AWS re:Invent

ARC411-R

# Reducing blast radius with cell-based architectures

**Vignesh Janakiraman**
Principal Software Engineer
Amazon.com

**Karthik Kumar Odapally**
Senior Solutions Architect
Amazon Web Services

AWS re:Invent



aws

# Agenda

- Why cellular?
- Fundamental concepts:
  - Control plane vs. data plane
  - Cell properties; router properties
- AWS Well-Architected Framework
- Design tenets
- Customer use case
  - Architecture
  - Routing rules
- Key takeaways

# Cell-based architecture

AWS re:Invent



aws

# Why cellular?

Contain the failures

Scale millions of TPS

Auto-healing

Pay for what you use

# What is a cell?

Fixed amount of resources


Well-defined function

No calling other cells

Will not expand

Diverse

# Fundamentals: Control plane and data plane

| Service | Control plane | Data plane |
|---|---|---|
| Amazon DynamoDB | DescribeTable API | Query API |
| Amazon EC2 | RunInstances API | A running EC2 instance |
| AWS Lambda | CreateFunction API | Invoke API |

- Control plane: administration of resources
- Data plane: usage of resources
- Separating control plane from data plane reduces blast radius

# Fundamentals: Service/system properties

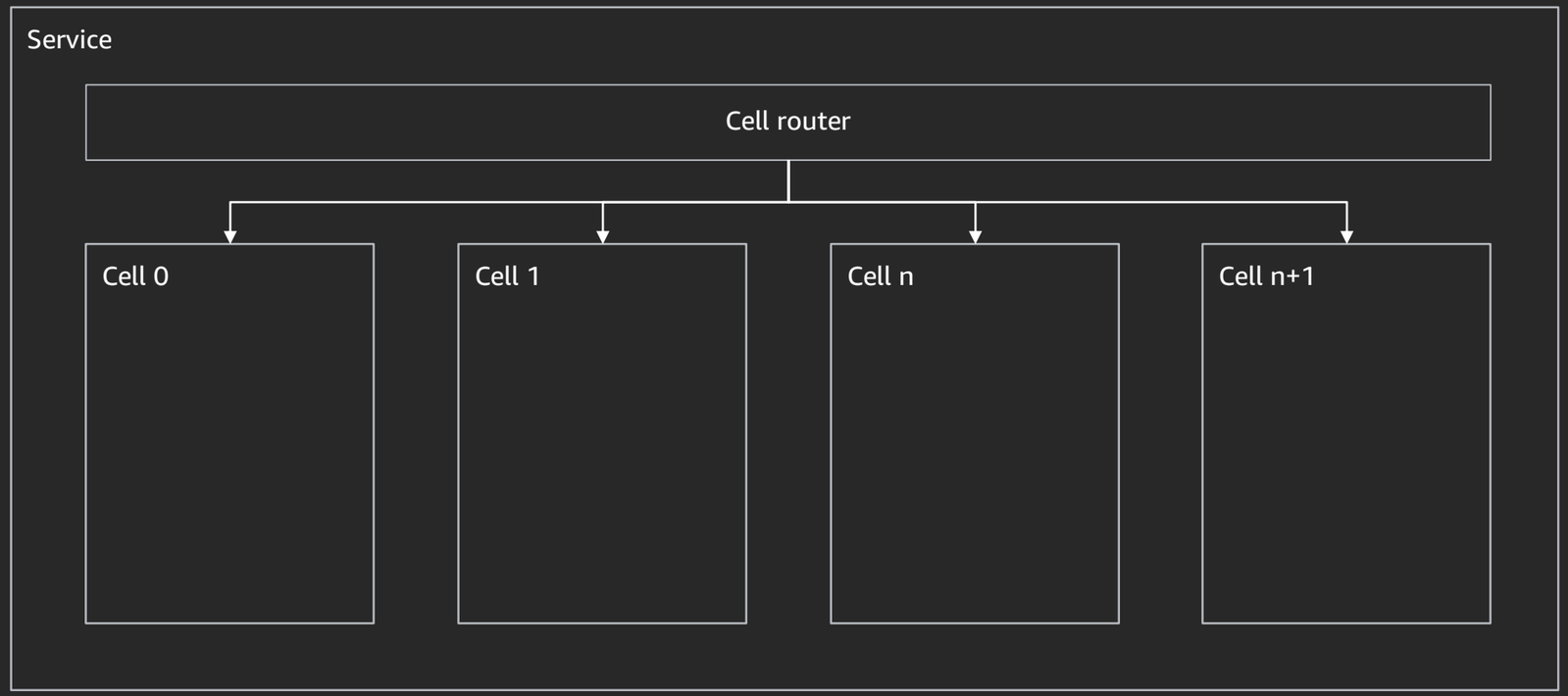
Workload isolation

Failure containment

Scale out versus scale up

Testability

Manageability



# Fundamentals: Small versus large cells

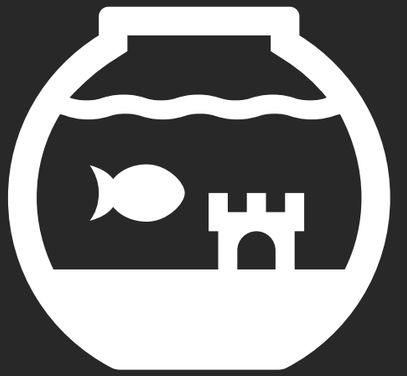
Smaller cells

- Reduced blast radius
- Easier to test
- Cells easier to operate

Larger cells

- Cost efficiency
- Reduced splits
- System easier to operate

versus



# Fundamentals: Router properties

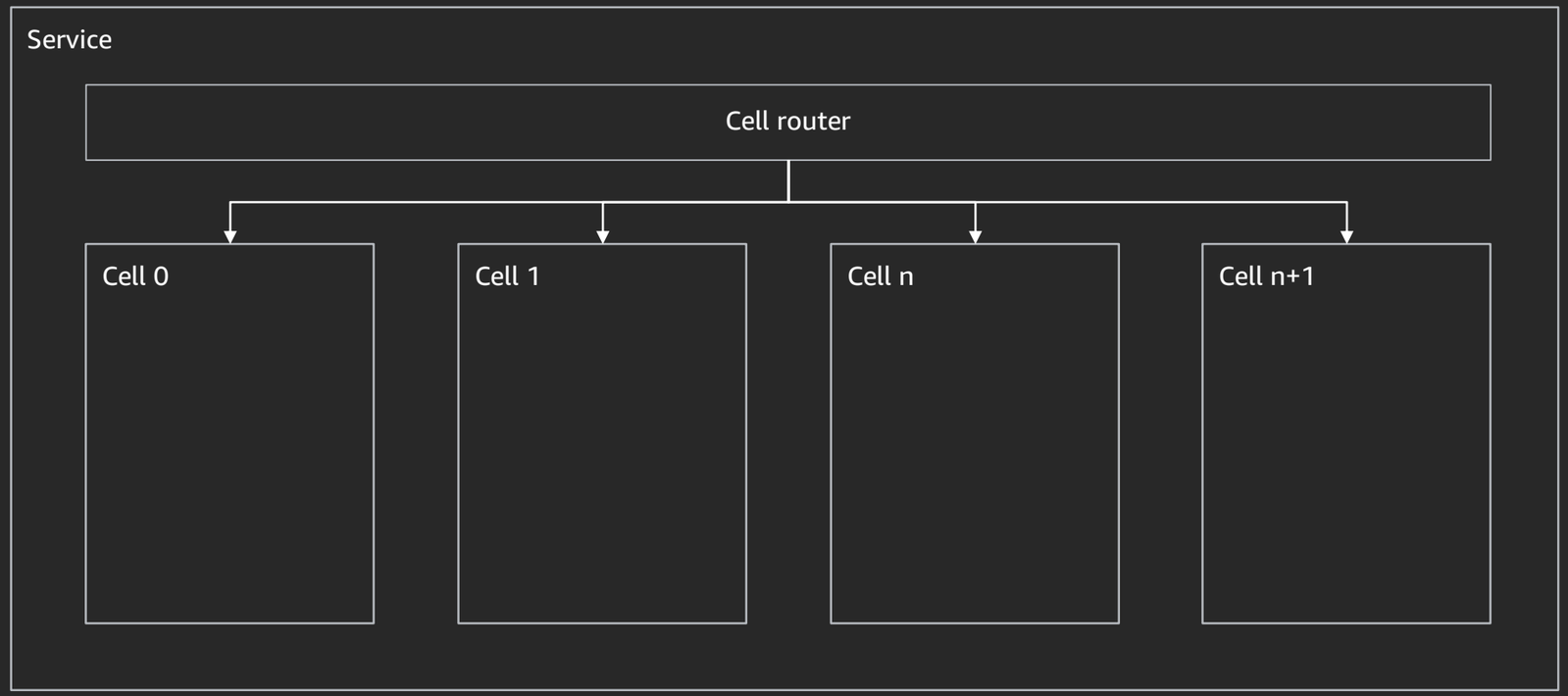
Routing policies

Stress testing

Battle hardening

Keep it simple!

“Thinnest possible layer”



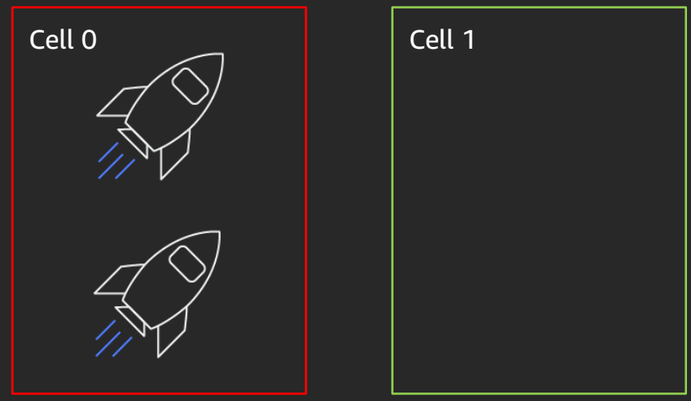
# Fundamentals: Cell State

Relocation of transactions

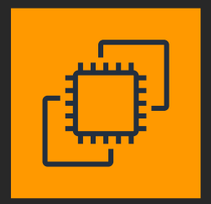
Driven by

- Heat management
- Size balancing
- Scale-out

Careful coordination between router and cells



# Zonal services versus serverless



Amazon Elastic Compute Cloud (Amazon EC2)

Amazon Elastic Block Store (Amazon EBS)

Amazon EMR

AWS CloudHSM

NAT gateway

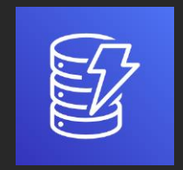
# Zonal services versus serverless

AWS Lambda

Amazon Elastic Kubernetes Service (Amazon EKS)

AWS Fargate

Amazon Aurora Serverless



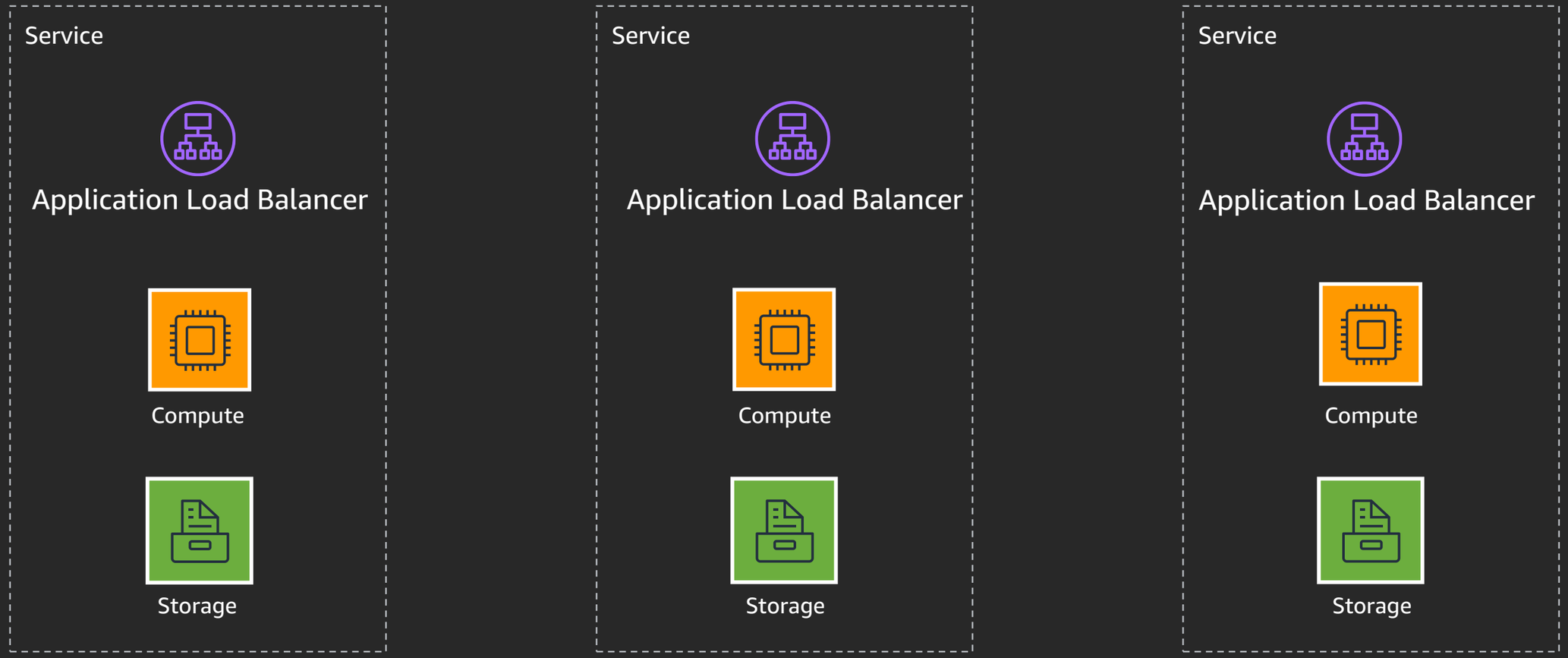
Amazon DynamoDB

Amazon Simple Storage Service (Amazon S3)

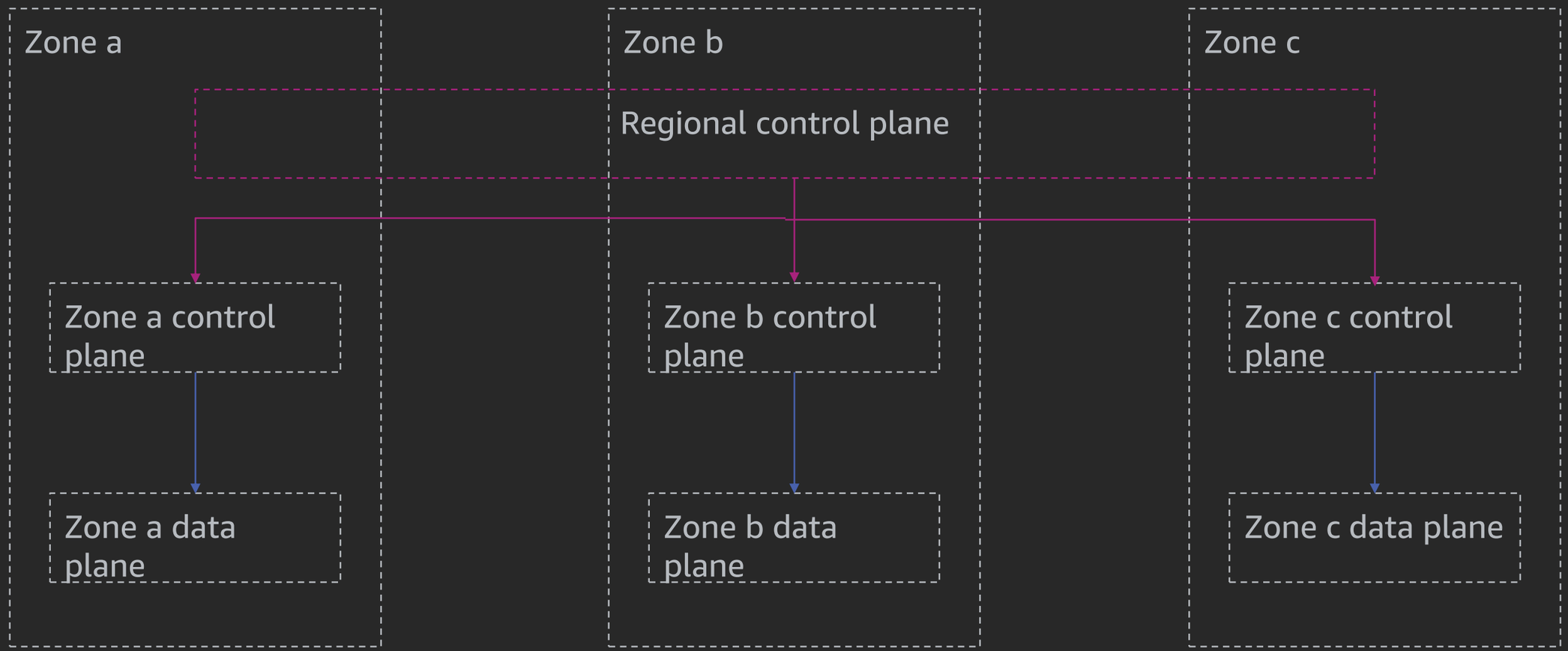
Elastic Load Balancing

Amazon Kinesis

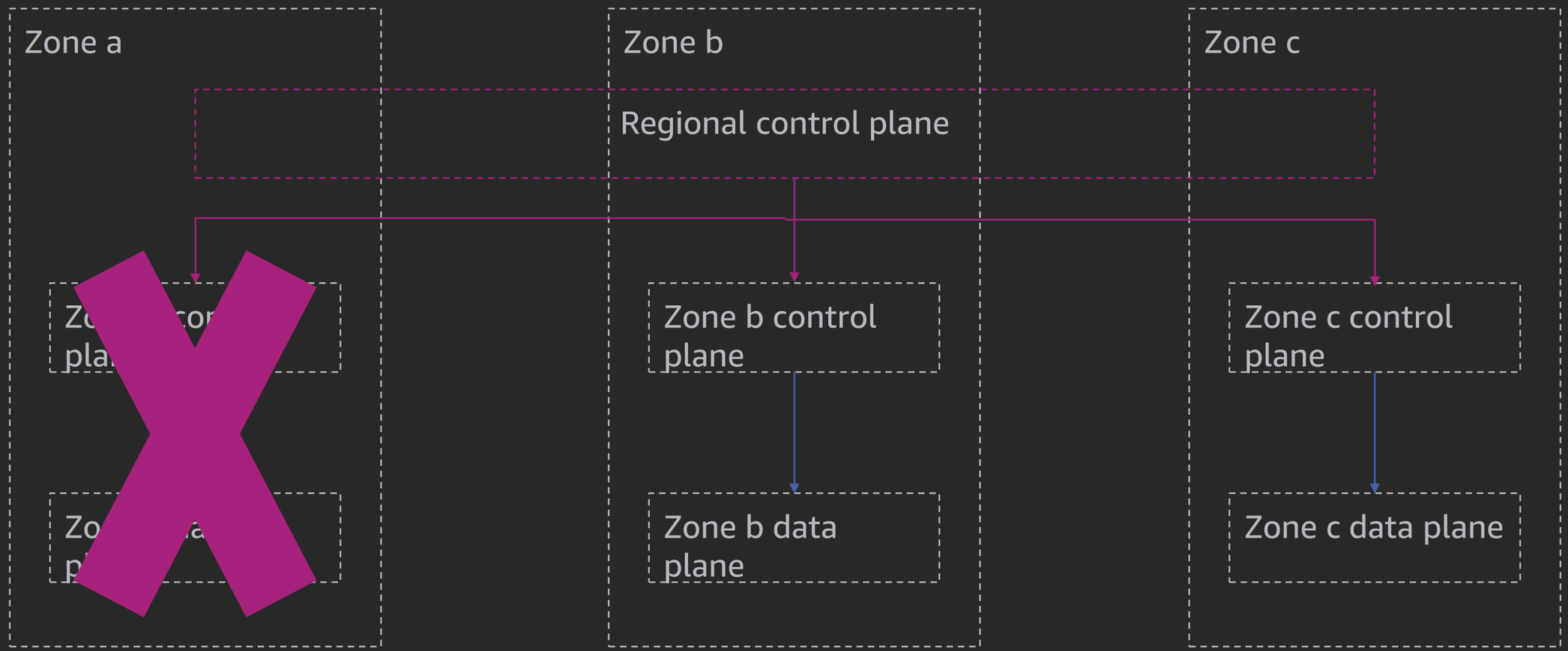
# Traditional service diagram



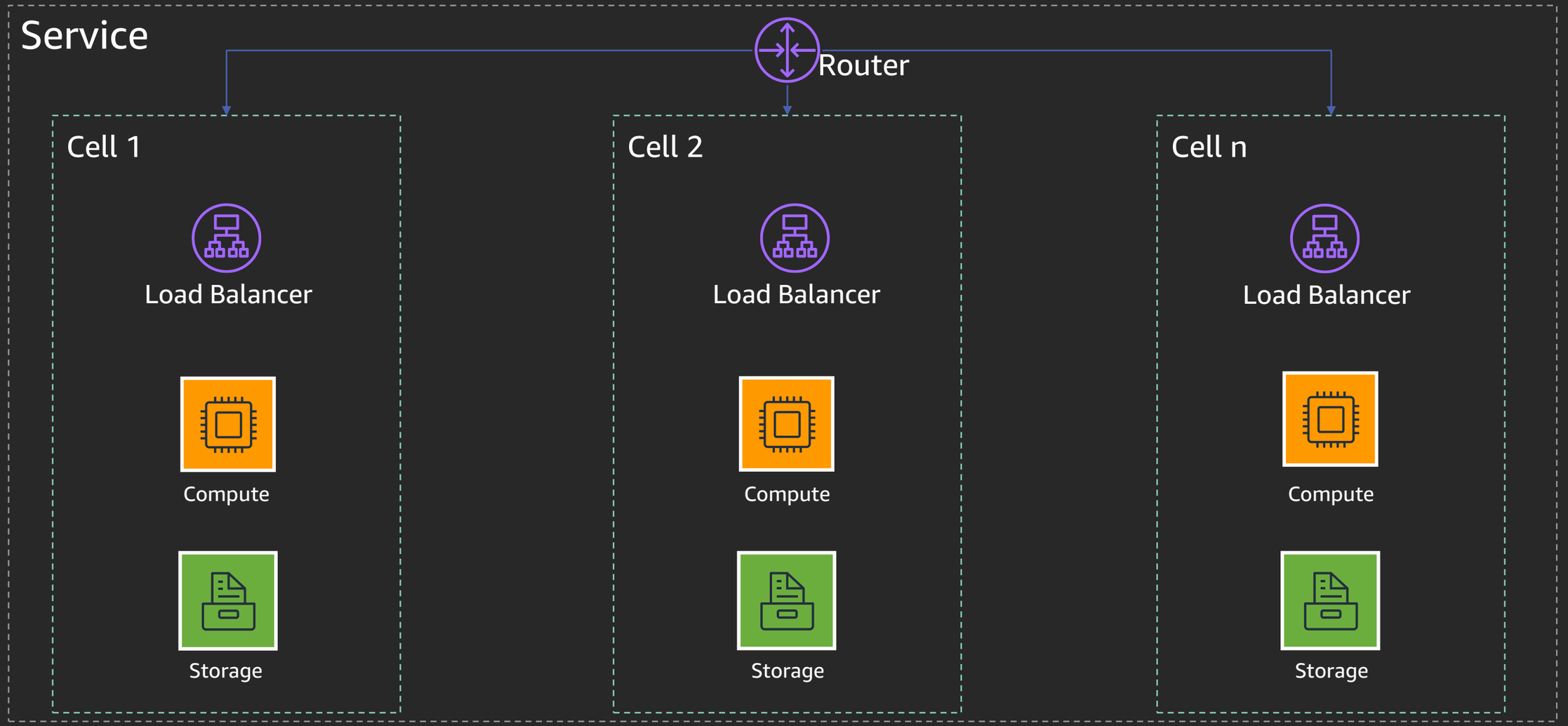
# Zone-based service diagram



# Availability Zone outage



# Cell-based service – AZ agnostic



# Well-Architected Framework

Operational excellence

Security

Reliability

Performance efficiency

Cost optimization

Review process

Consistent

Technology portfolio

# Customer use case

# Using the cellular principles

- Amazon.com, Information Security
- Conduit – AWS account management for internal services
- ASAP – Detect and remediate AWS infrastructure

# Security event data management system

Expanding platform scope to application logs

Real-time detection and remediation

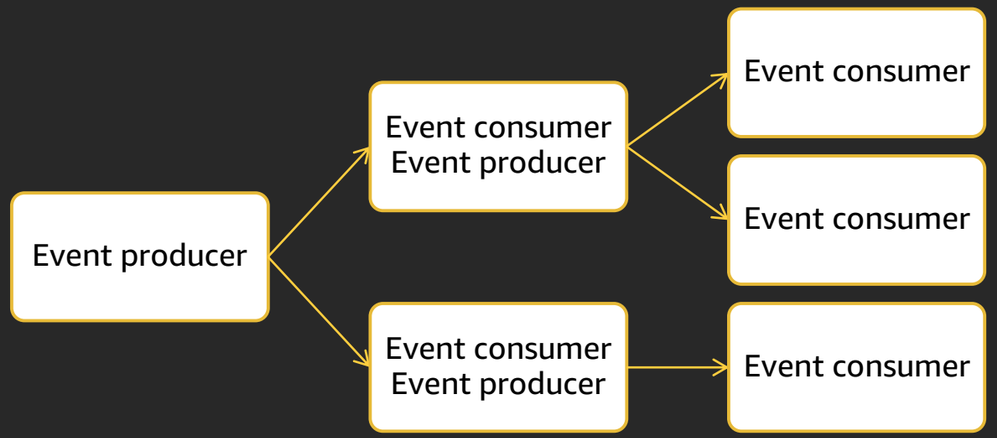
Faster security investigations

# Scale of data

- Growth of current AWS infrastructure
- Onboarding application logs for 75 new services by 2020
- 28 petabytes in 2019 to 90 petabytes in 2020

# Tenets

Minimize blast radius



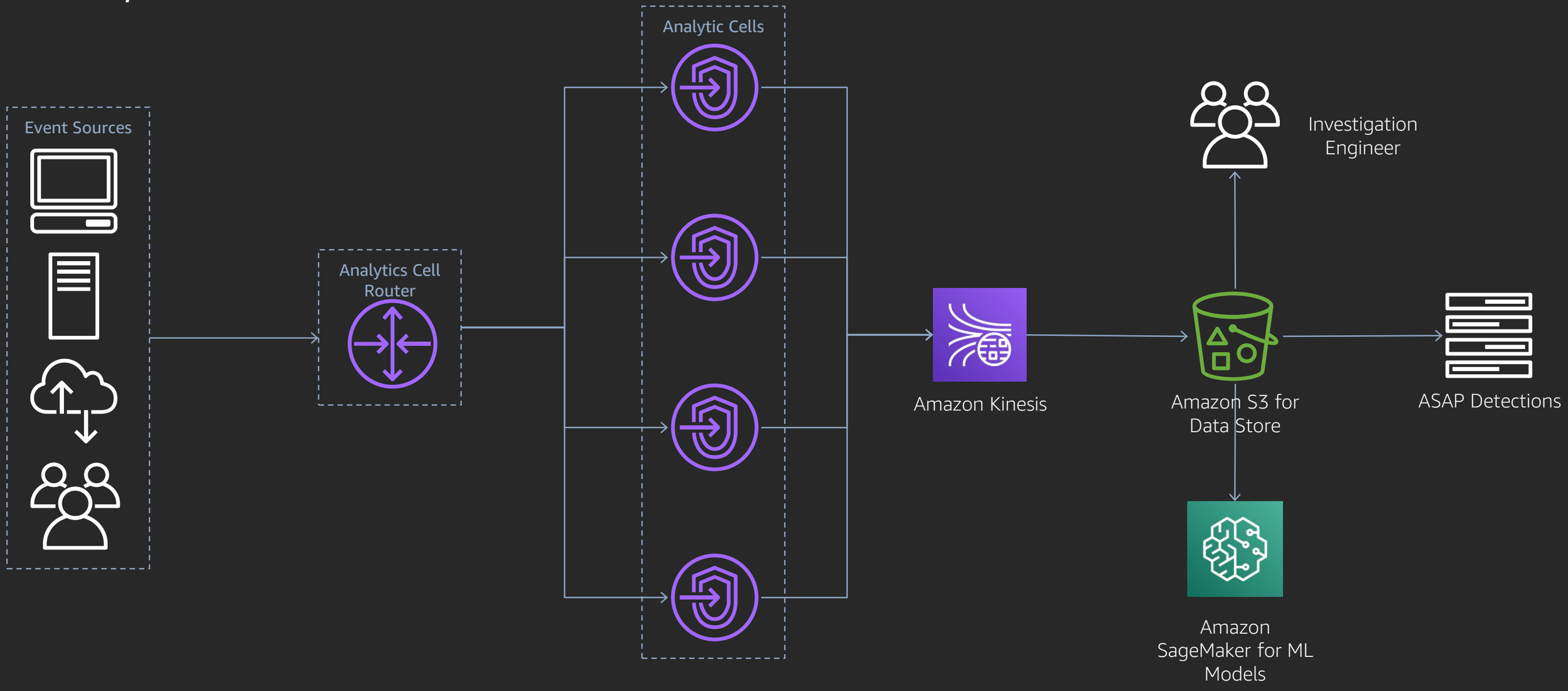

Event-driven architecture

# Tenets

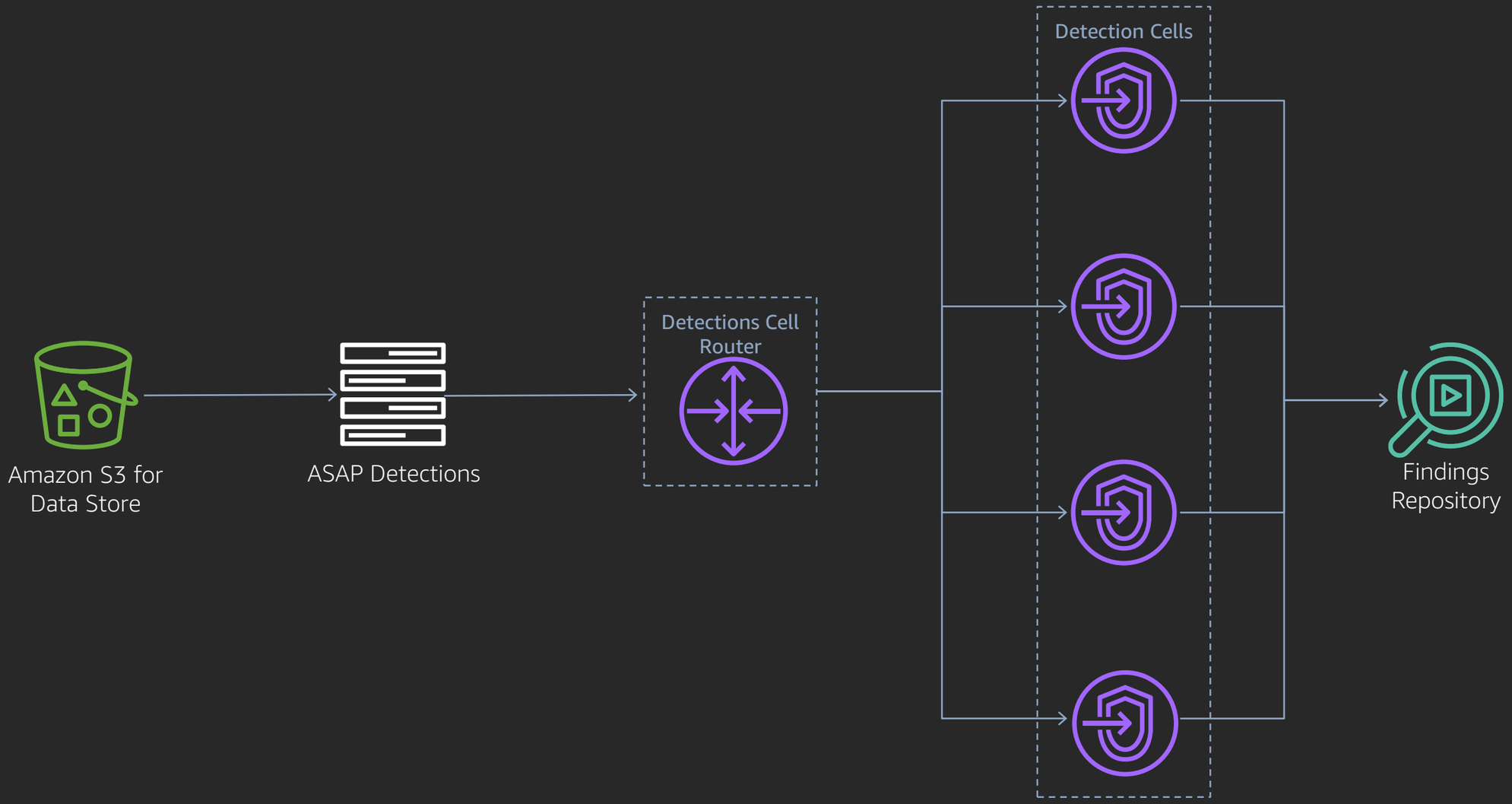
Serverless application

Infrastructure as code

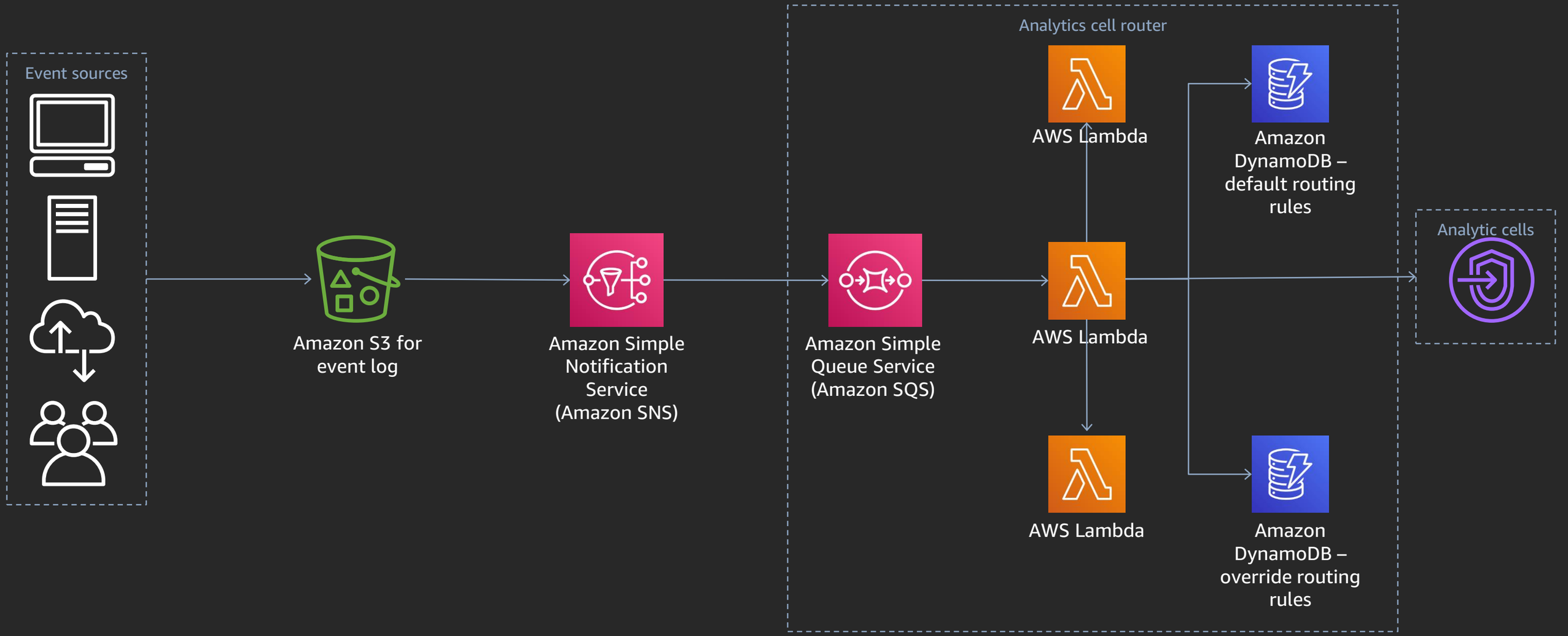
# 10,000-foot view



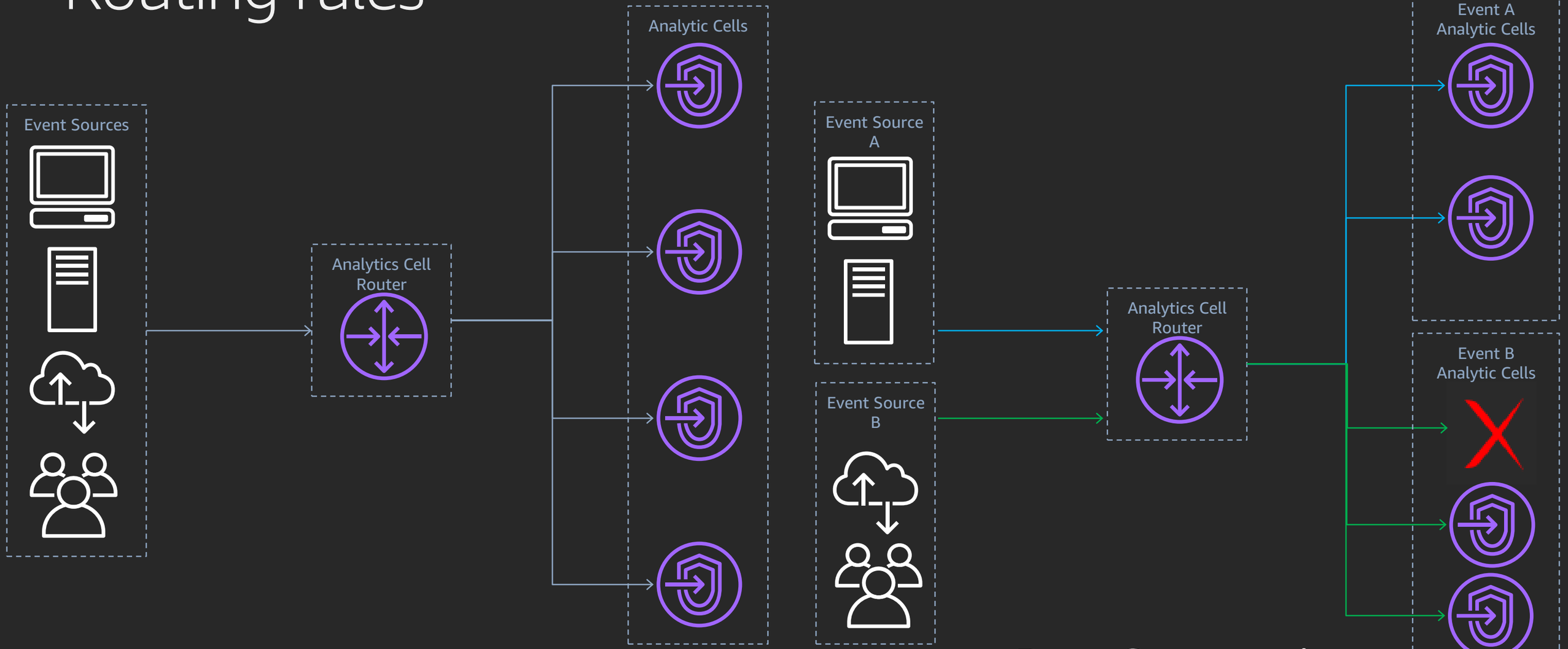
# Cellular design pattern



# Router architecture



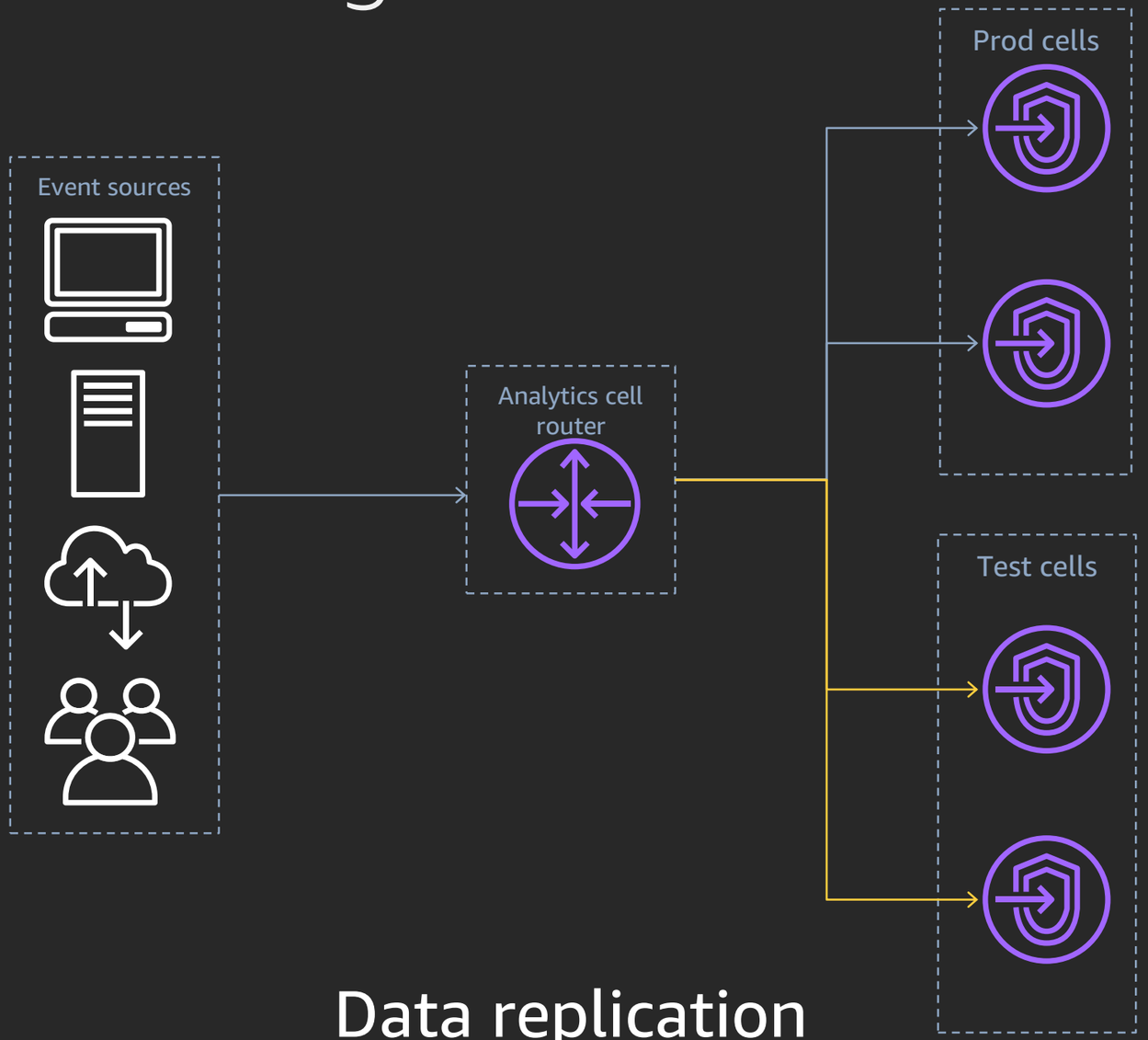
# Routing rules



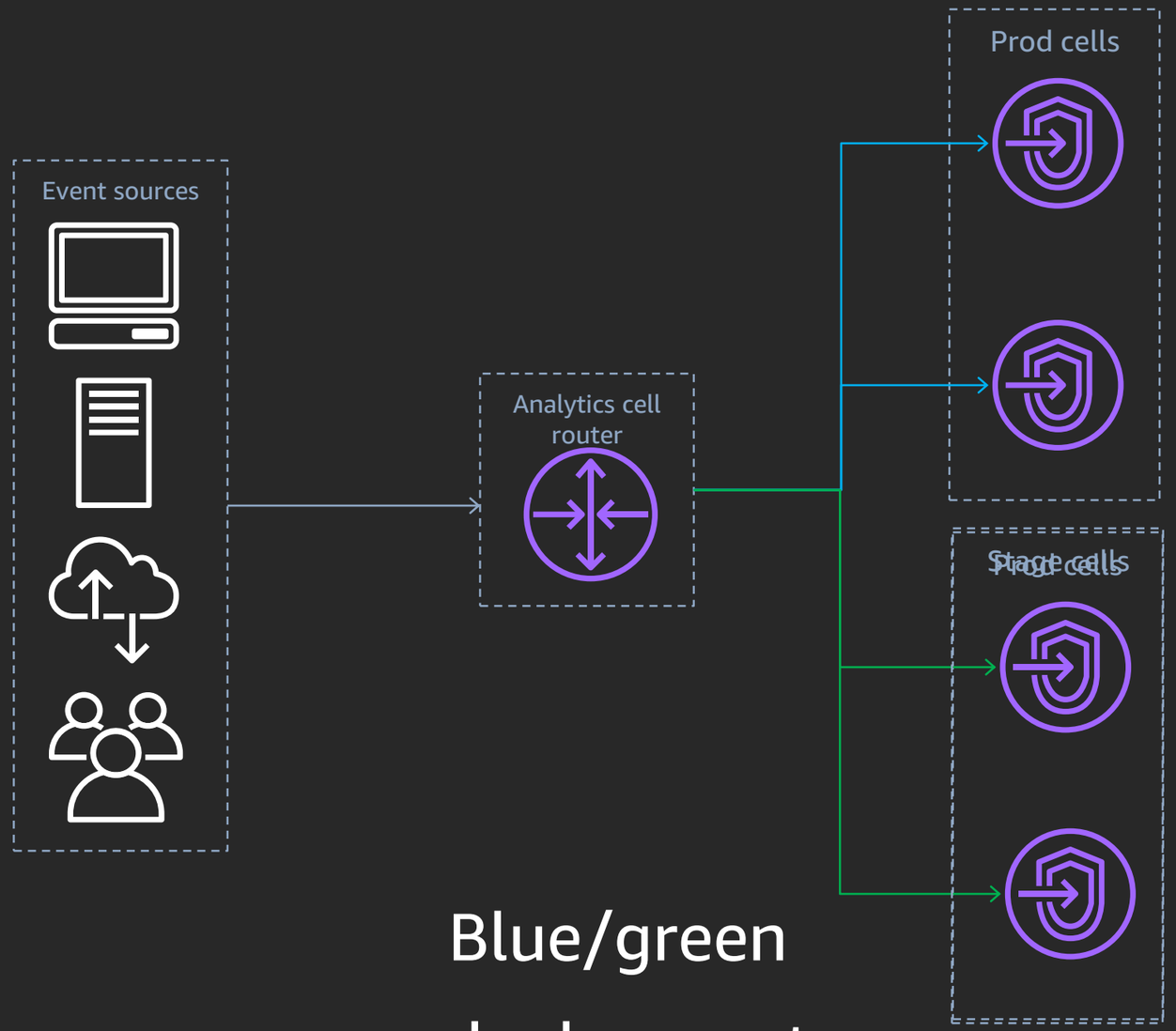

Load Balancing

Data Segregation with Load Balancing

# Routing rules




Data replication




Blue/green deployment

# Routing for SNS message

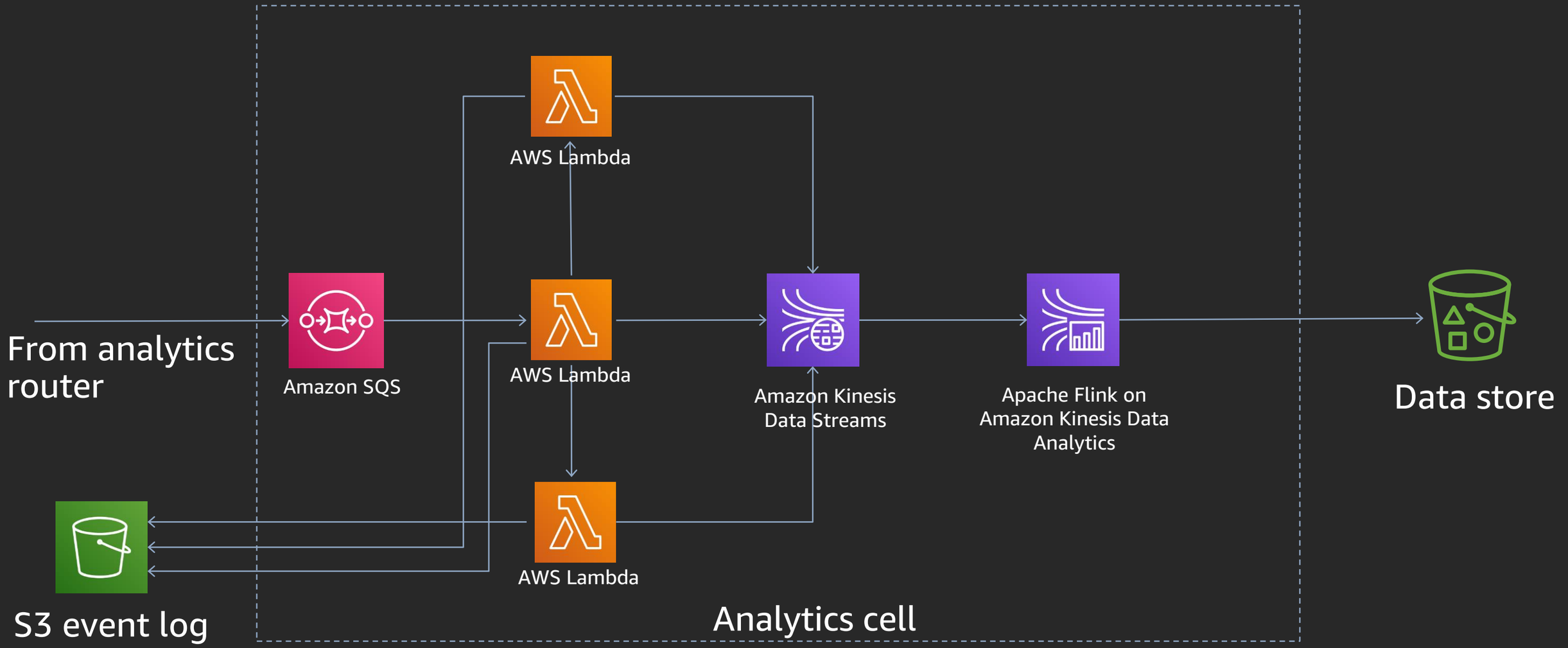

```
"Type"      : "Notification",
"MessageId" : "faf46c26-e7fb-5771-b6f4-38ec4b1bbc95",
"TopicArn"  : "arn:aws:sns:us-east-1:[redacted]:asap2-beta-us-east-1-log-events-topic",
"Subject"   : "Amazon S3 Notification",
"Message"   : "{\"Records\":[{\"eventVersion\":\"2.1\",\"eventSource\":\"aws:s3\",\"awsRegion\":\"us-east-1\",\"eventTime\":
      \"2019-09-15T21:20:36.132Z\",\"eventName\":\"ObjectCreated:CompleteMultipartUpload\",\"userIdentity\":{\"principalId\":
      \"AWS:AROA6L7TYIE5SPURYRA:asap2-beta-us-east-1-LogNormalizer\"},\"requestParameters\":{\"sourceIPAddress\":\"34.219.241.86\"},      \"responseElements
      \":{\"x-amz-request-id\":\"311A2040382E47BC\",\"x-amz-id-2\":\"j6TthhO+s2ZcptNRB5SM9Sw3b6napT1a3USBTAse2    +LT4LEsTpBcyEbZagQvkpdEXNvjaq06pvg=\"},
      \"s3\":{\"s3SchemaVersion\":\"1.0\",\"configurationId\":\"CloudtrailLogsCreateNotifications\",  \"bucket\":{\"name\":\"asap2-beta-us-east-1-log-
      events-bucket\",\"ownerIdentity\":{\"principalId\":\"A3AM8R2YBZLY\"},\"arn\":\"asap2-beta-      us-east-1-log-events-bucket\"},\"object\": {\"key\":
      \AndesService/application_log/AndesService/Prod/application.log/2019/09/15/19/application.log.2019-09-15-19.andes-prod-1b-841d3119.us-
      east-1.amazon.com.json.gz\",\"size\":11089,\"eTag\":\"017f62b142f21ad7fb9621749a329332\",\"sequencer\":\"005C50C1E6F538D4A3\"}}}]}",
      "Timestamp":  "2019-01-29T19:33:52.001Z",
```


Data segregation

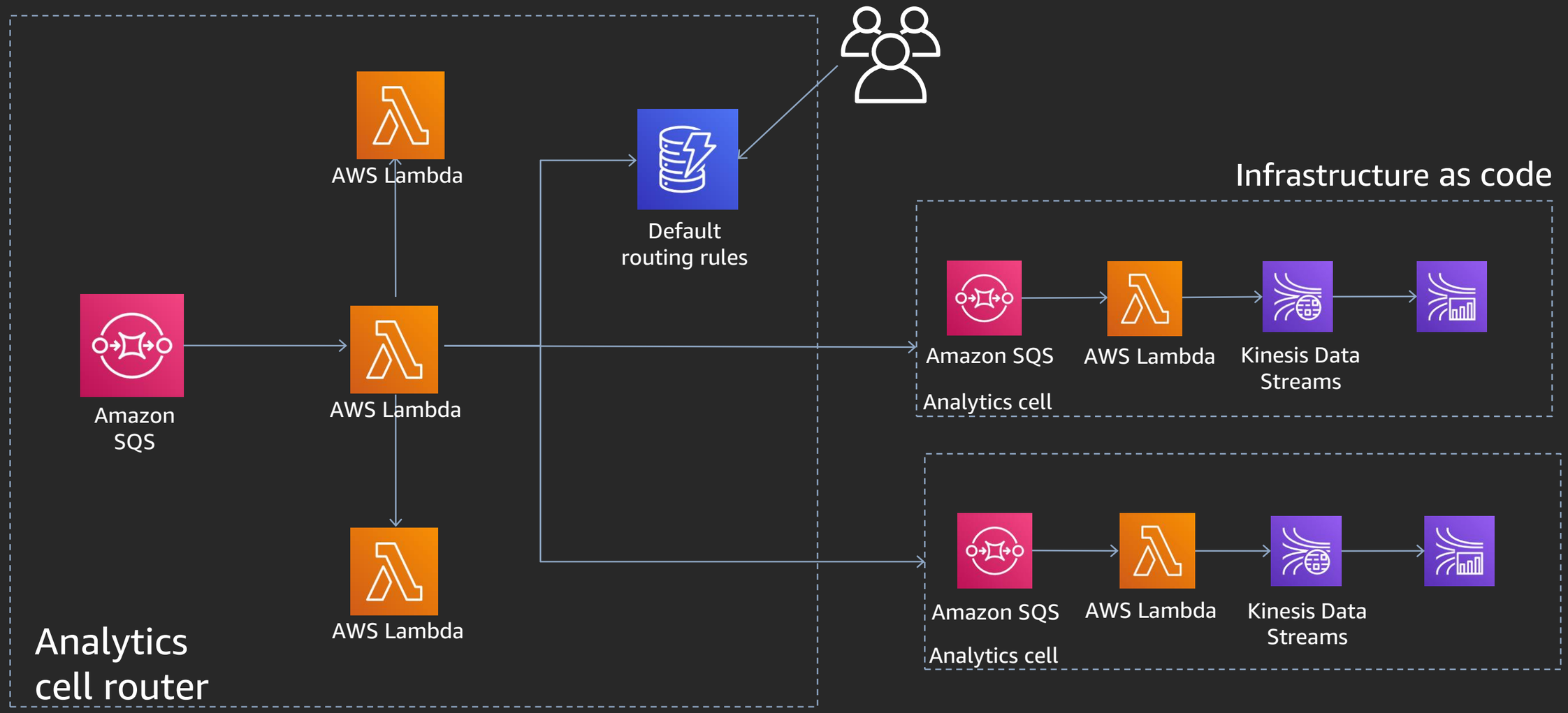

```
"Type" : "Notification",
"MessageId" : "faf46c26-e7fb-5b71-b6f4-38ec4b1bbc95",
"TopicArn" : "arn:aws:sns:us-east-1:[redacted]:Conduit-Cloudtrail-Log-Notification",
"Subject" : "Amazon S3 Notification",
"Message" : "{\"Records\":[{\"eventVersion\":\"2.1\",\"eventSource\":\"aws:s3\",\"awsRegion\":\"us-east-1\",\"eventTime\":\"2019-01-29T21:13:11.072Z\",\"eventName\":\"ObjectCreated:Put\",
           {\"principalId\":\"AWS:AROAI2LRE6HOGRN3IWEKI:regionalDeliverySession\"},\"requestParameters\":{\"sourceIPAddress\":\"34.219.241.86\"},\"responseElements\":
           {\"x-amz-request-id\":\"311A2040382E47BC\",\"x-amz-id-2\":\"j6TthhO+s2ZcptNRB5SM9Sw3b6napT1a3USBTAse2+LT4LEsTpBcyEbZagQvkpdEXNvjaq06pvg=\"},\"s3\":
           {\"s3SchemaVersion\":\"1.0\",\"configurationId\":\"CloudtrailLogsCreateNotifications\",\"bucket\":{\"name\":\"conduit-cloudtrail-logs\",\"ownerIdentity\":
           {\"principalId\":\"A3AM8R2VAYBZLY\"},\"arn\":\"arn:aws:s3:::conduit-cloudtrail-logs\"},\"object\": {\"key\":\"trail-logs/AWSLogs/549802330112/CloudTrail/us-west-2/2019/01/29/
```


Load balancing by AWS Account ID

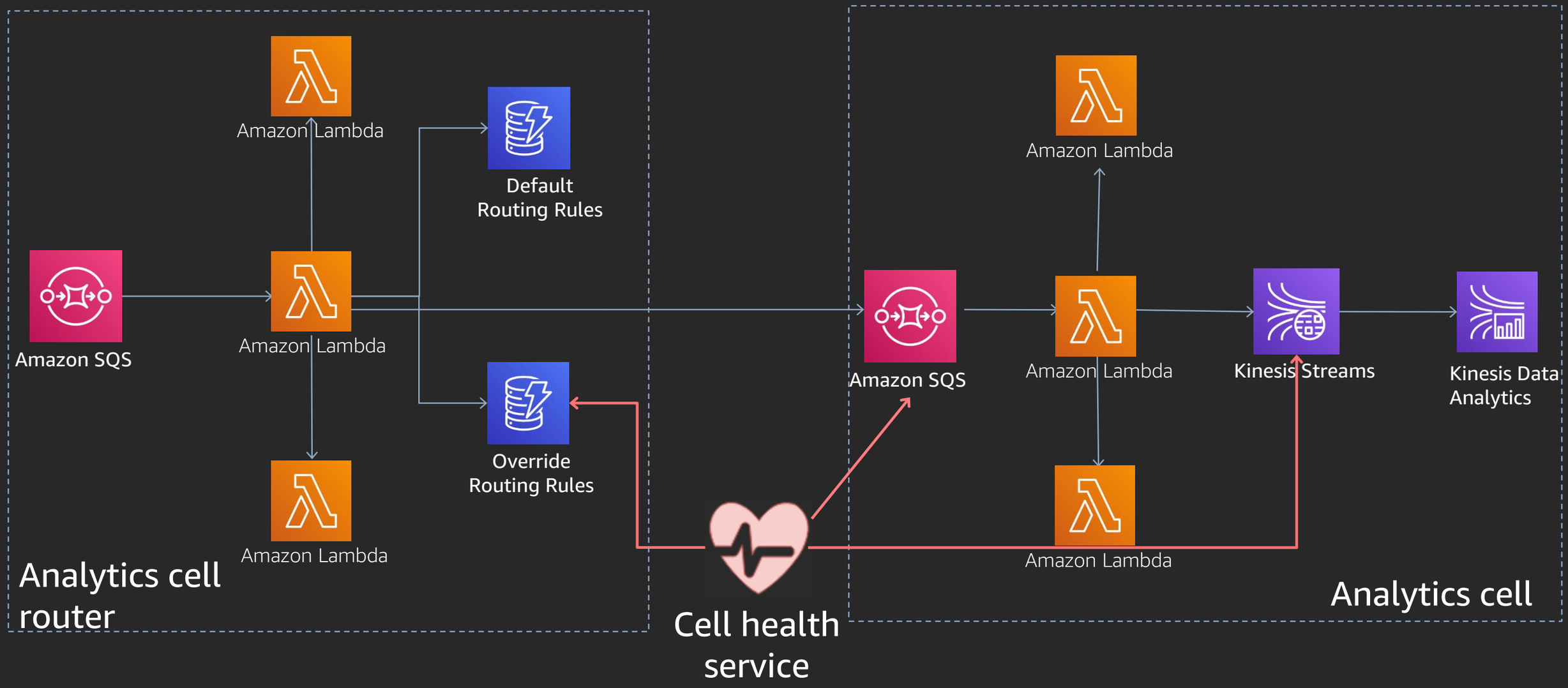
# Analytics cell



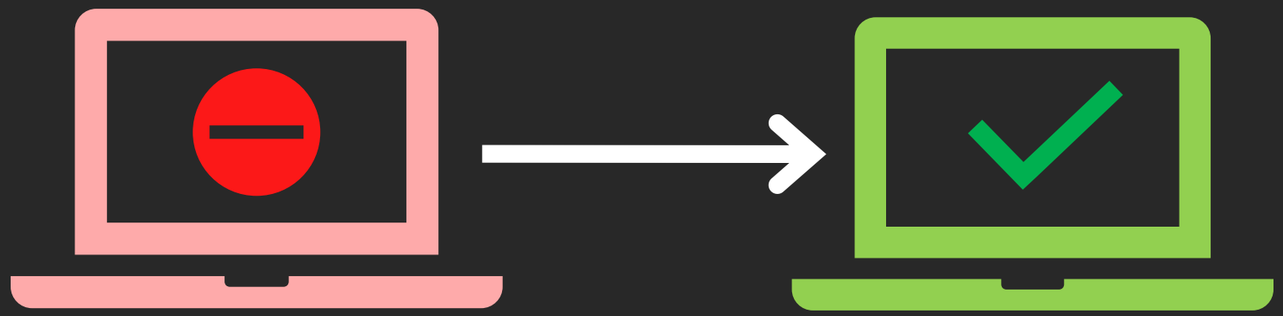
# Adding cells



# Cell health service

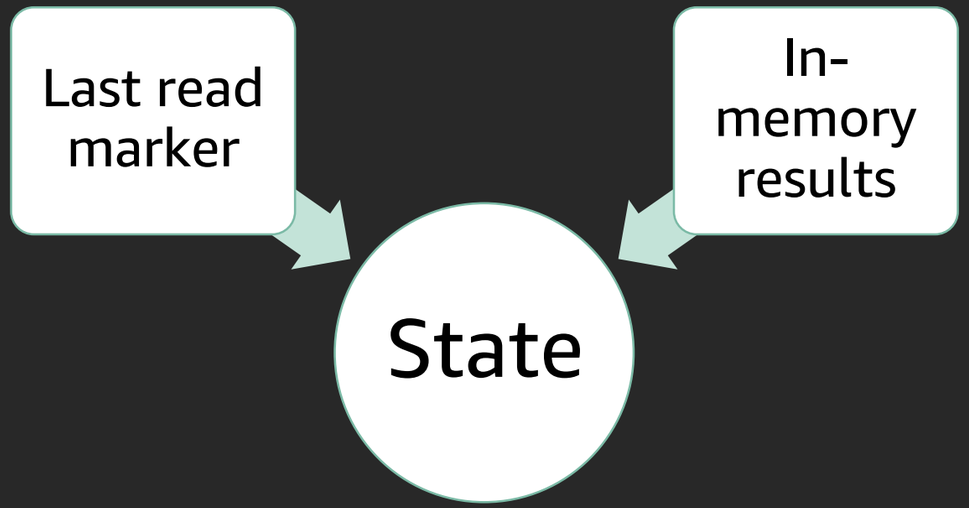


# Cell state

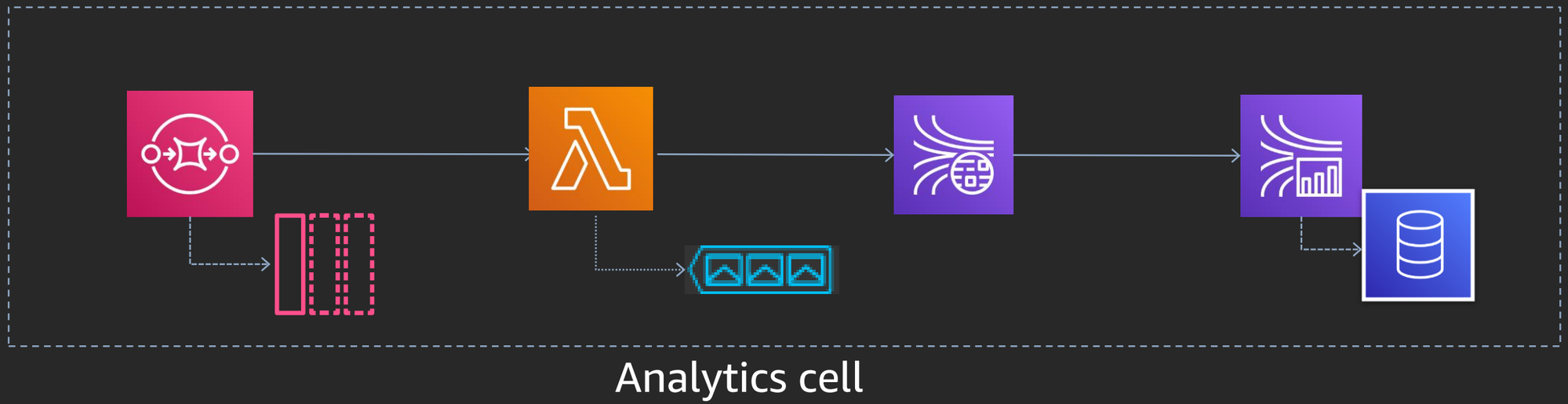


Why state?

What is state?



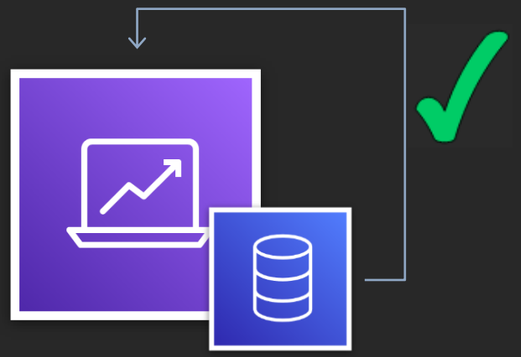
# State within analytics cell




- Amazon SQS
- Lambda failures – Dead-letter queue
- Apache Flink checkpoints and save points

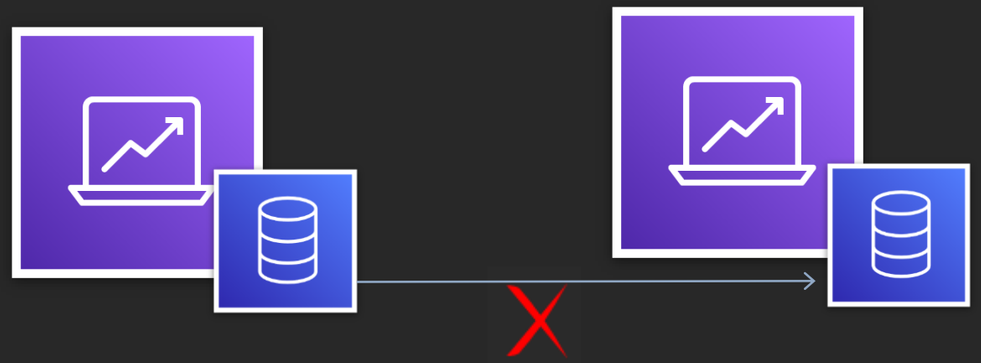
# What happens when a cell fails

## Recoverable failure



State maintained within a cell

## Unrecoverable failure



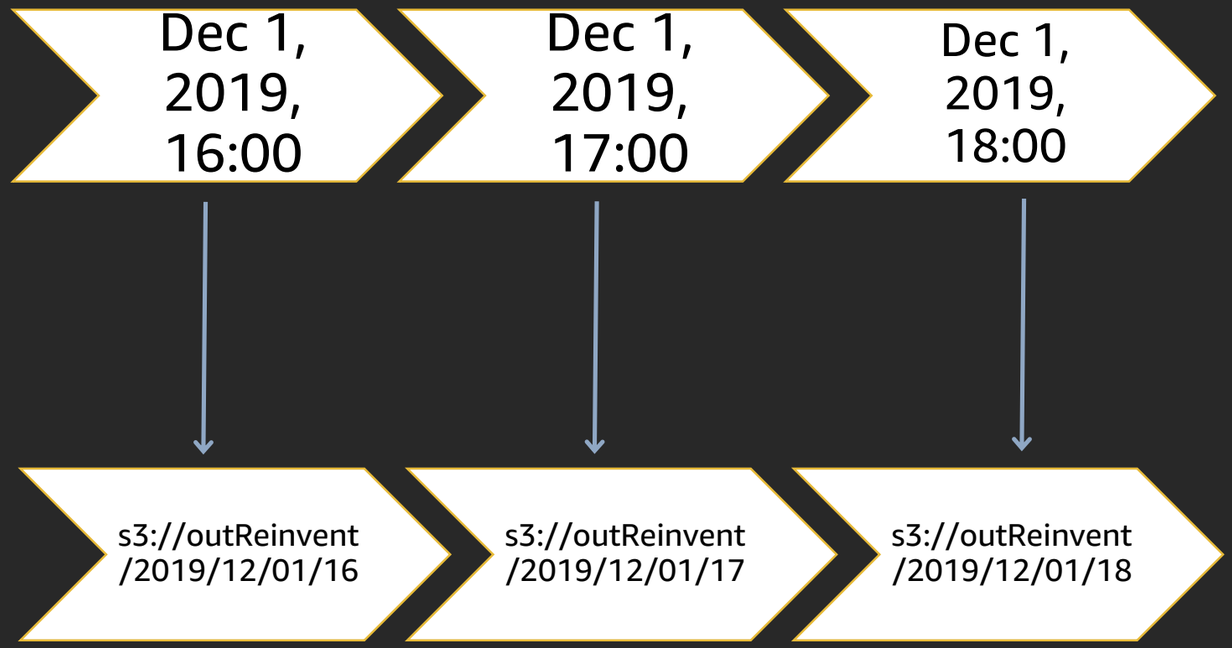
State never migrated across cells

# Unrecoverable failure

- Historical event cell
- Lambda to regenerate SNS notifications

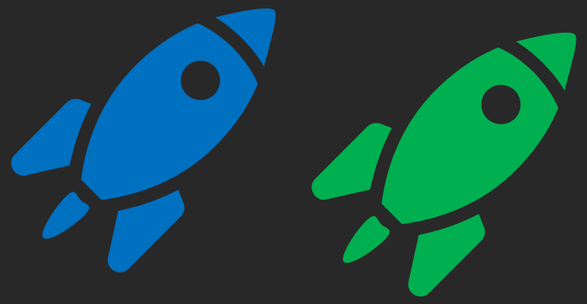
# Unrecoverable failure

- Backfill and restate historical results
- Event time partition on output data



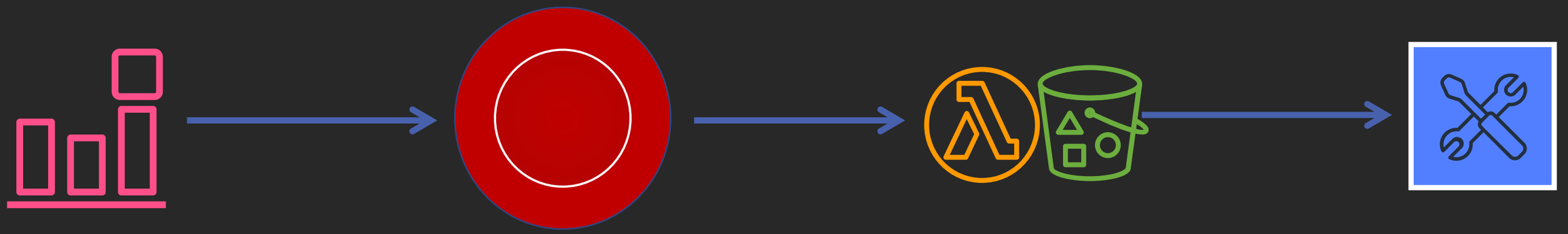
# More ways of reducing blast radius

Isolate noisy neighbors



Blue/green deployment

# Back to tenets



Event-driven architecture

Minimize failures

Serverless

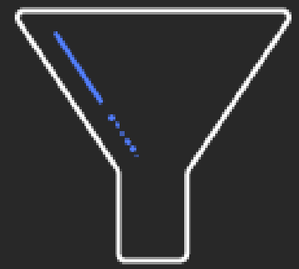
Infrastructure as code

# Key takeaways

vs.

Small vs. large cells

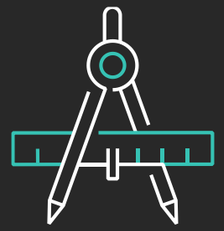
Design for failure



Thinnest possible layer

# 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

# Related breakouts

ARC342-R Cell-based architectures for global, well-architected apps

ARC349-R Beyond five 9s: Lessons from our highest available data planes

# Thank you!

AWS re:Invent



aws

# Please complete the session survey in the mobile app.