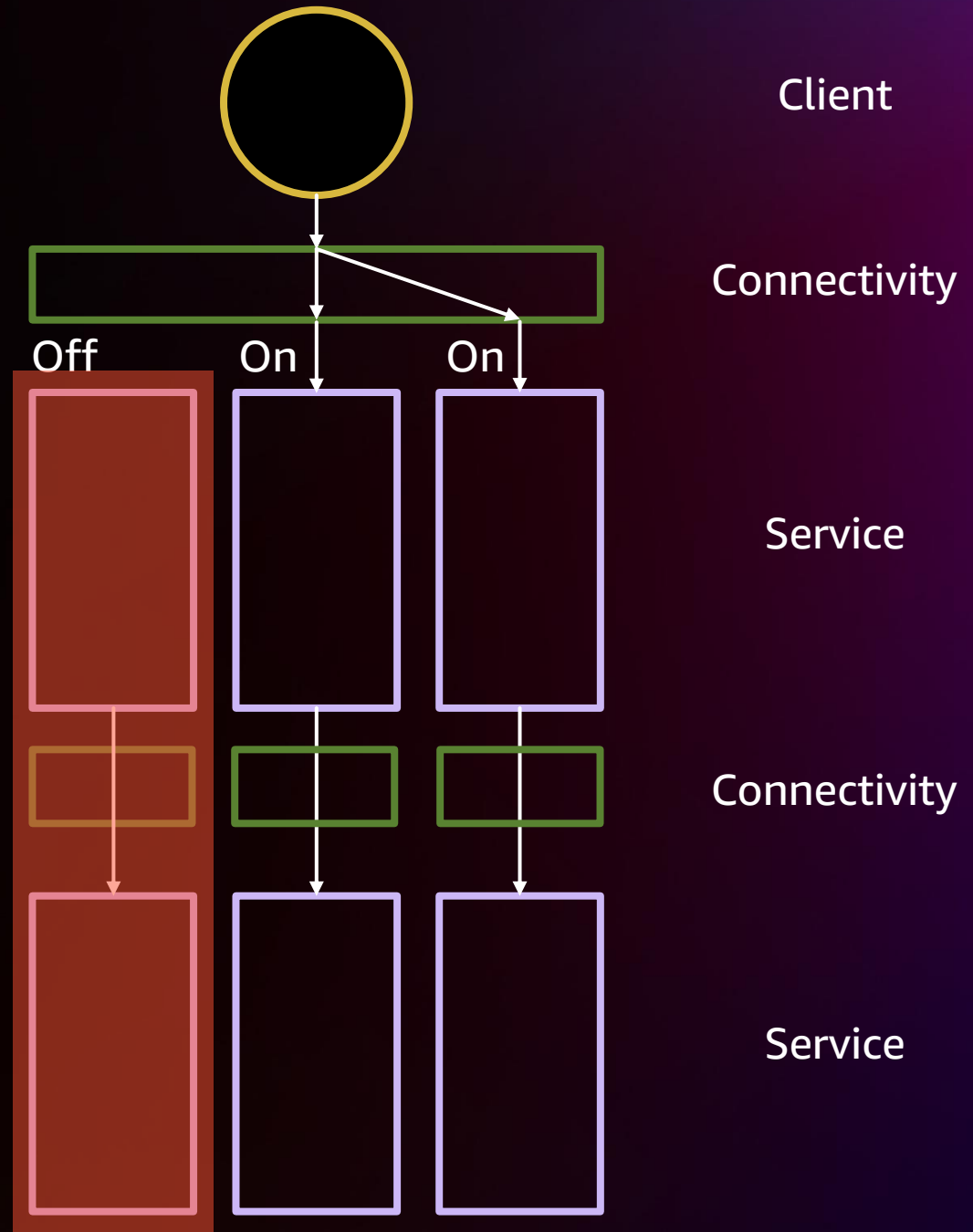
Categories of failure

Code and configuration

Infrastructure

Dependencies

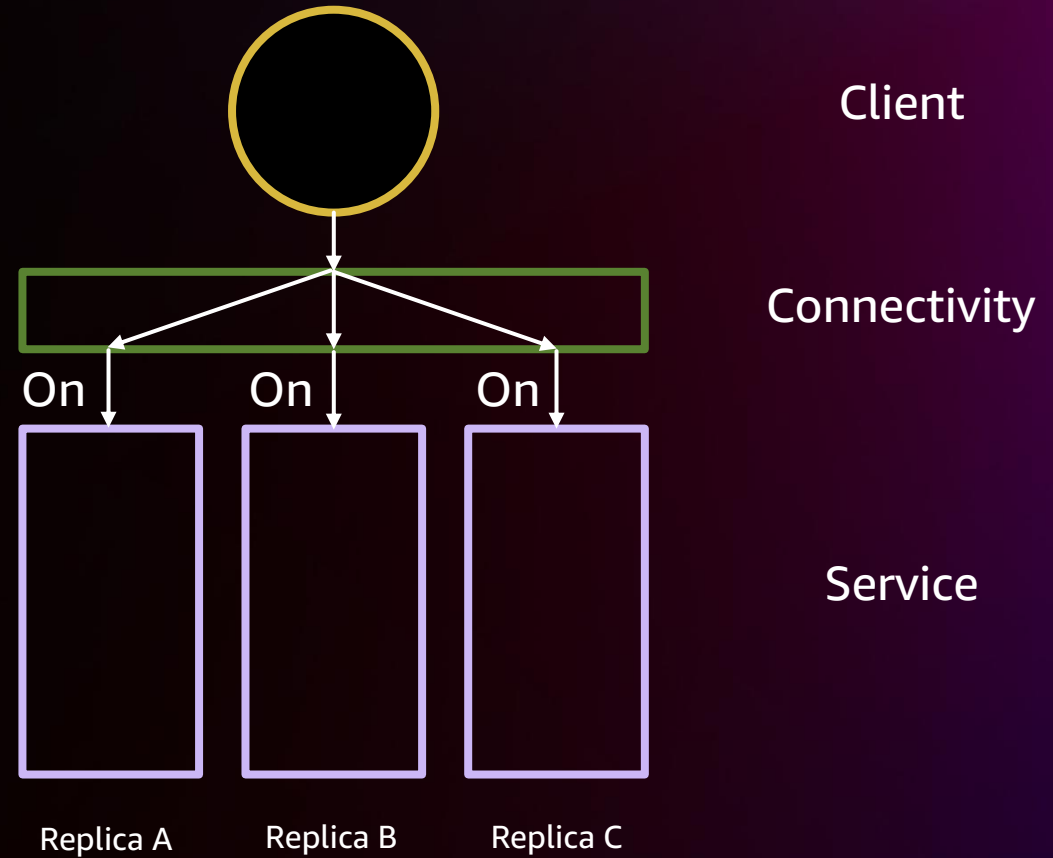
Data and State



Detecting failure

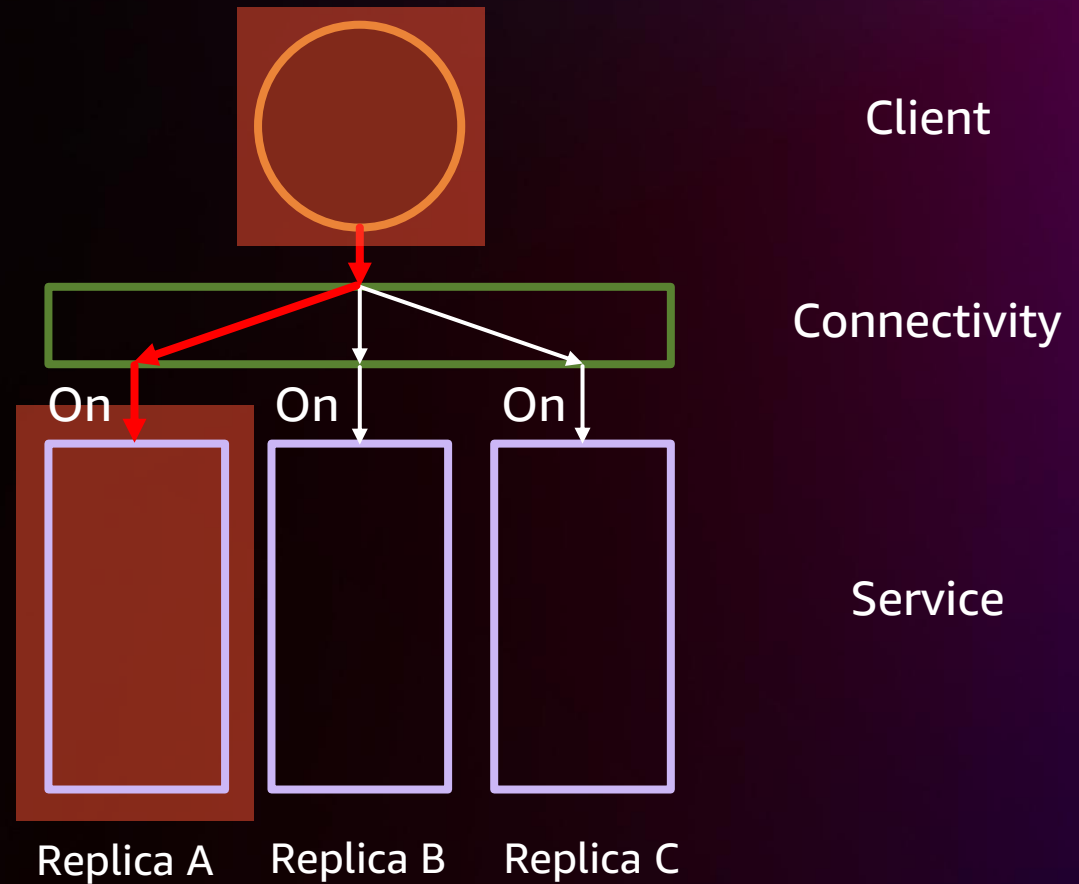
Automatic failure recovery requires automatic failure detection

Correlating specific failures to specific recovery mechanisms is difficult



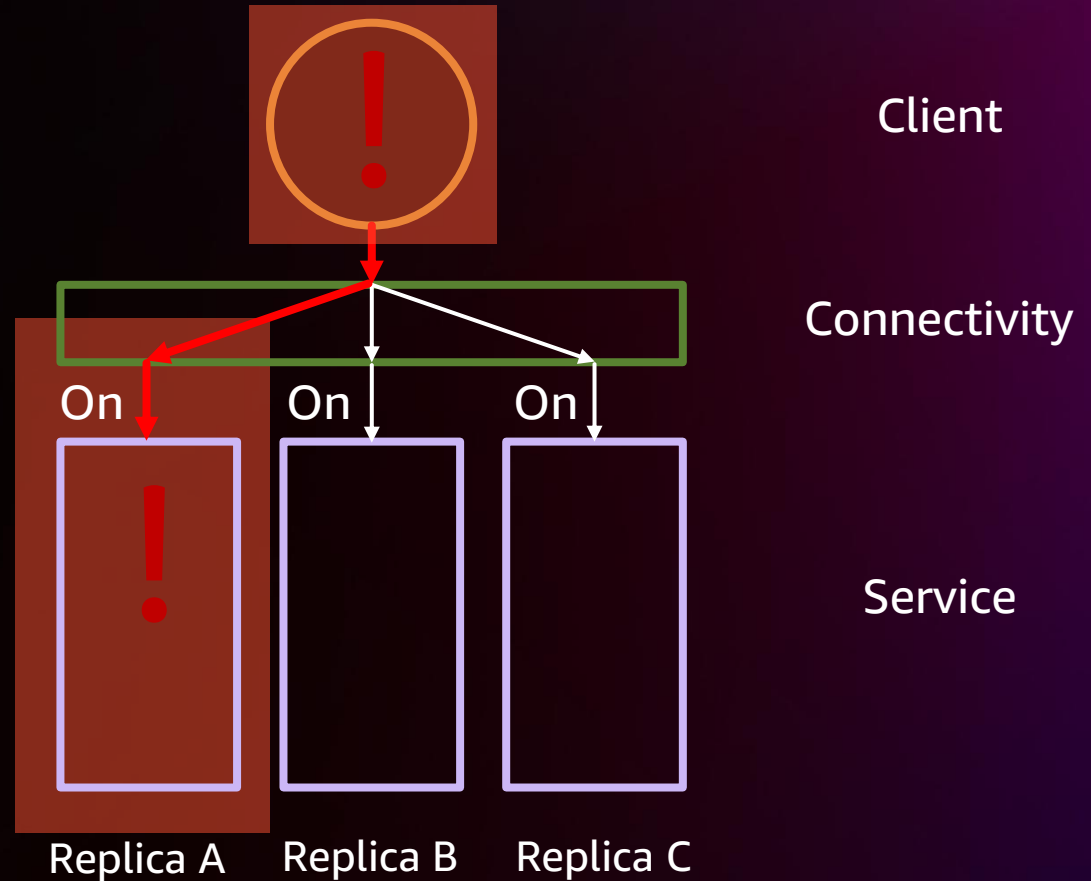
Detecting failure

1. A replica failure impacts clients



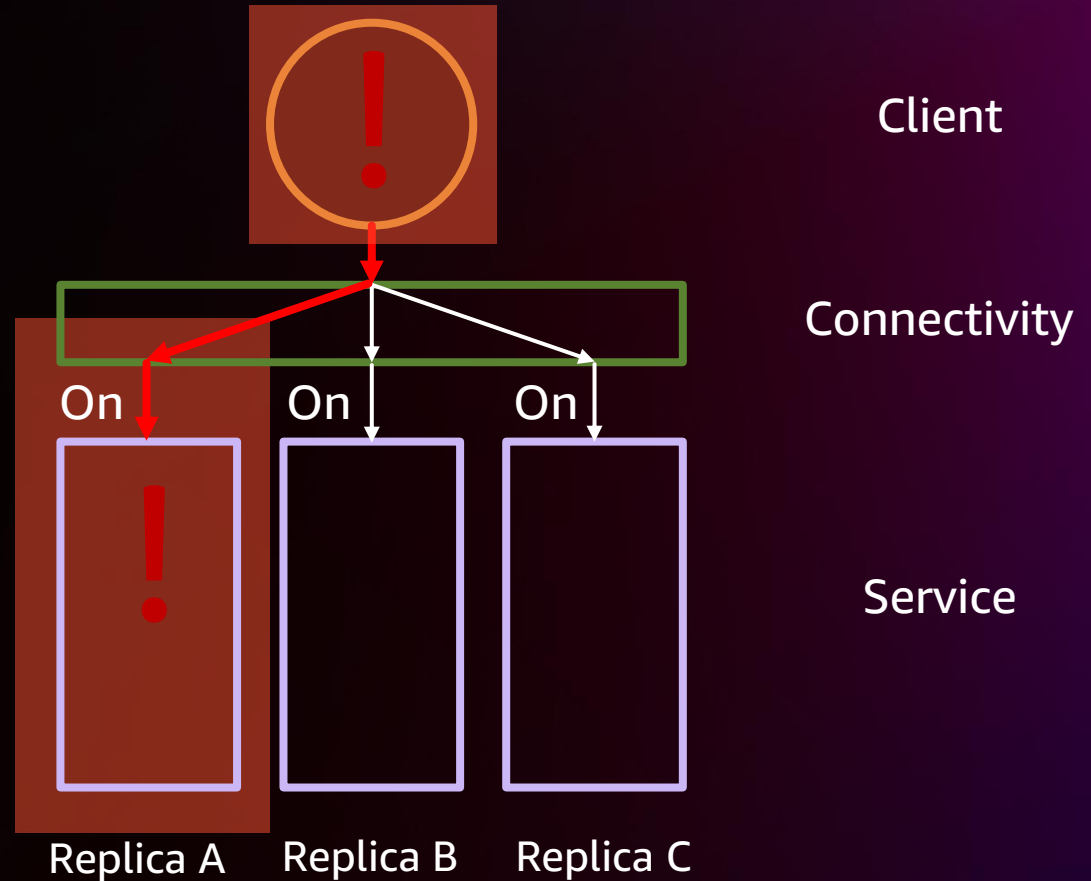
Detecting failure

1. A replica failure impacts clients
2. Application monitors alert:
 - a) Client impact



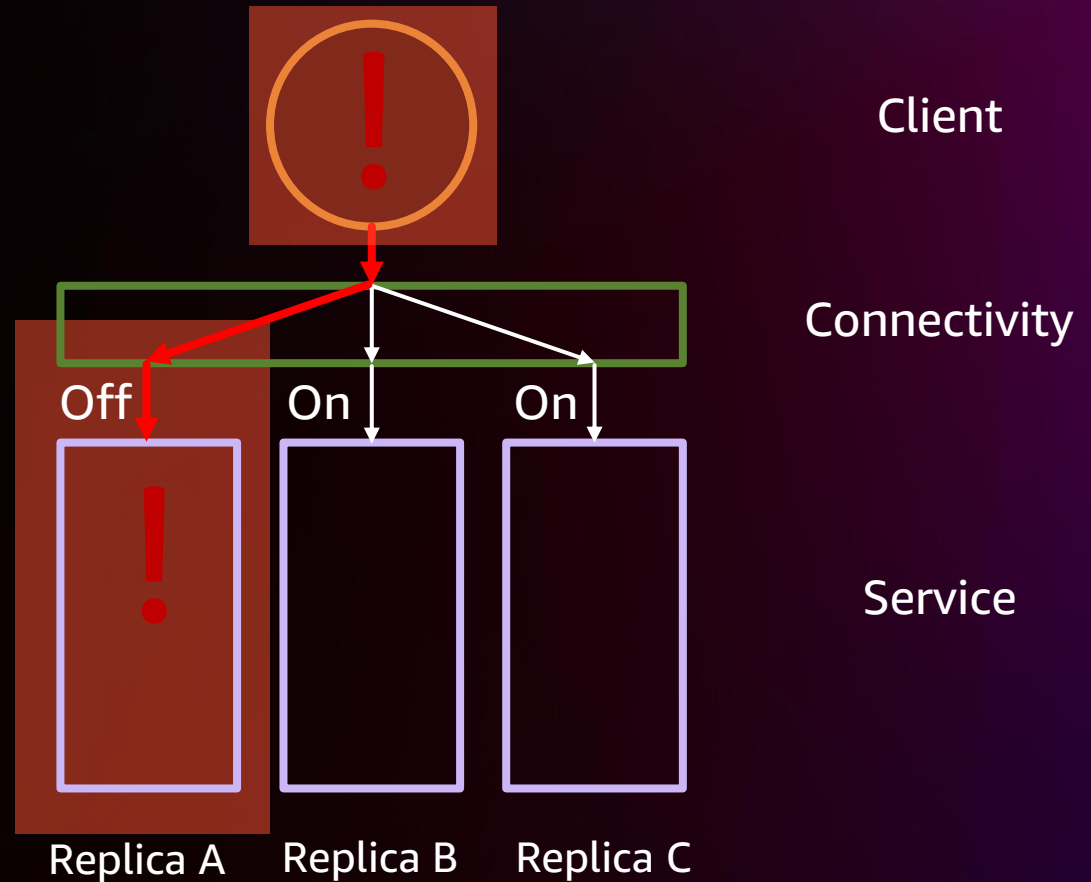
Detecting failure

1. A replica failure impacts clients
2. Application monitors alert:
 - a) Client impact
 - b) Replica A failure



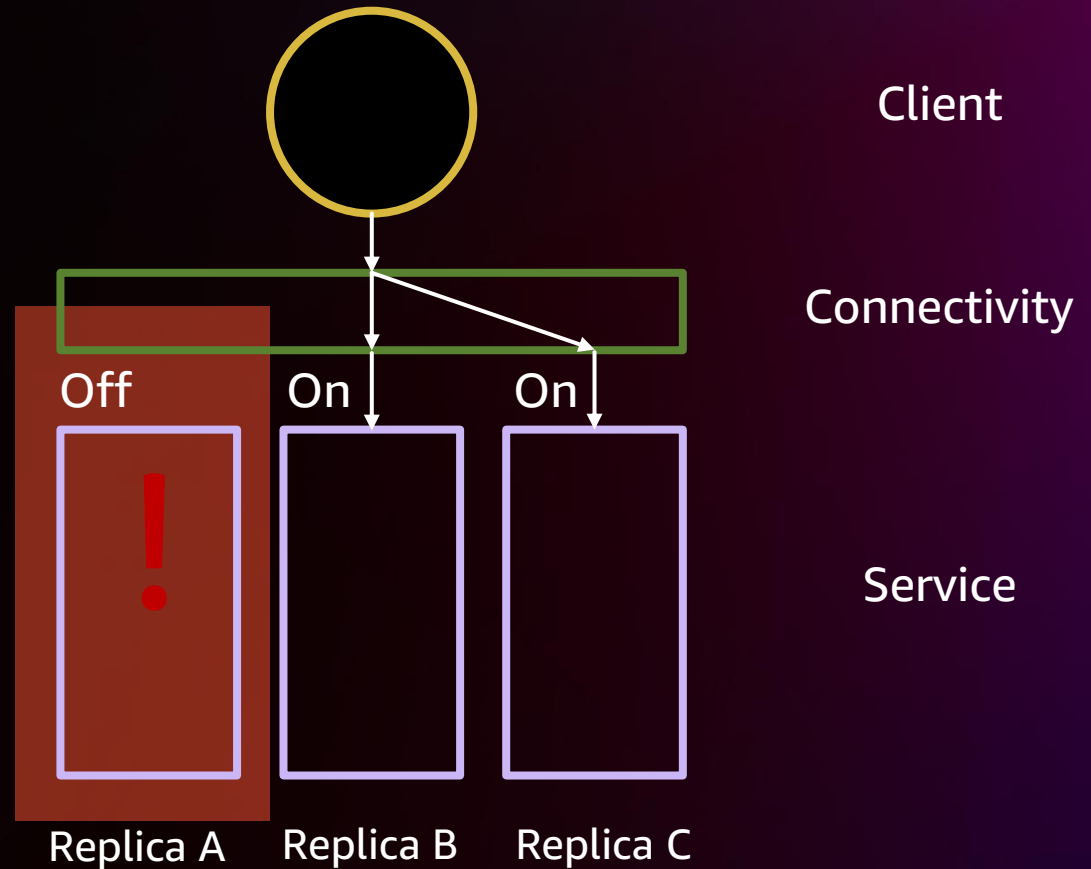
Detecting failure

1. A replica failure impacts clients
2. Application monitors alert:
 - a) Client impact
 - b) Replica A failure
3. Disable Replica A



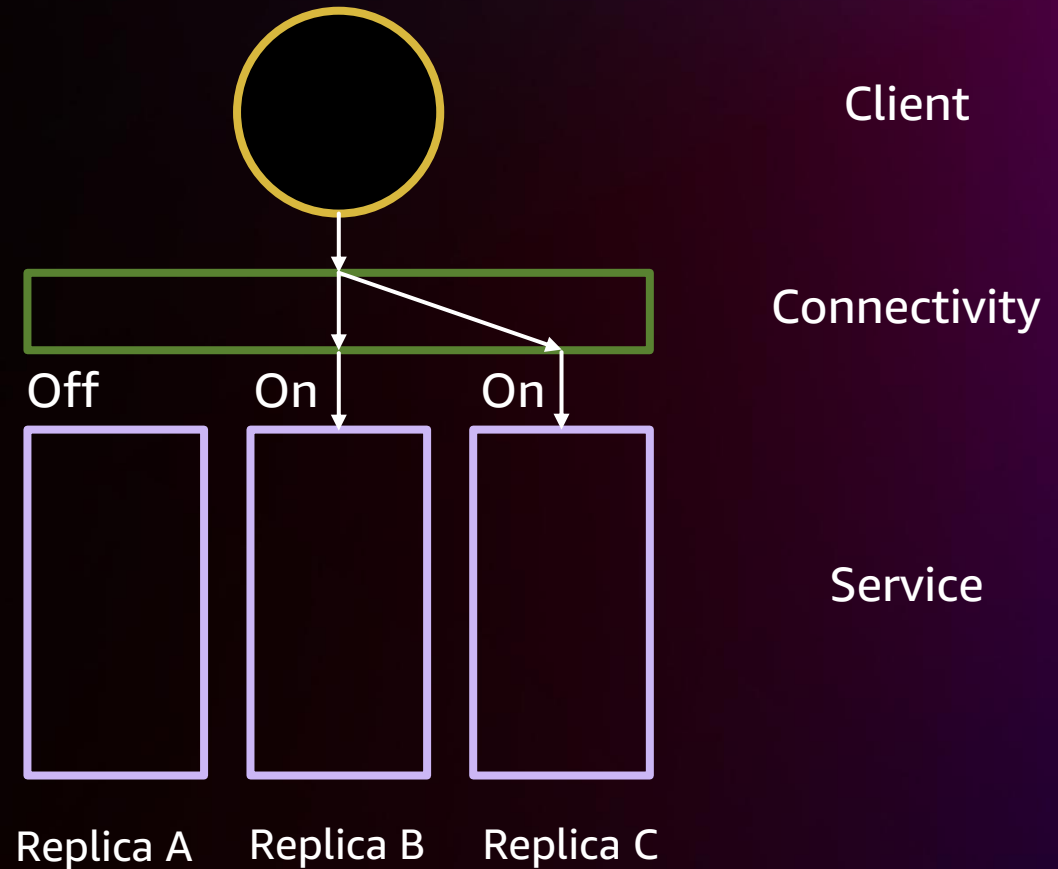
Detecting failure

1. A replica failure impacts clients
2. Application monitors alert:
 - a) Client impact
 - b) Replica A failure
3. Disable Replica A
4. Client impact mitigated; alarm clears



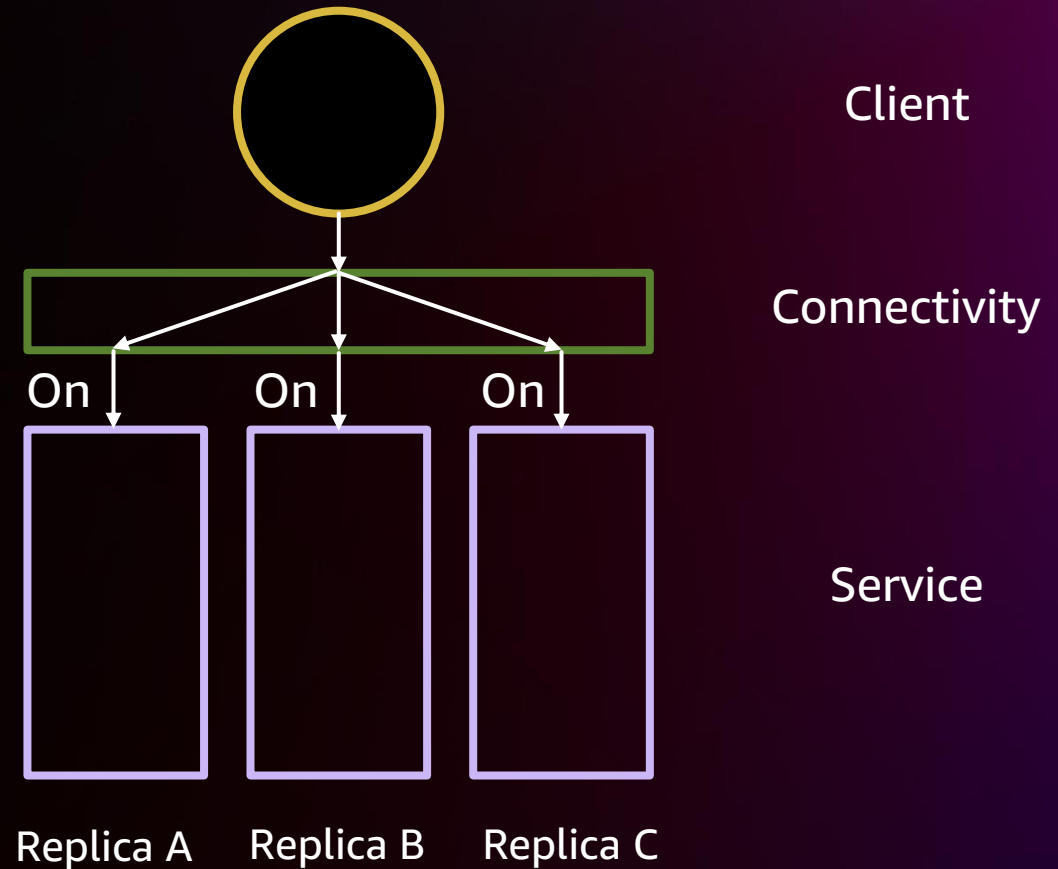
Detecting failure

1. A replica failure impacts clients
2. Application monitors alert:
 - a) Client impact
 - b) Replica A failure
3. Disable Replica A
4. Client impact mitigated, alarm clears
5. Replica A rollback or repair; alarm clears



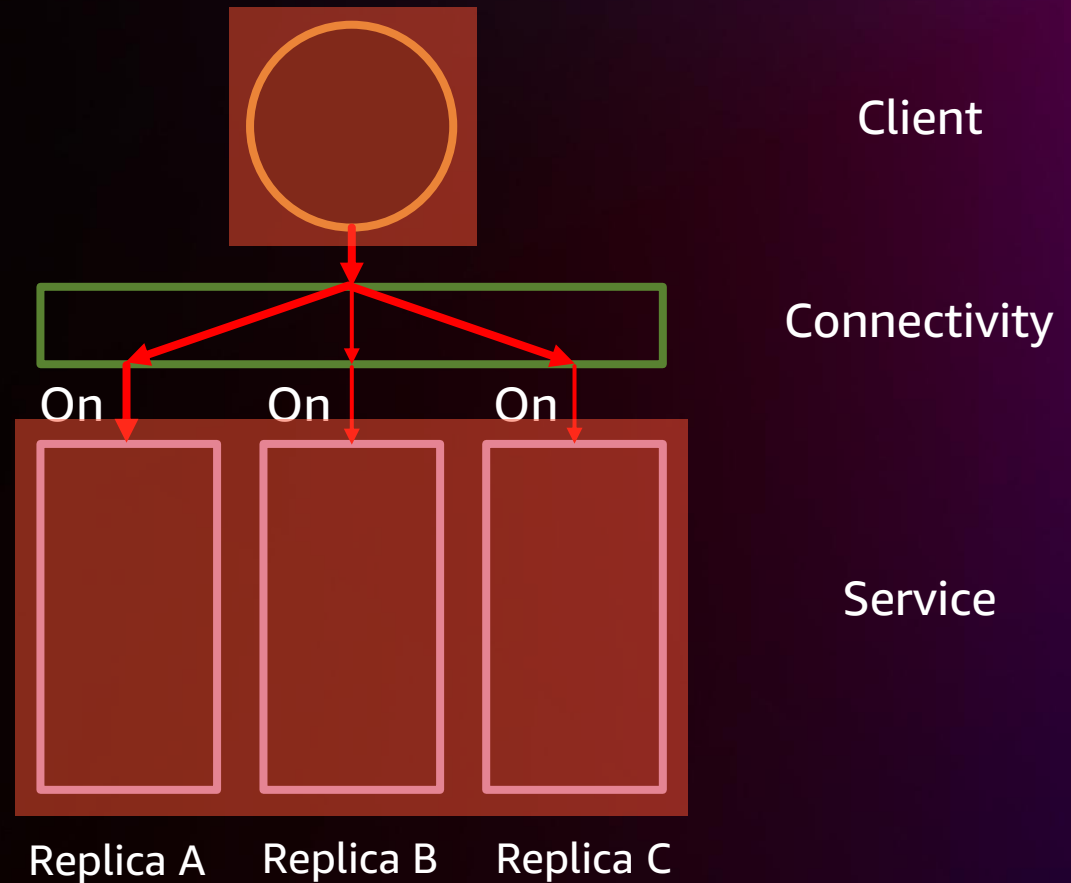
Detecting failure

1. A replica failure impacts clients
2. Application monitors alert:
 - a) Client impact
 - b) Replica A failure
3. Disable Replica A
4. Client impact mitigated, alarm clears
5. Replica A rollback or repair, alarm clears
6. Return Replica A to service



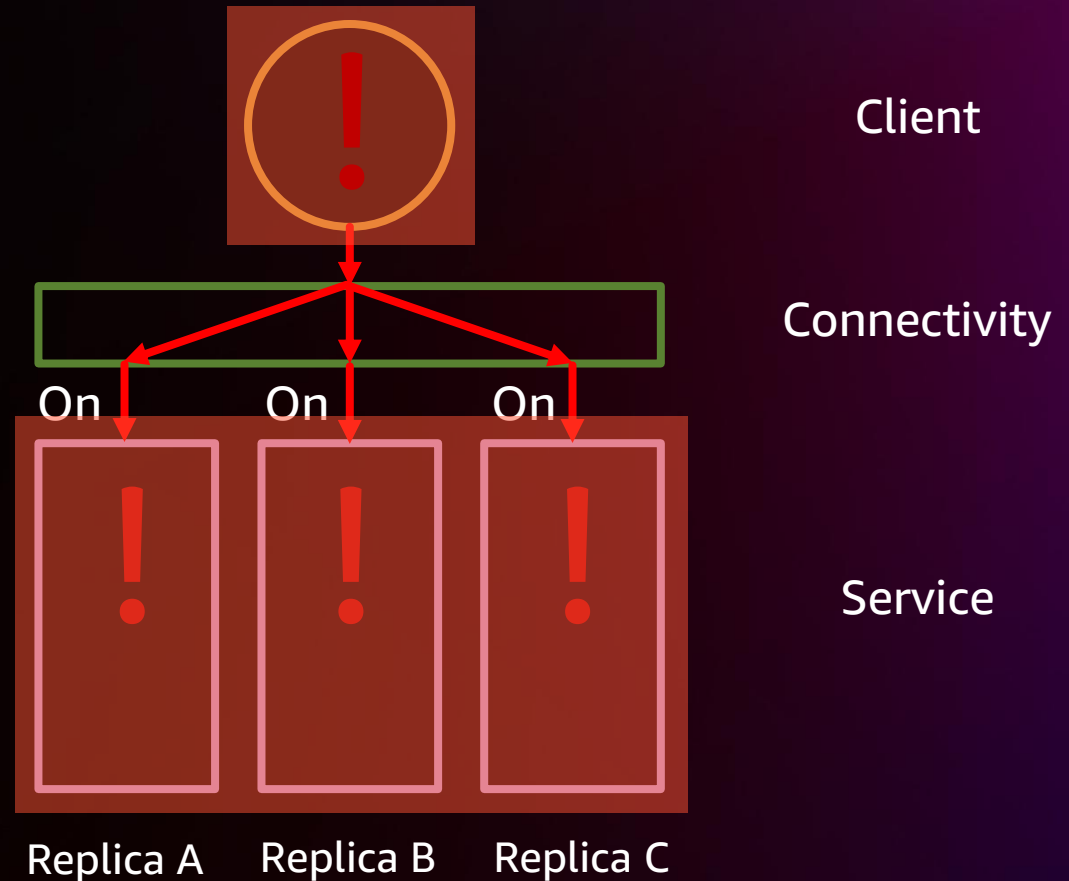
Detecting failure

1. **Multi-replica failure** impacts clients
2. Application monitors alert:
 - a) Client impact
 - b) All replicas impacted



Detecting failure

1. **Multi-replica failure impacts clients**
2. Application monitors alert:
 - a) Client impact
 - b) All replicas impacted
3. Disable change and deploy pipelines
4. Disable automated recovery
5. Page operator



Detecting failure

Synthetic canaries

Detecting failure

Synthetic canaries

- Replicate real customer requests
- Full API coverage
- Target individual replicas

Detecting failure

Monitor	Alarm threshold (automated detection)
Synthetic canaries	2 consecutive or 2 of last 10
Faults/5xx	>1% or >10 in 1 min
Errors/4xx	>75%
Request volume/2xx	>90% drop
Latency	10x @ 90th percentile

Detecting failure

Monitor	Alarm threshold (automated detection)
Synthetic canaries	2 consecutive or 2 of last 10
Faults/5xx	>1% or >10 in 1 min
Errors/4xx	>75%
Request volume/2xx	>90% drop
Latency	10x @ 90th percentile

Per-replica metrics **and**
whole-service metrics

Recovering from failure

Amazon Route 53 Application Recovery Controller



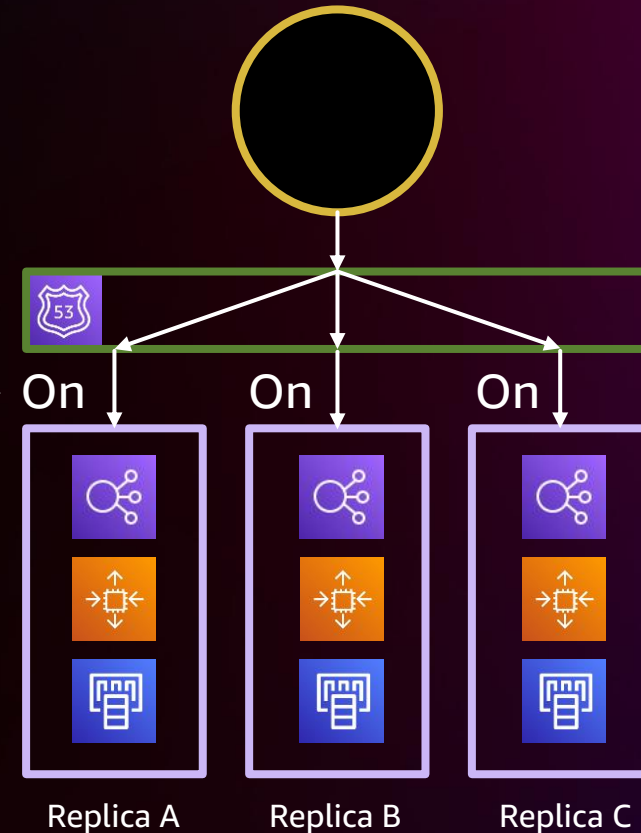
Route 53 Application Recovery Controller

ROUTING CONTROLS

Routing controls (3) Add routing control Change routing control

< 1

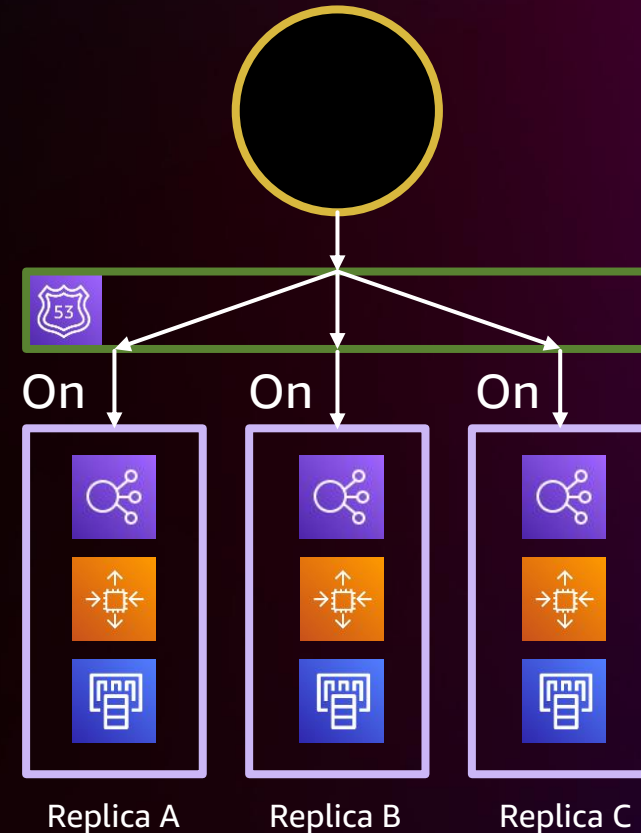
<input type="checkbox"/>	Name	Routing control state	Routing control ARN
<input type="checkbox"/>	resilienceAppDemo_Replica_C_DNS	✓ On	arn:aws:route53-recovery-con
<input type="checkbox"/>	resilienceAppDemo_Replica_B_DNS	✓ On	arn:aws:route53-recovery-con
<input type="checkbox"/>	resilienceAppDemo_Replica_A_DNS	✓ On	arn:aws:route53-recovery-con



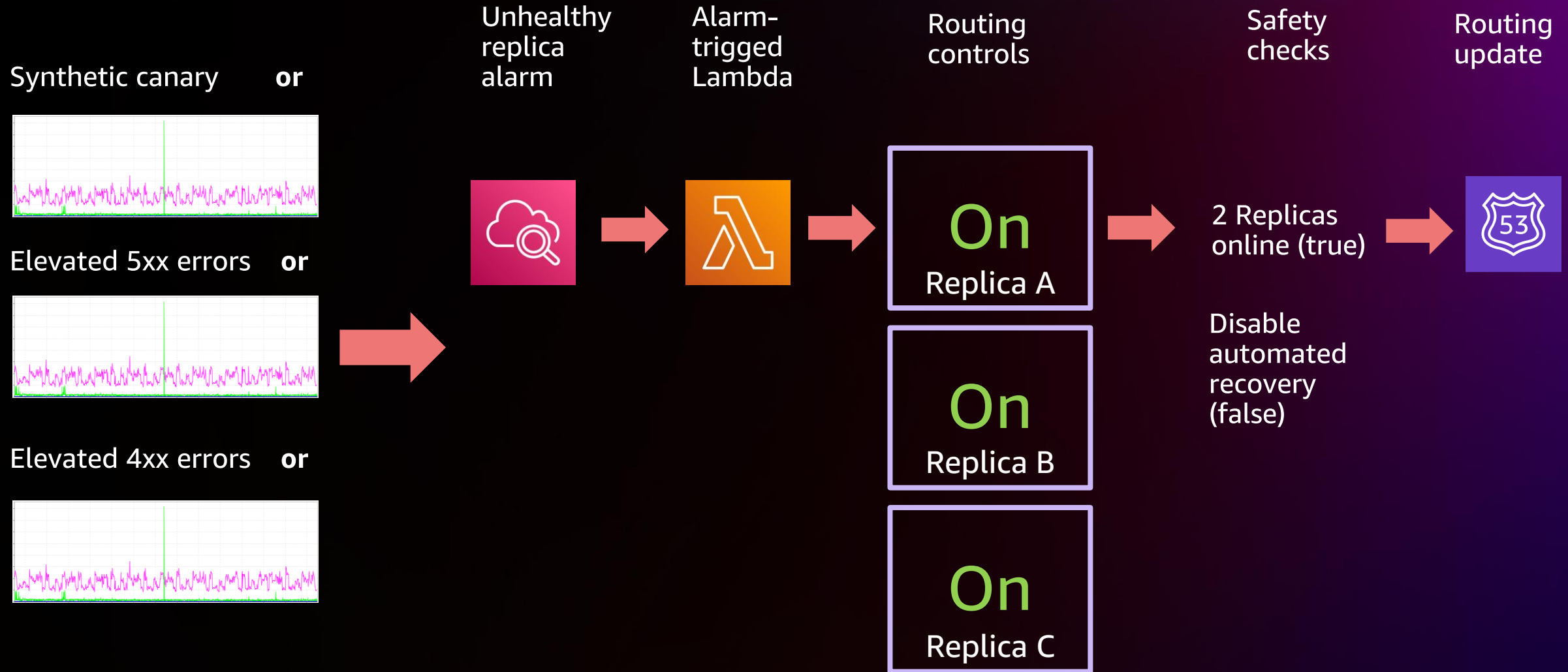
Route 53 Application Recovery Controller

SAFETY RULES

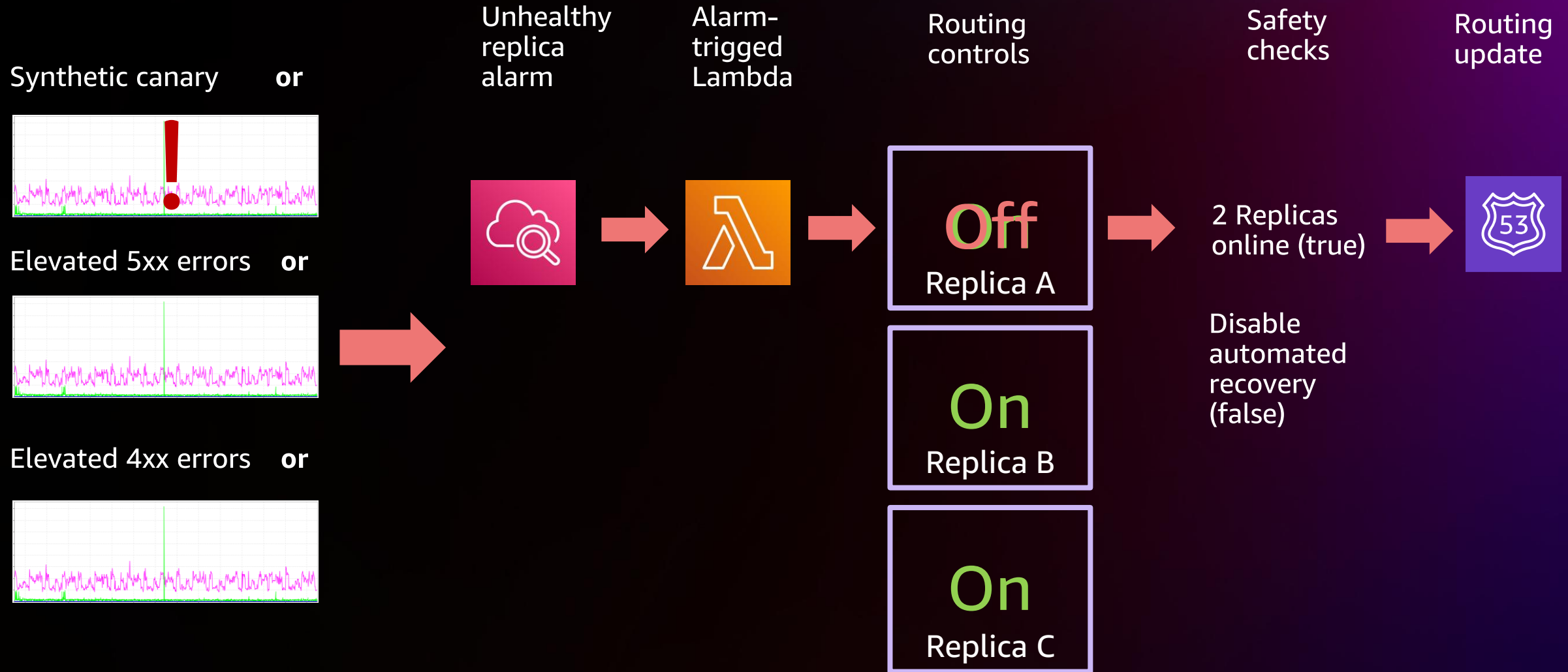
Safety rules (2)	
<input type="text" value="Filter resources by property or value"/>	
Name ▼	Type ▼
Disable_Automated_Recovery	Gating
2_Replicas_Online	Assertion



Automated recovery in under 3 minutes




Automated recovery in under 3 minutes



Route 53 Application Recovery Controller



READINESS CHECKS

ResilienceReinventDemo





Action ▼




Recovery group details

<div>Readiness status</div> <div>Info</div> <div> Ready</div> <div>Last checked 10/13/2021 2:44:42 PM</div>	<div>Recovery group name</div> <div>ResilienceReinventDemo</div>	<div>ARN</div> <div></div> <div>arn:aws:route53-recovery-readiness::553409351405:recovery-group/ResilienceReinventDemo</div>
---	--	---

Cells (3)

 *Filter resources by property or value*

< 1 > 

Cell name ▲	Cell status ▼
Replica_A	 Ready
Replica_B	 Ready
Replica_C	 Ready

Let's chalk and talk



Recap and next steps

Expect failure and design for it

Partition your app and dependencies to limit blast radius

Automate recovery controls and test them regularly

Target individual partitions with deep automated metrics

Take a look at Route 53 Application Recovery Controller

Team Sessions

Please check out other sessions by our team

ARC304 Build for resilience using Amazon Route 53 Application Recovery Controller

ARC322 Networking lens on resiliency architectures

ARC306 Multi-region patterns and best practices

ARC329 Operating highly available Multi-AZ applications



Thank you!

James Lamanna
lamannaj@amazon.com

Steven Hooper
hoopsta@amazon.co.uk



Please complete the session
survey in the **mobile app**

