
AWS
re:Invent
NOV. 28 – DEC. 2, 2022 | LAS VEGAS, NV

STP208

Designing resilient architectures for startups

Laura Hyatt (she/her)

Global Solutions Architect, Resilience Hub
AWS

Rajan Patel (he/him)

Sr. Solutions Architect, Startups
AWS

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

What drives startups

- ✓ Laser focus
- ✓ Decisions on daily basis
- ✓ Operating milestone to milestone
- ✓ Short-term plans, long-term goals
- ✓ No recipe for success
- ✓ Experimentation
- ✓ Frequent adjustments



aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

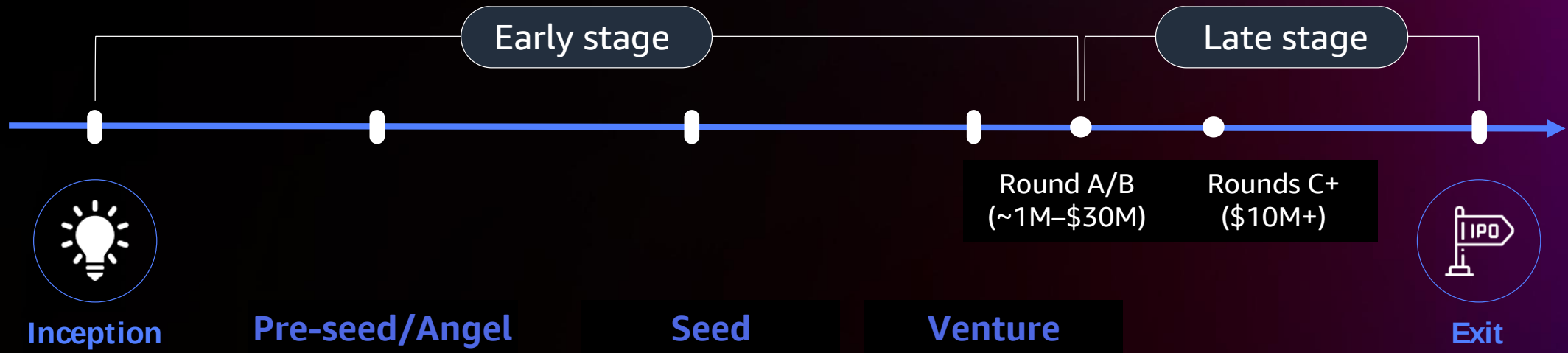
“The only way to win is to learn faster than anyone else.”

Eric Ries

The Lean Startup

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

The journey of a startup



- Start with (near) zero investments
- Create PoCs or MVPs
- Fit product to market
- Learn rapidly before scaling

- Move quickly
- Start to scale
- Build out product
- Keep costs low
- Develop product
- Expand teams

- Scale out with new customers
- Find new markets
- Deploy in other regions
- Prepare for IPO or acquisition

aws
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

A serverless business strategy

What do we mean when we say serverless?

- ✓ No server management
- ✓ Flexible, automated scaling
- ✓ Pay for value
- ✓ Automated high availability

Serverless-first is the decision to opt for serverless technologies in your application as a first choice.

aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Serverless on AWS

Serverless is more than compute

Compute


AWS Lambda


AWS Fargate

Data stores


Amazon S3


Amazon Aurora Serverless


Amazon DynamoDB

Integration


Amazon EventBridge


Amazon API Gateway


Amazon SQS


Amazon SNS


AWS Step Functions


AWS AppSync

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

“If serverless scales automatically, what is left for us architects to do?”

Oh, so much my friend…

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

reliability

A reliable workload performs its intended function correctly and consistently when it’s expected to.

resiliency

A resilient workload is able to withstand partial and intermittent failures across components and eventually recover from unexpected conditions.

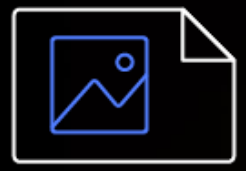
aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

But what about resilience for my startup if I go serverless?

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

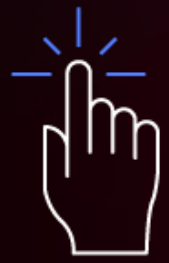
Four essential capabilities in a resilient system



Anticipating



Monitoring



Responding



Learning

aws

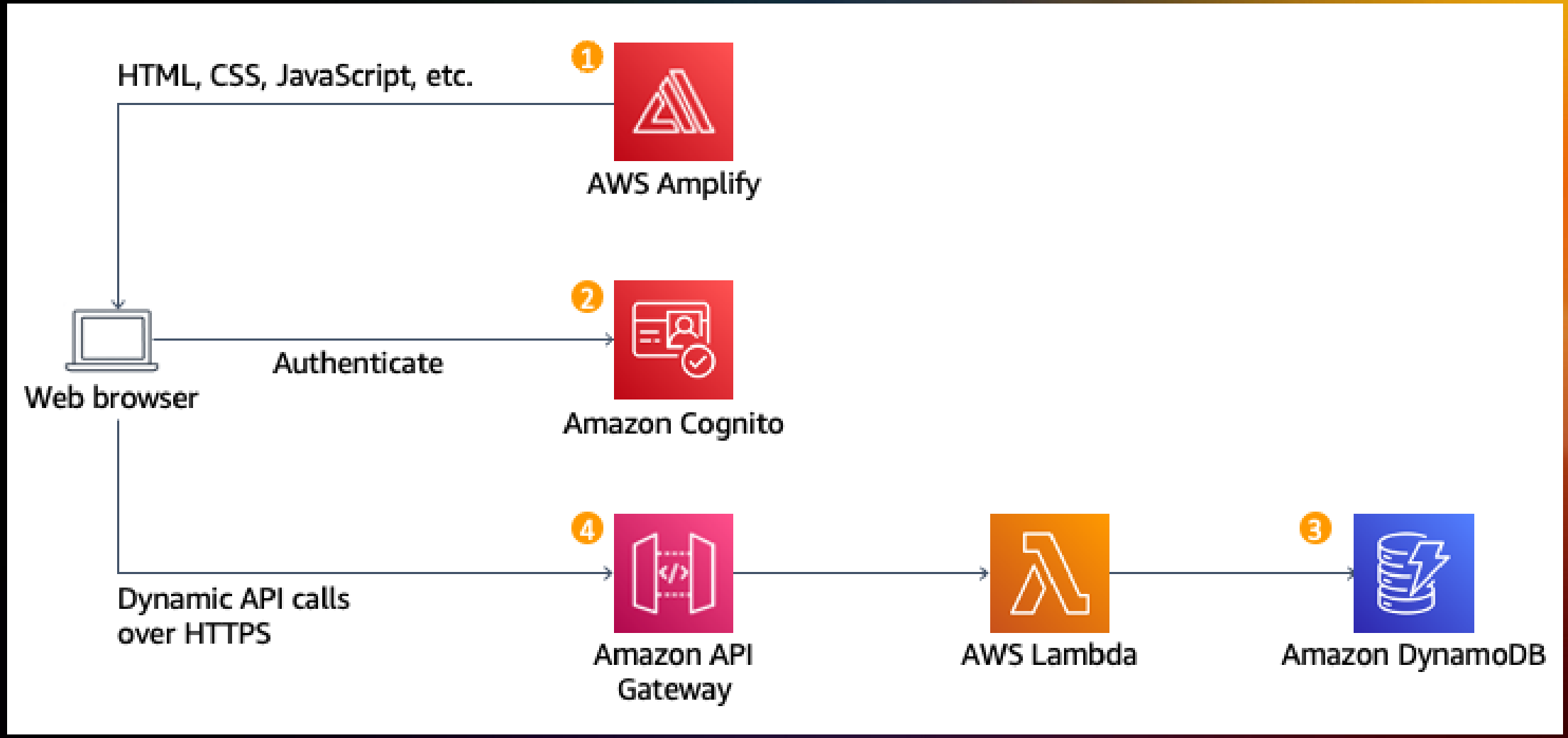
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Let’s dive into a architectural use case…

…To the whiteboard!

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

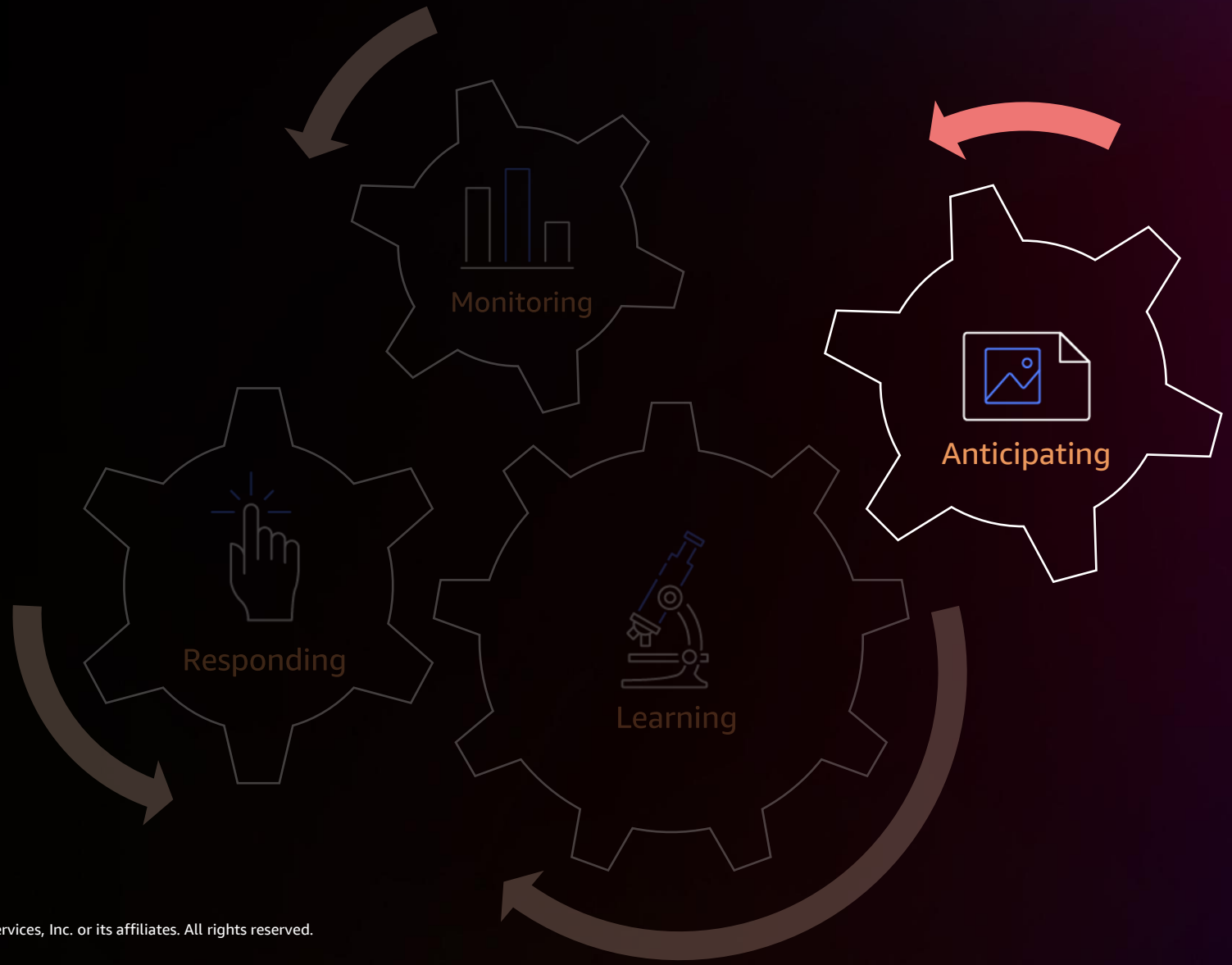
Whiteboard base



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Continuous resilience

ANTICIPATING



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Let’s anticipate some problems with serverless

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Lambda has fixed resources per unit of work

SCALING IS IMPORTANT BUT ALSO ONE OF THE HIGHEST CAUSES OF DISRUPTIONS

- Each **function execution environment** has the same finite resources:
  - CPU (up to 6 vCPUs)
  - Memory (up to 10GB)
  - Storage (up to 10GB)
  - Network
- Each execution environment handles **one, and only one, event** at a given time. After it is done handling that event it can handle additional events.
- Each request gets the same resources.



aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

As work increases, scale concurrent environments*

- Workload isolation means predictable performance
- Latency and throughput are constant

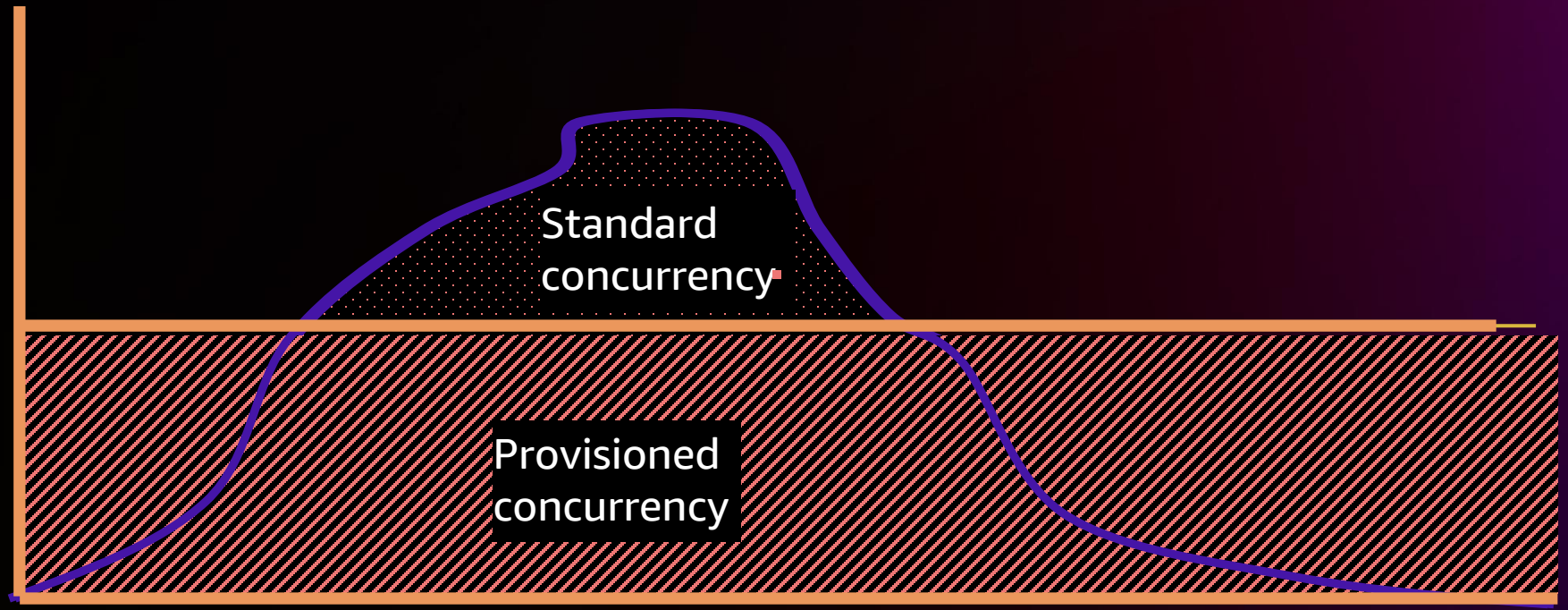




* Concurrency limits apply

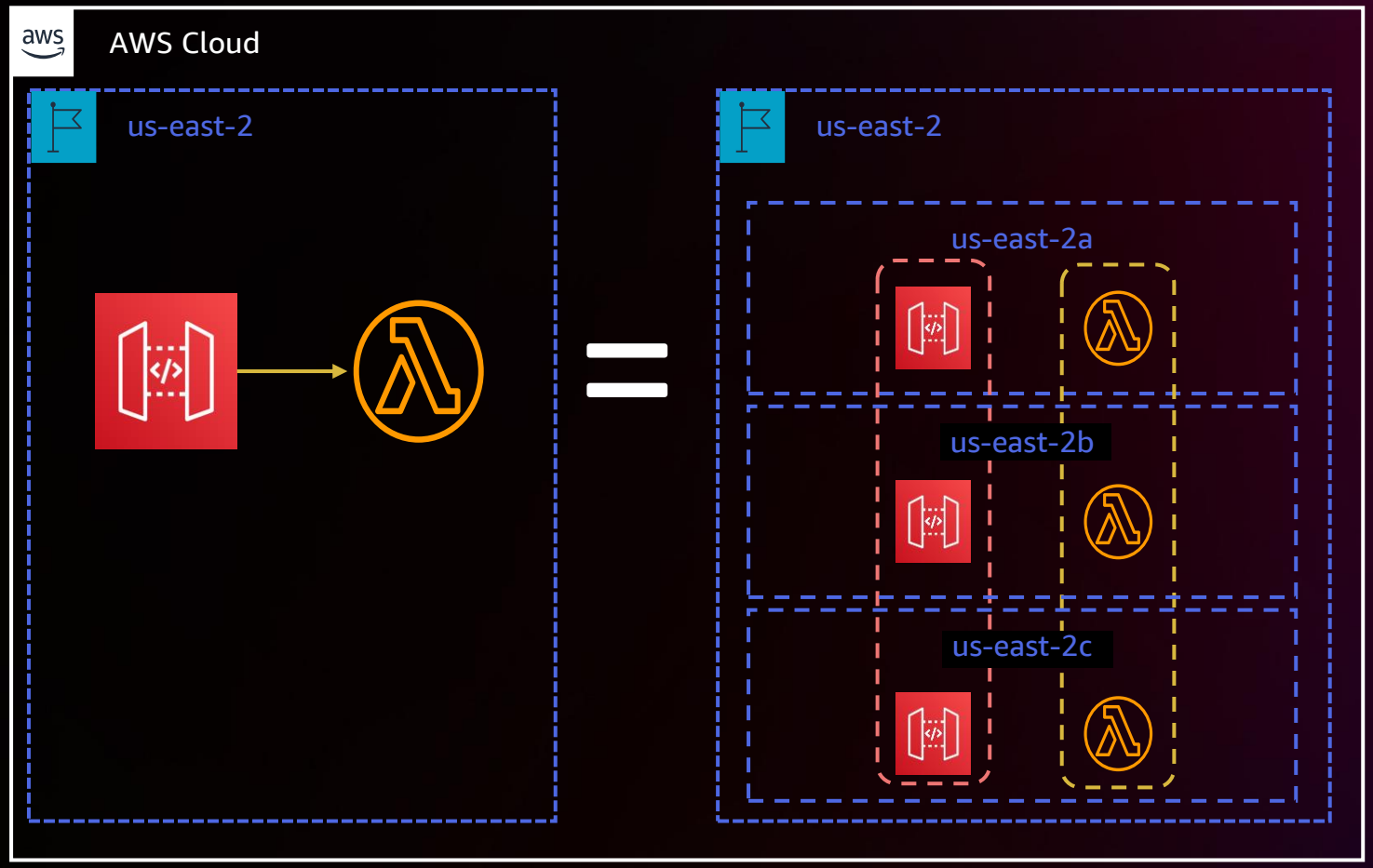
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Provisioned concurrency to scale without fluctuations



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

With serverless, Multi-AZ comes for free



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

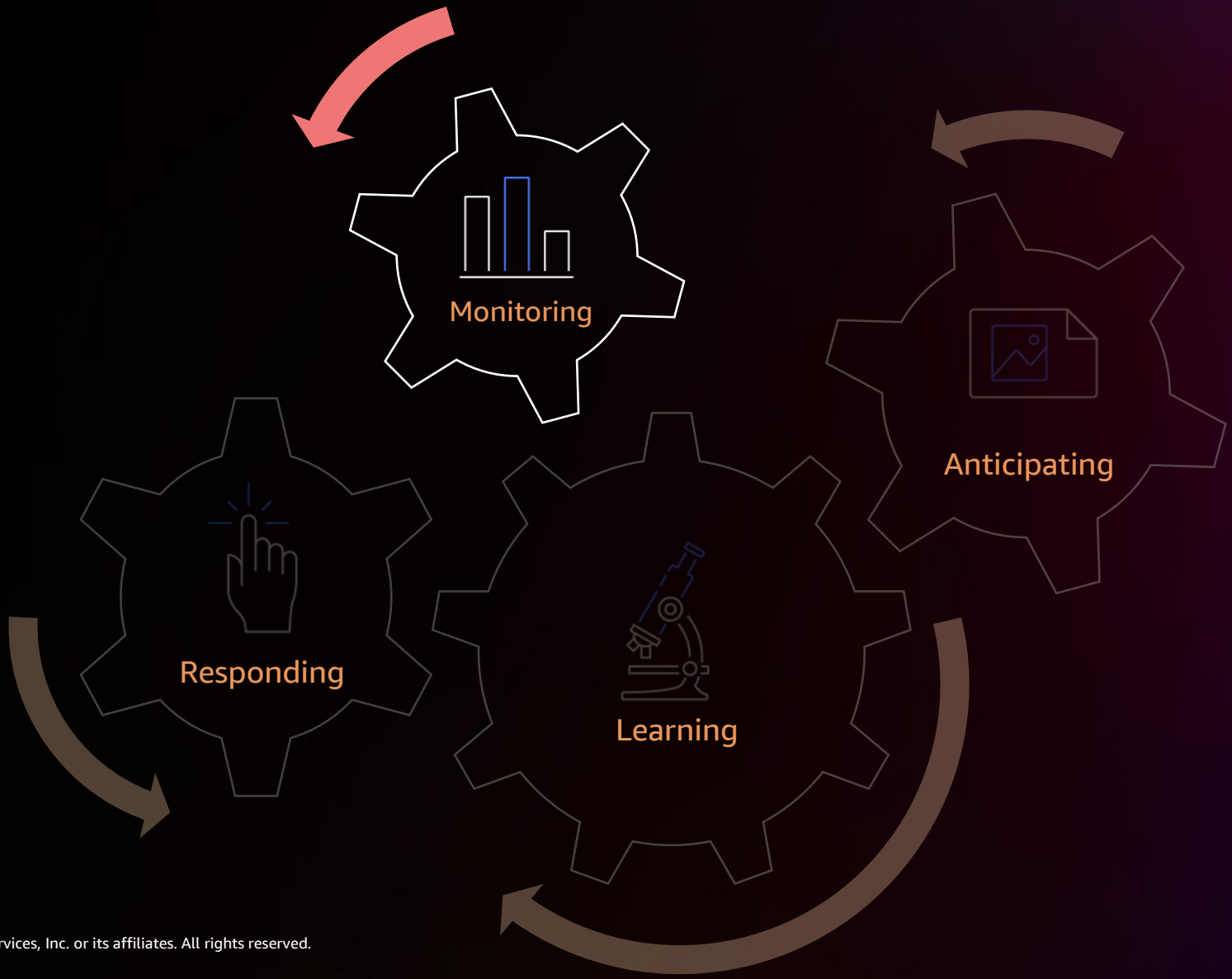
… however

If you are attaching Lambda to a VPC, you do need to include subnets in Multi-AZ – this is not automatic.

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

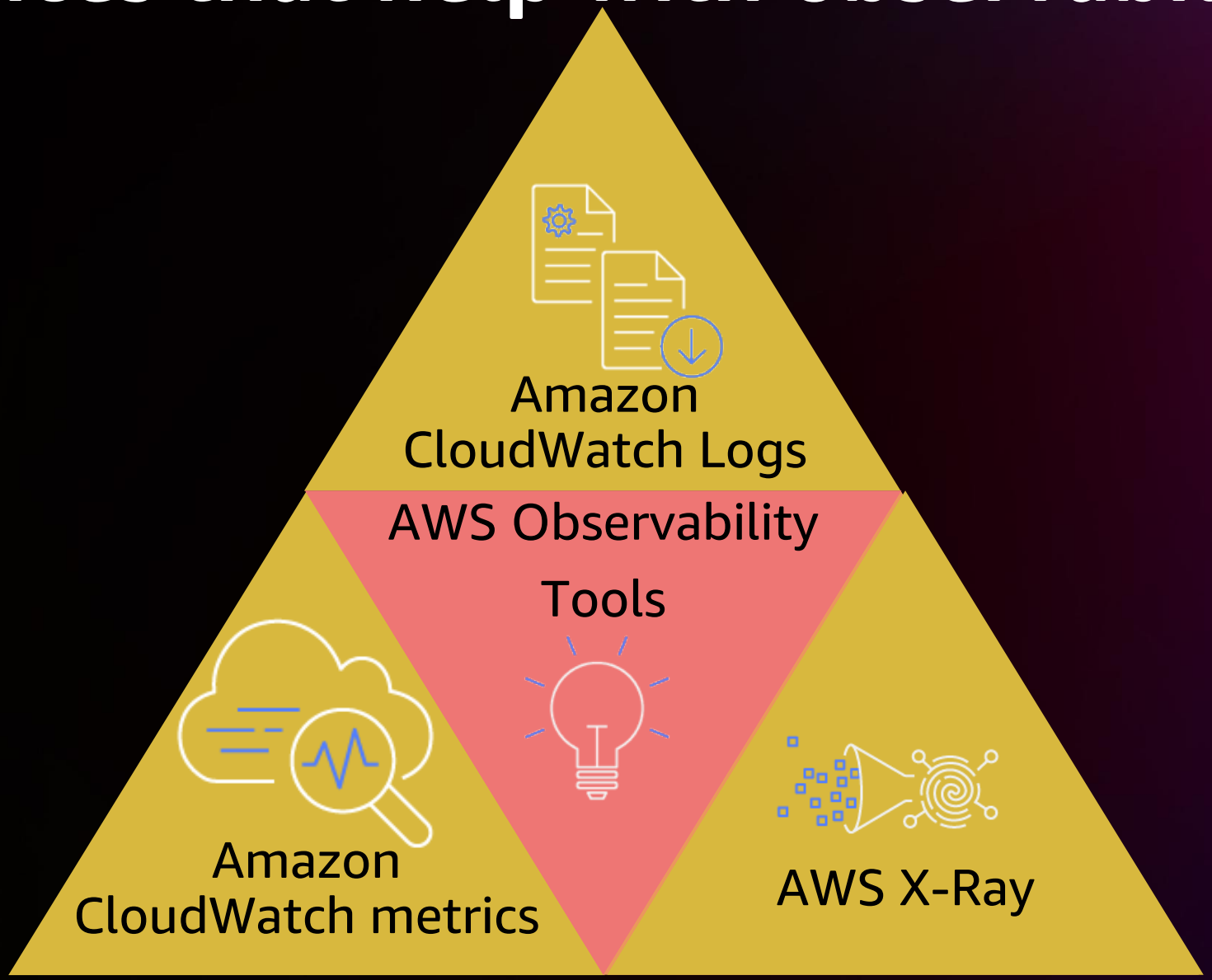
Continuous resilience

MONITORING



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

AWS services that help with observability



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Take action with Amazon CloudWatch

AWS metrics

- AWS Lambda
  - Errors, throttles, iterator age, concurrency
- Amazon API Gateway
  - Success rate
  - Latency (tail p90, p95, p99)
- Amazon SQS
  - Message age

Custom metrics

- Use embedded metric format (EMF) to generate custom metrics:
  - Business, customer metrics
  - Operational metrics
  - Errors

aws
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

AWS X-Ray enables tracing of distributed applications

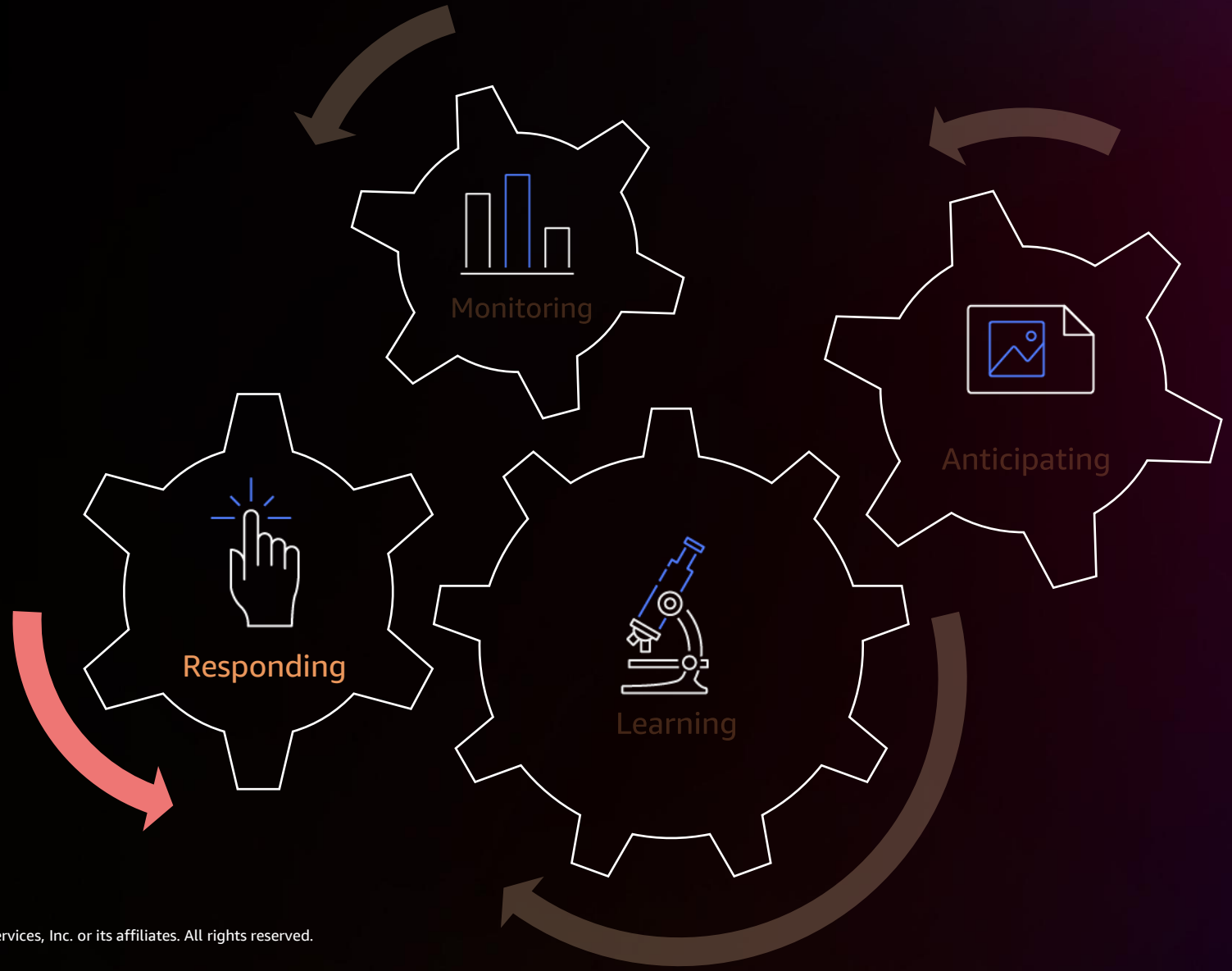
- Scales for microservice and serverless architectures
- Enables the identification of the root cause of performance issues and errors
- Provides a cross-service view of requests made to application



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Continuous resilience

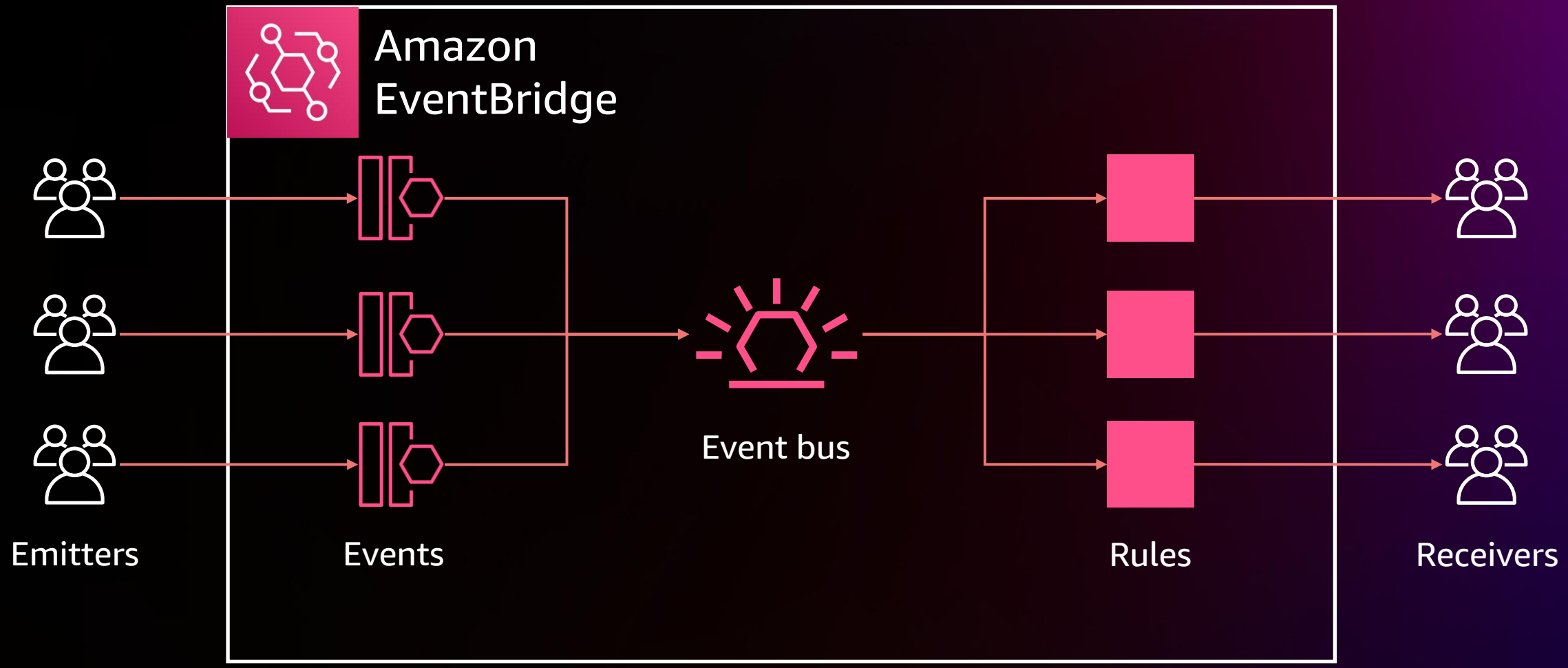
RESPONDING



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Event-driven patterns

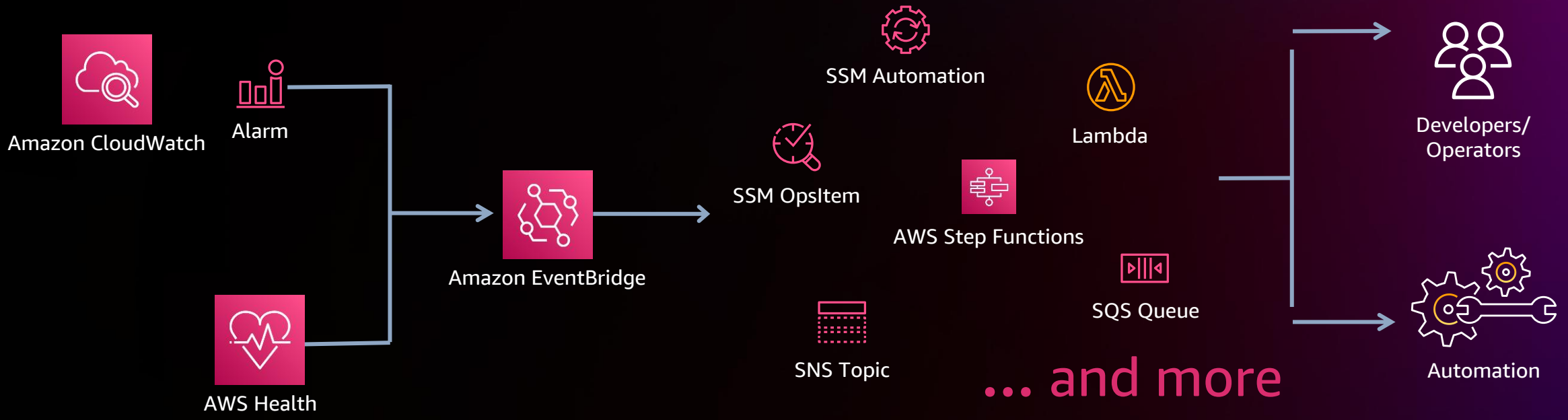
AUTOMATED RESPONSES THAT RUN IN RESPONSE TO EVENTS



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Real-time processing and alarming

- Send notifications
- Automate responses

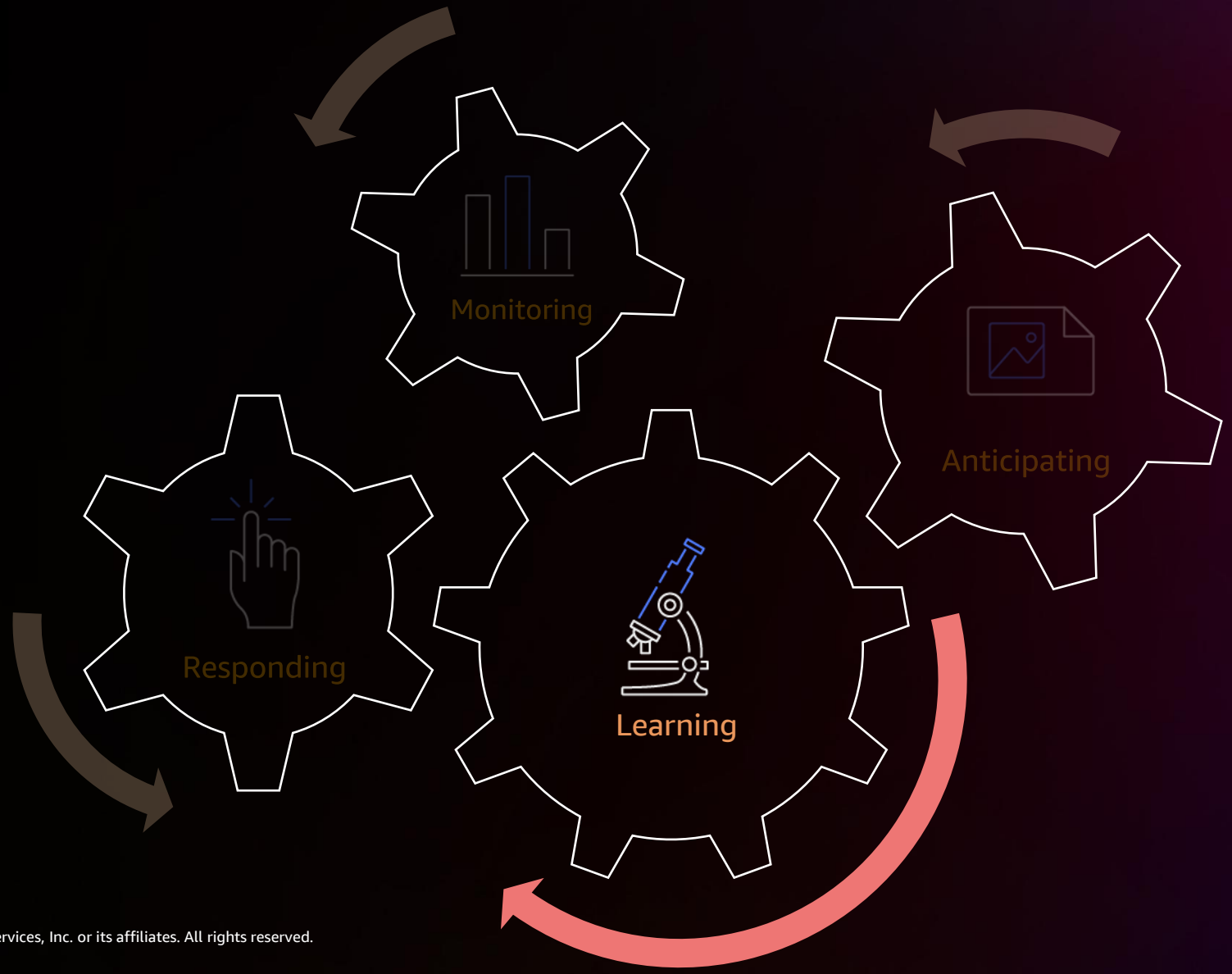



```
{
  "source": ["aws.health"],
  "detail-type": ["AWS Health Event"],
  "detail": {
    "service": ["S3"],
    "eventTypeCategory": ["issue"],
    "eventTypeCode": ["AWS_S3_INCREASED_GET_API_ERROR_RATES",
"AWS_S3_INCREASED_PUT_API_ERROR_RATES"]
  }
}
```

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Continuous resilience

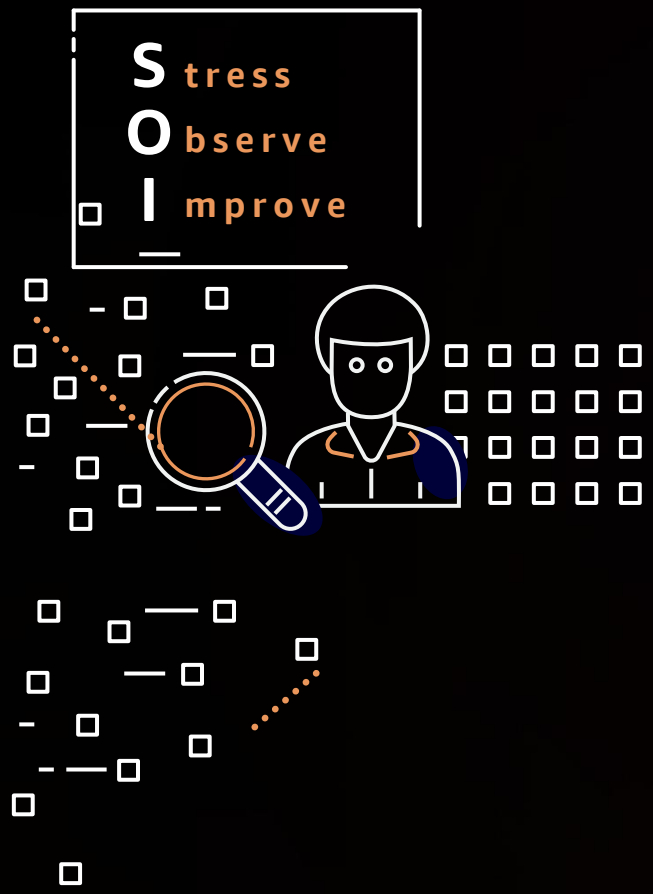
LEARNING



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Chaos engineering

WITH AWS FAULT INJECTION SIMULATOR



- Improve resilience and performance
- Uncover hidden issues
- Expose blind spots
  Monitoring, observability, and alarm
- And more

aws
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

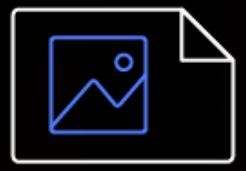
Native Actions

- Amazon EC2 actions
  - aws:ec2:reboot-instances
  - aws:ec2:send-spot-instance-interruptions
  - aws:ec2:stop-instances
  - aws:ec2:terminate-instances
- Amazon ECS actions
  - aws:ecs:drain-container-instances
  - aws:ecs:stop-task
- Amazon EKS actions
  - aws:eks:inject-kubernetes-custom-resource
  - aws:eks:terminate-nodegroup-instances
- Network actions
  - aws:network:disrupt-connectivity
  - Denies the specified traffic to the target subnets.
- Amazon RDS actions
  - aws:rds:failover-db-cluster
  - aws:rds:reboot-db-instances

aws
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

What you should take away from this

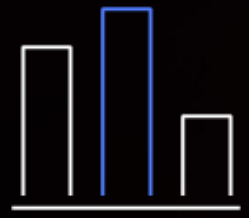
CALL TO ACTION: APPLY TO ALL YOUR SERVERLESS WORKLOADS



ANTICIPATING

Action:

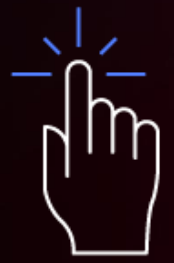
- What are are you trying to anticipate?
- Understand your own architecture



MONITORING

Action:

- Use the tools out there for your specific business needs
- Design metrics and traces that matter for your scenario



RESPONDING

Action:

- Take automated action when something does occur



LEARNING

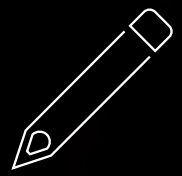
Action:

- Test, test, test!
- Test the unknown with Chaos Engineering
- Use this data to improve your architecture

aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

References



- Building well-architected serverless applications: Building in resiliency

  (https://aws.amazon.com/blogs/compute/building-well-architected-serverless-applications-building-in-resiliency-part-1/)



- Well-Architected Serverless Application Lens – Reliability

  (https://docs.aws.amazon.com/wellarchitected/latest/serverless-applications-lens/reliability-pillar.html)



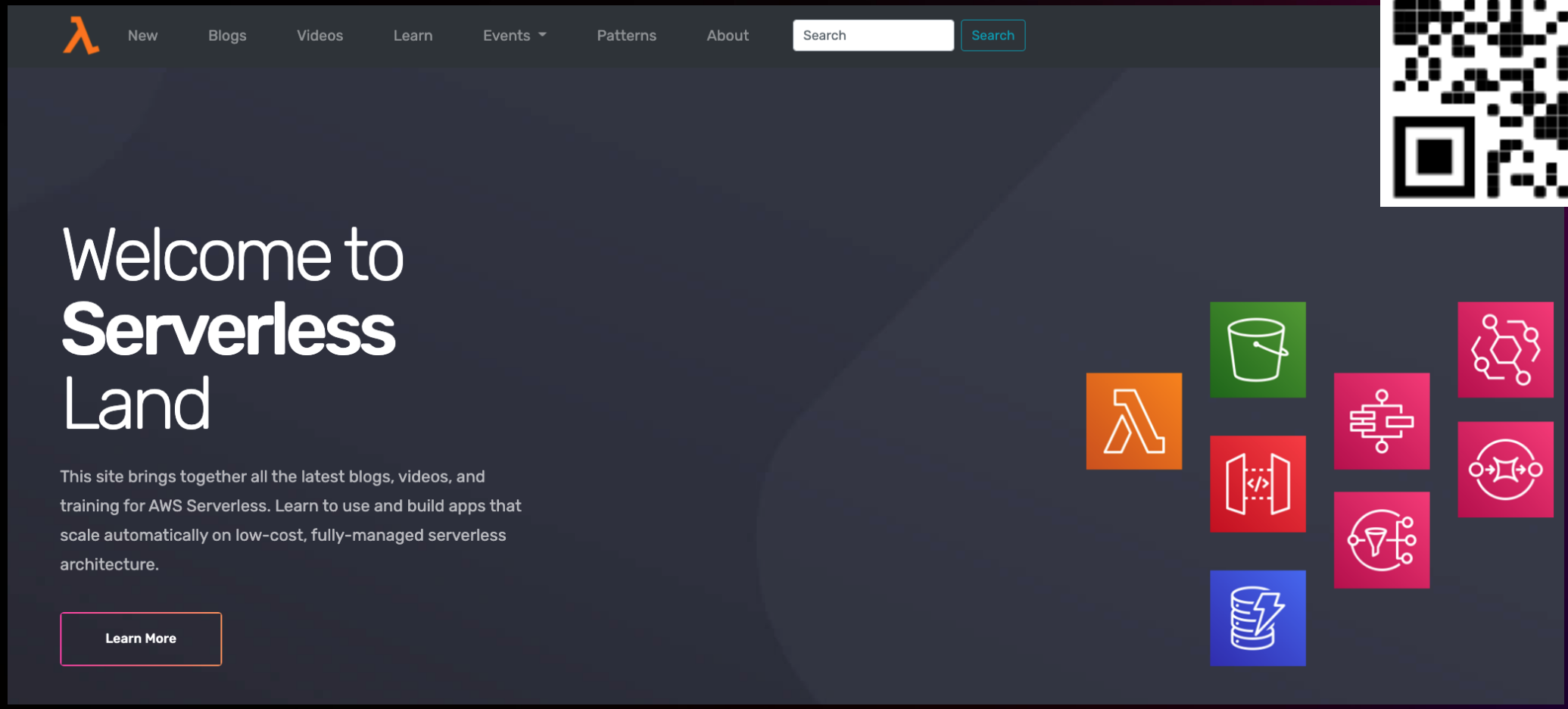
- Resilience Engineering – Serverless workshop

  (https://catalog.us-east-1.prod.workshops.aws/workshops/3015a19d-0e07-4493-9781-6c02a7626c65/en-US)

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

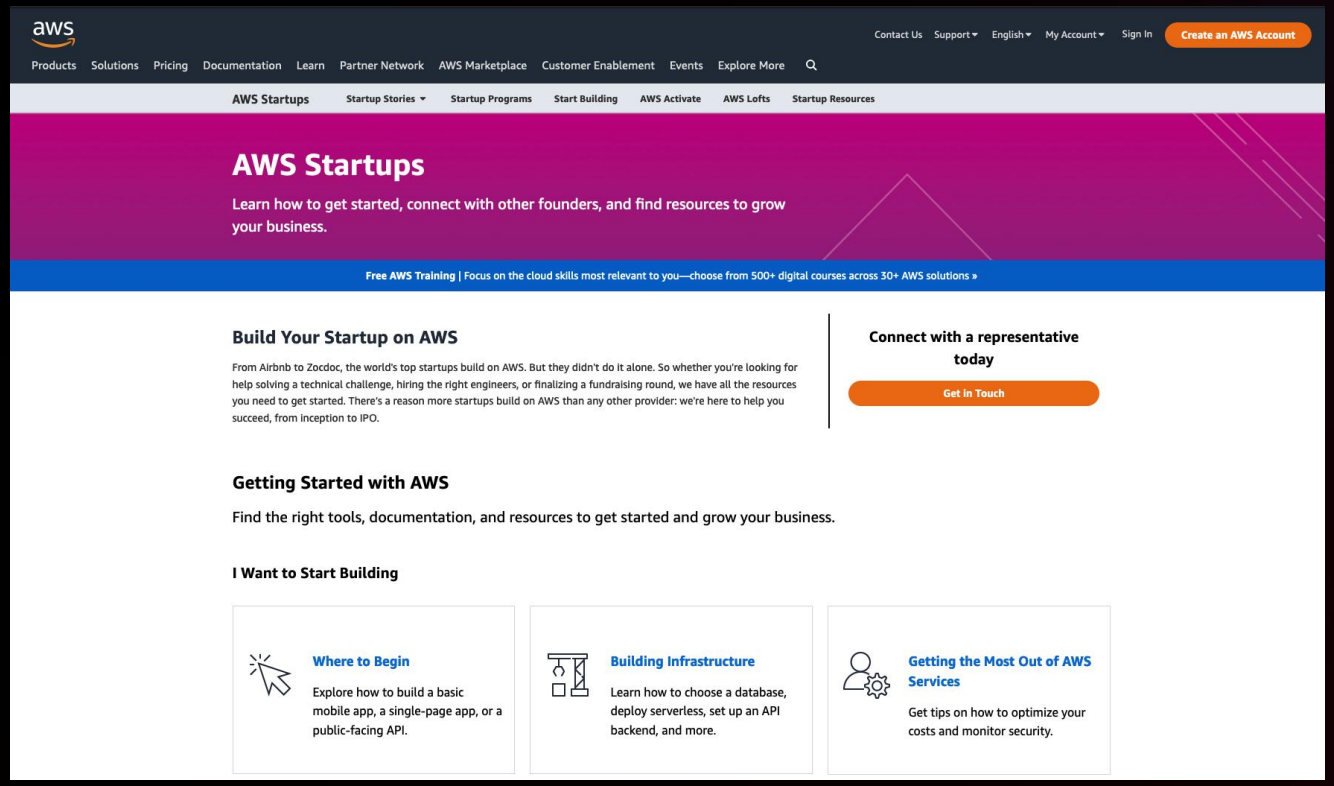
Serverless Land

HTTPS://SERVERLESSLAND.COM/



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

AWS helps startups grow



Visit an AWS Startup Loft

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Thank you!

Laura Hyatt
@laurajhyatt

Rajan Patel
@rajp404



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

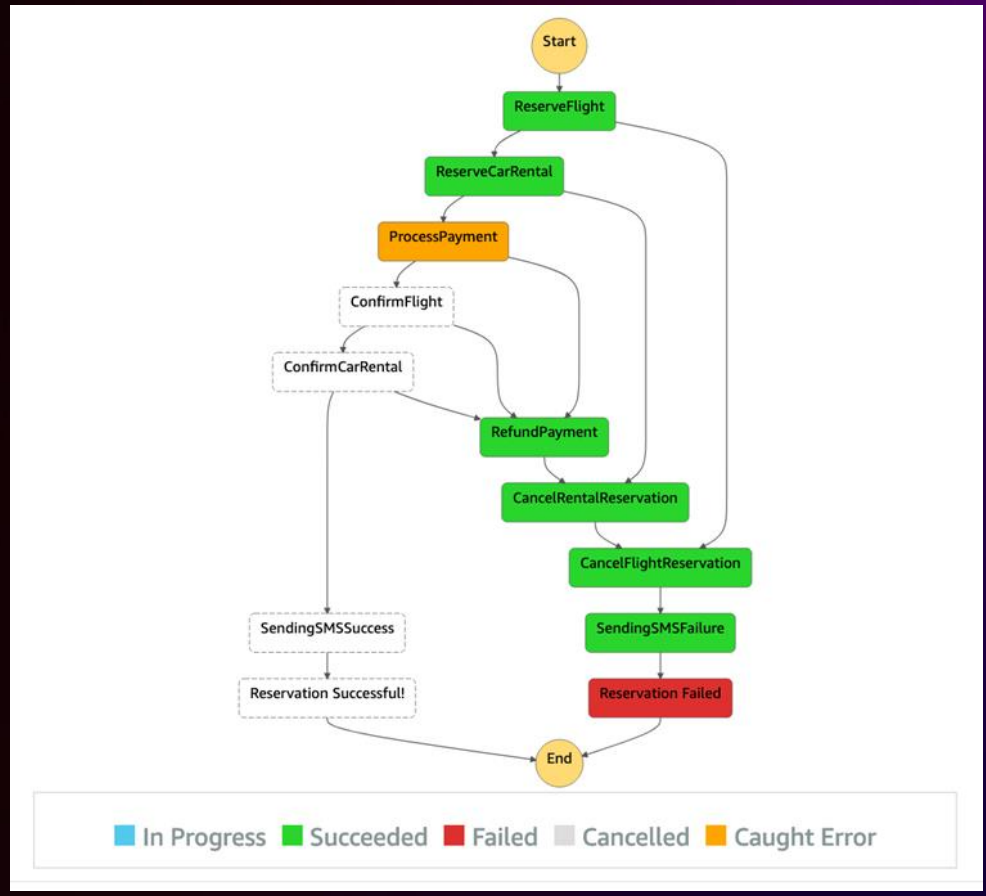
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

If you have long-running transactions, use AWS Step Functions

AWS Lambda can run up to 15 minutes per execution.

With AWS Step Functions, you can string multiple Lambda functions together, and this has resilience benefits too!

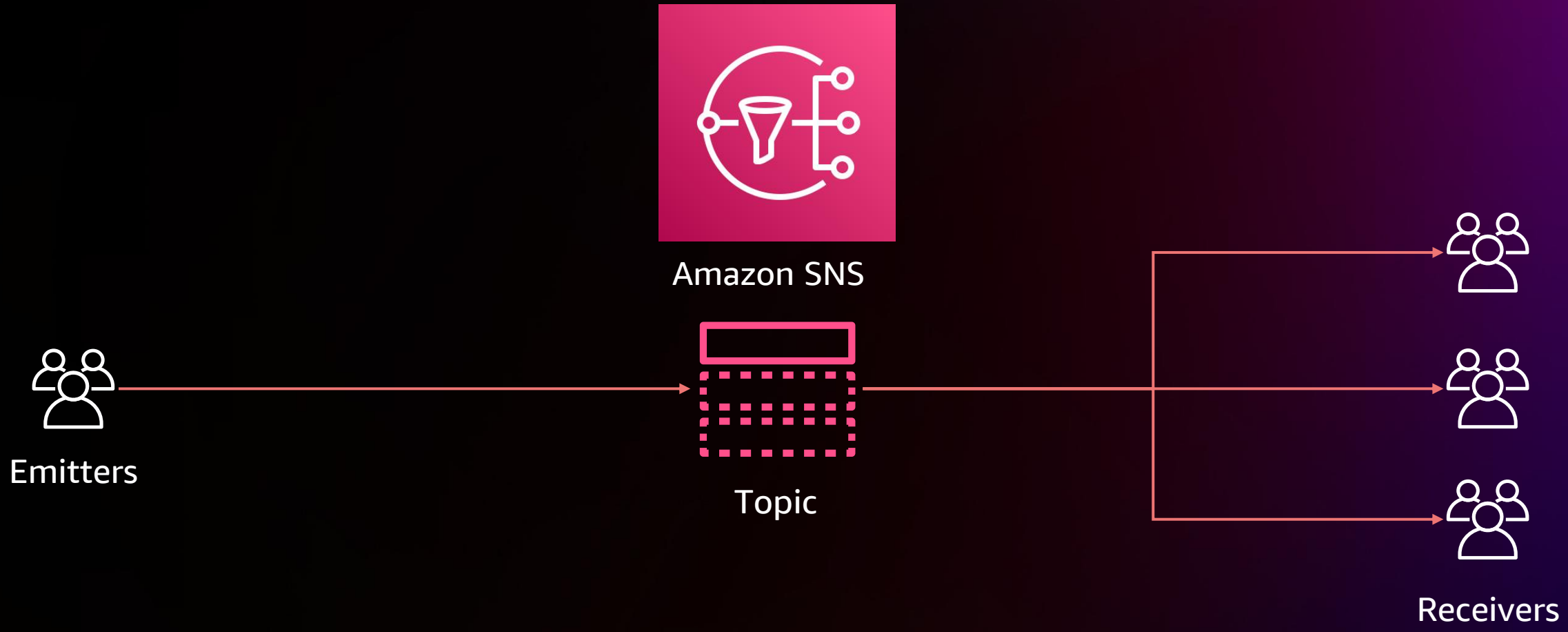
- Significant cause of Lambda failures is timeout. Breaking up long functions gives you a runtime buffer.
- You can handle errors gracefully and invoke another function.



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Event-driven patterns

AUTOMATED RESPONSES THAT RUN IN RESPONSE TO EVENTS

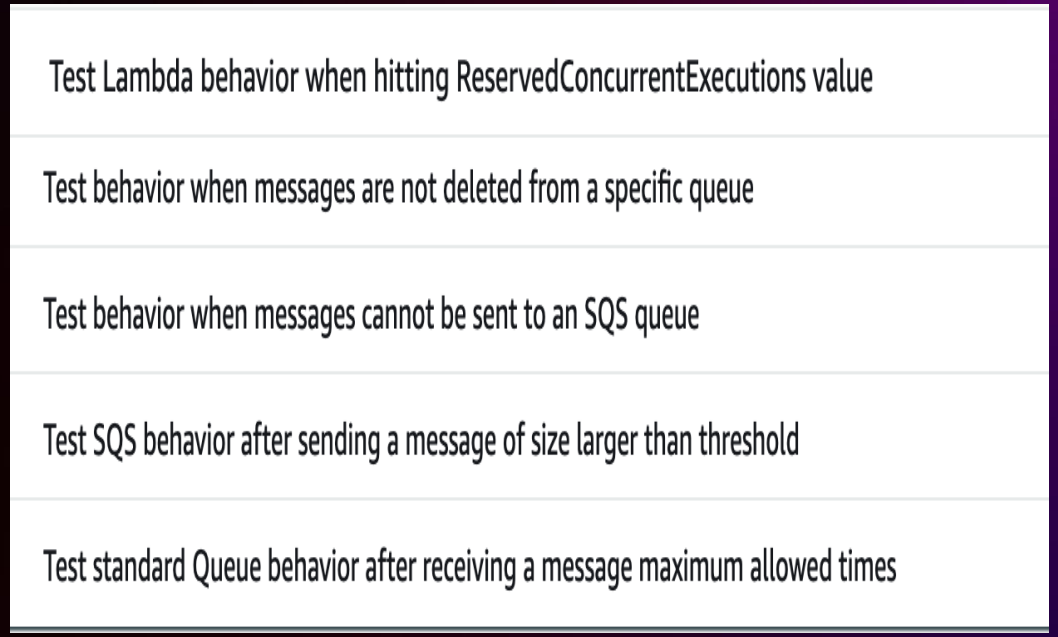


aws
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Lambda and SQS testing from Resilience Hub

With Lambda and SQS as part of an application, Resilience Hub recommends AWS Fault Injection Simulator (FIS) tests to verify resilience under stress

- Lambda thresholding
- SQS deletion failure
- Unsent messages
- Message size
- Allowable tries

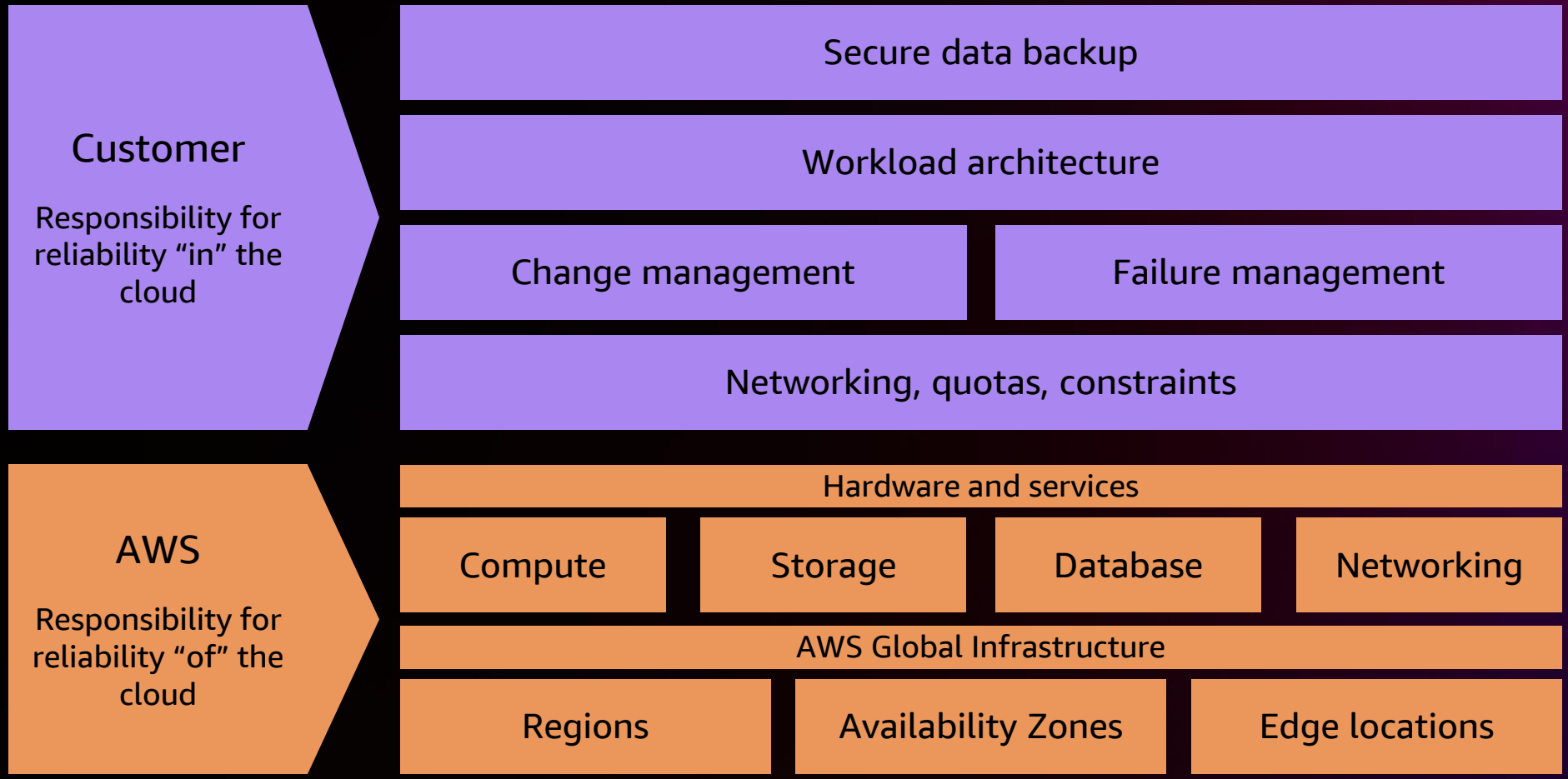


These are the most common tests.

Applications onboarded with Resilience Hub will receive new FIS tests as made available. Customers with existing FIS tests can augment with Resilience Hub generated tests.

aws
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

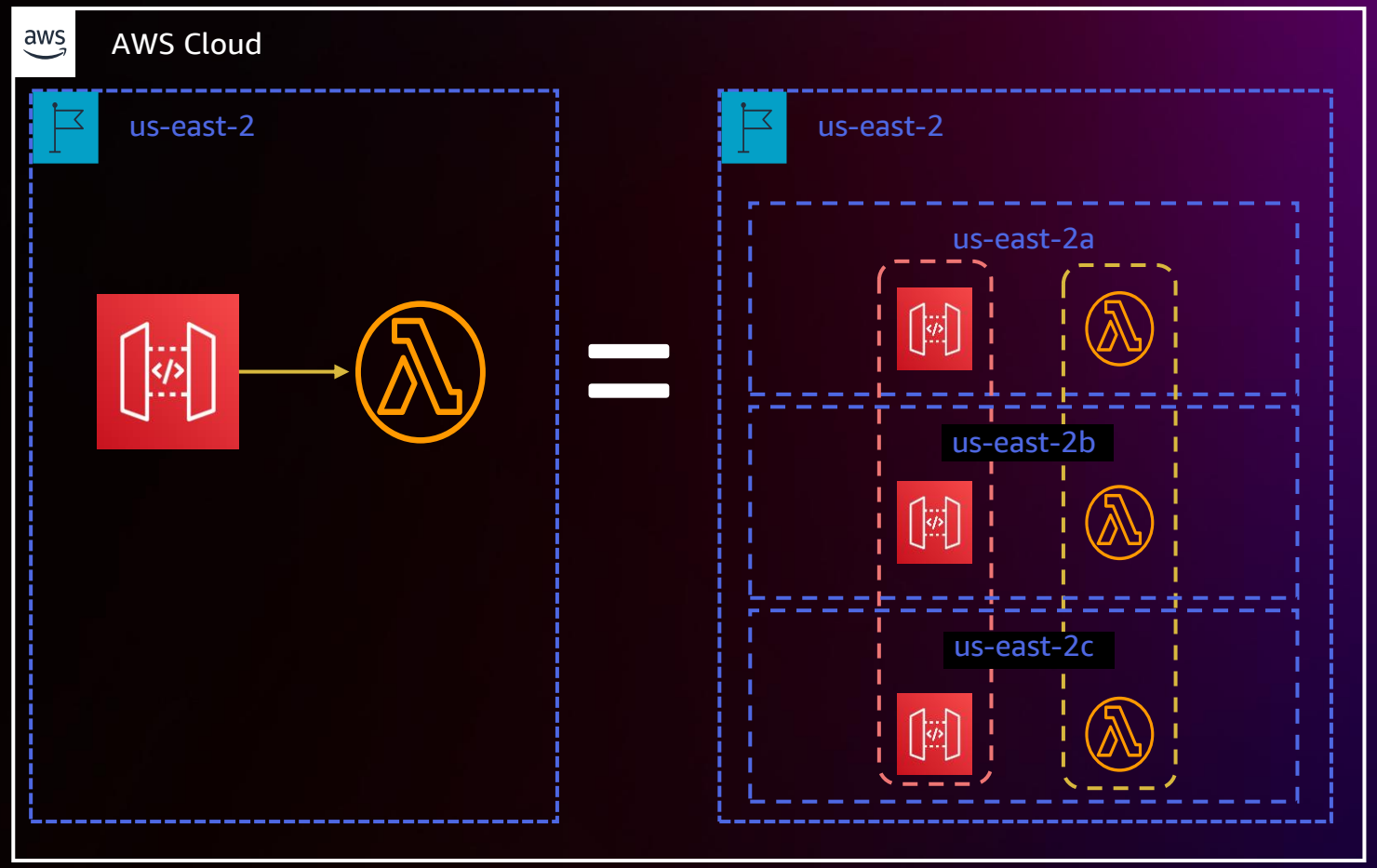
Shared responsibility for workload reliability



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

With Serverless, Multi-AZ comes for free

- Regional services:
  - High availability
  - Fault tolerant
- Service is responsible for:
  - Scaling
  - Health checks
  - Managing failure



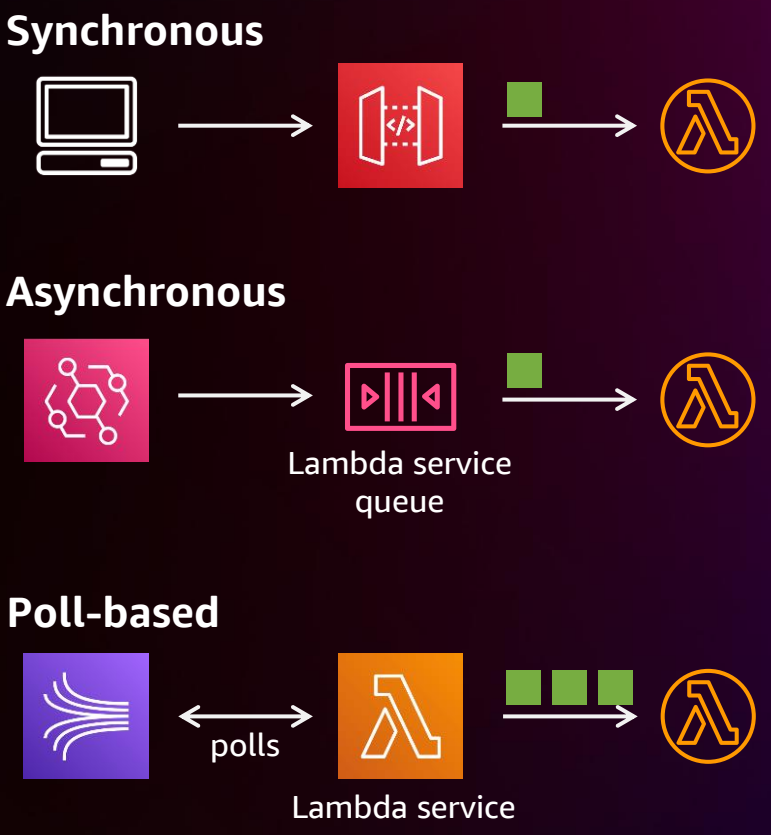
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

```
{
    “message”: “Internal Server Error”
}
```

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Lambda event source integrations

- Three types of Lambda integrations:
  - Synchronous
  - Asynchronous
  - Poll-based (event source mapping)
- Managed by configuration:
  - Destination/DLQ
  - Retry
  - Timeout
  - Batching



aws
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Synchronous invocation

Retry behavior: None

- Client must retry on failure



Error types

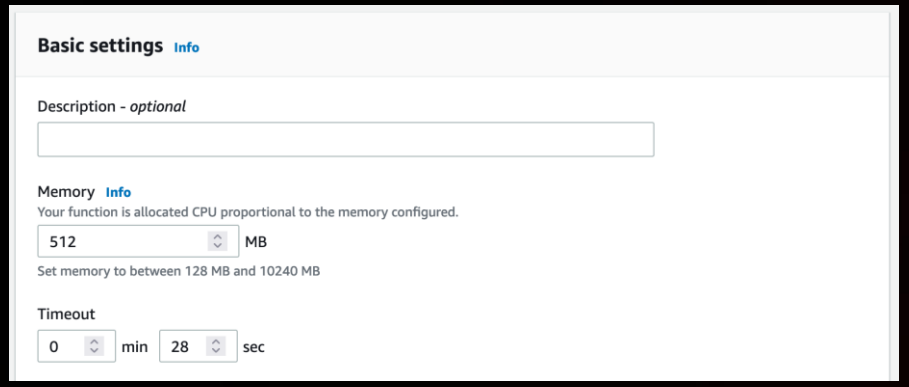
- Client (e.g., bad payload, record not found)
- Server/function (e.g., access denied, throttles)

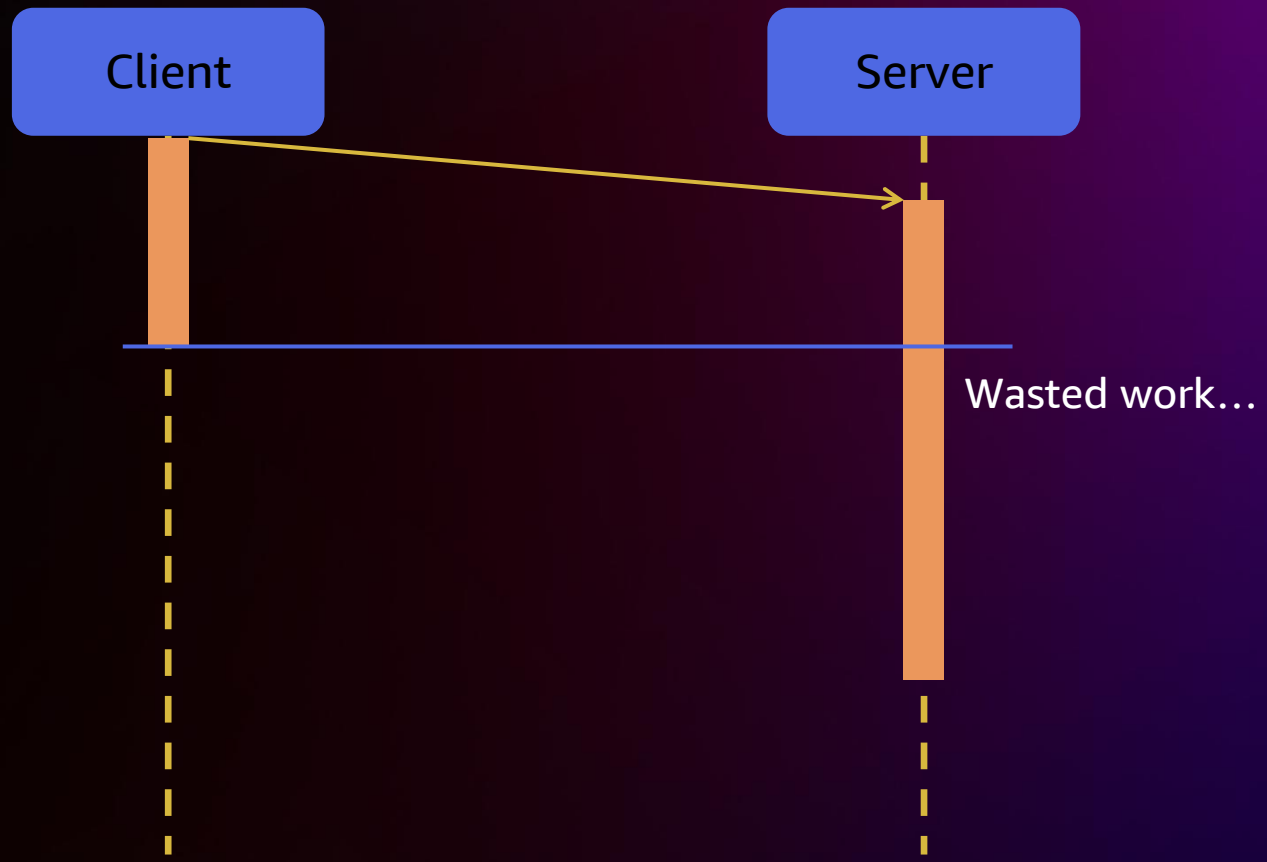
aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Don’t waste work

- Use timeouts to avoid expending resources when not necessary
- Consider source timeouts
  - API Gateway: 30 seconds





© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Asynchronous invocations

Retry behavior – 0–2 additional attempts

- Throttled/system errors – hours to days
- Charged as additional invocations



Configuration

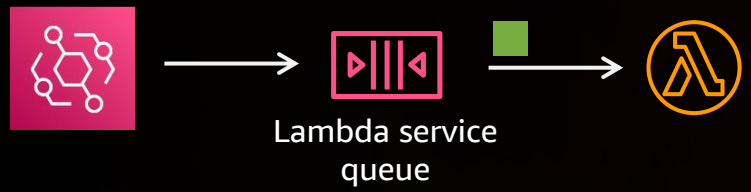
- `MaximumEventAge`
- `MaximumRetryAttempts`
- `Destination (success, failure)`

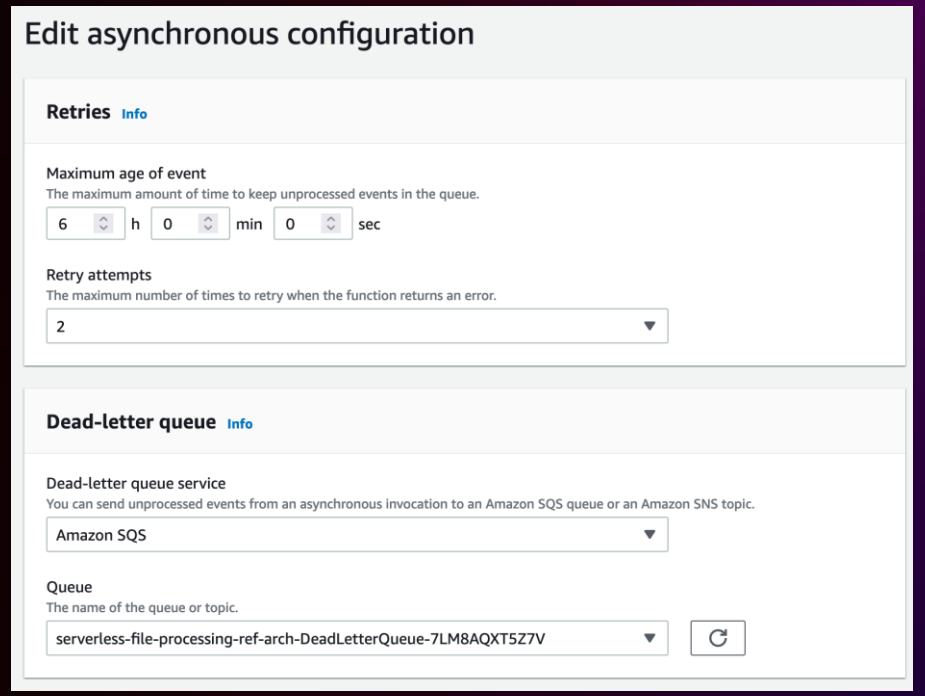
aws
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Don’t work on old messages

Amazon Builders’ Library

- Use **retry attempts** and **event age** to avoid working on old messages
- Dead-letter queues (DLQ) are used to catch unprocessed messages
  - Redrive later to process, if needed





© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Poll-based invocations

Retry behavior: Until message expires

- Retries entire batch of messages (default)

Configuration (varies by source)

- `MaximumRecordAge`
- `MaximumRetryAttempts`
- `BisectBatchOnFunctionError`



aws
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Handling poison pill in batch and insurmountable queues

- Shed excess load:
  - Maximum record age
  - Maximum retry attempts
  - Send to dead-letter queue
- Manage partial failures
  - Checkpoint
  - Split batch recursively
  - … use both together




```yaml
Events:
  KinesisEvent:
    Type: Kinesis
    Properties:
      Stream: !GetAtt Stream.Arn
      BisectBatchOnFunctionError: true # if error, split the batch in two and retry
      MaximumRecordAgeInSeconds: 3600  # one hour
      MaximumRetryAttempts: 2 # retry at most twice (bisect not included)
      StartingPosition: LATEST
      DestinationConfig:
        OnFailure:
          Destination: !GetAtt DeadLetterQueue.Arn
```

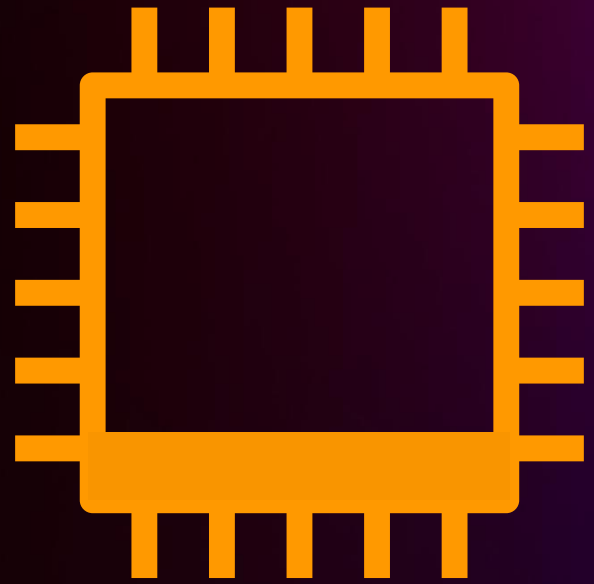
aws
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Scale horizontally to increase aggregate workload availability

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Resource exhaustion

- Servers are optimistic
- Finite resources, such as:
  - CPU
  - Memory
  - Storage
  - Network

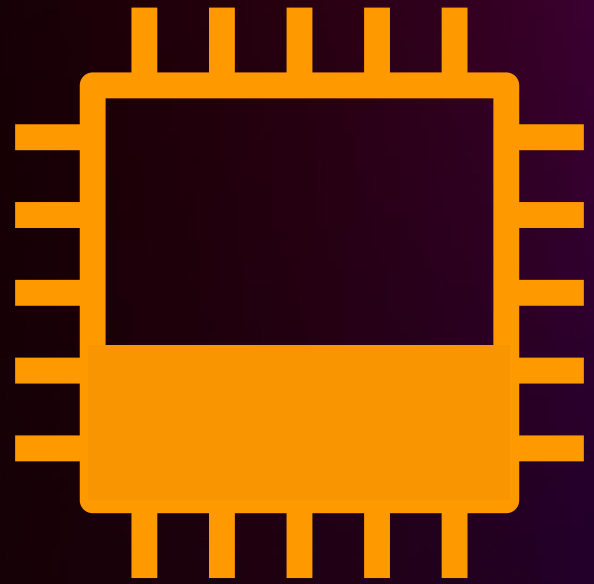


aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

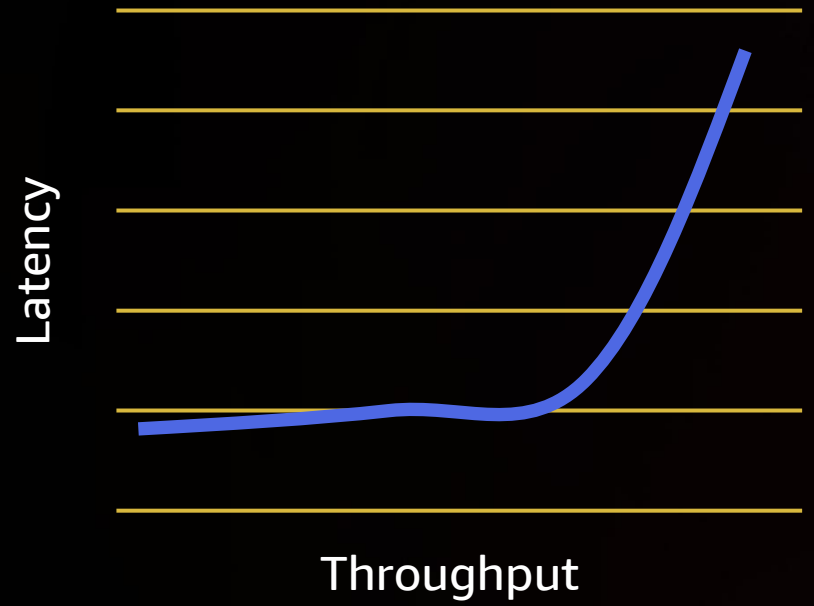
Resource exhaustion

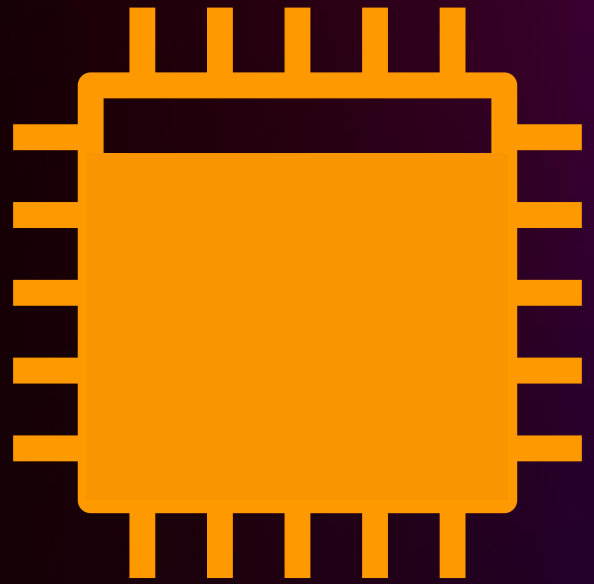
- As overall load increases, scale out horizontally
- Throughput on an exhausted instance starts to suffer
  - Latency may increase so much that clients timeout, resulting in retries



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Resource exhaustion: Degraded performance with load





aws
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Lambda has fixed resources per unit of work

- Each **function execution environment** has same finite resources:
  - CPU
  - Memory (up to 10GB)
  - Storage (512MB by default)
  - Network
- Each execution environment handles **one, and only one, event**
- Each request gets the same resources



aws
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

As work increases, scale concurrent environments*

Amazon Builders' Library

- Workload isolation means predictable performance
- Latency and throughput are constant





* Concurrency limits apply

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Scaling impact on downstream dependencies



What if we burst to 3000 concurrency?

aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Downstream services can cause upstream impact



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

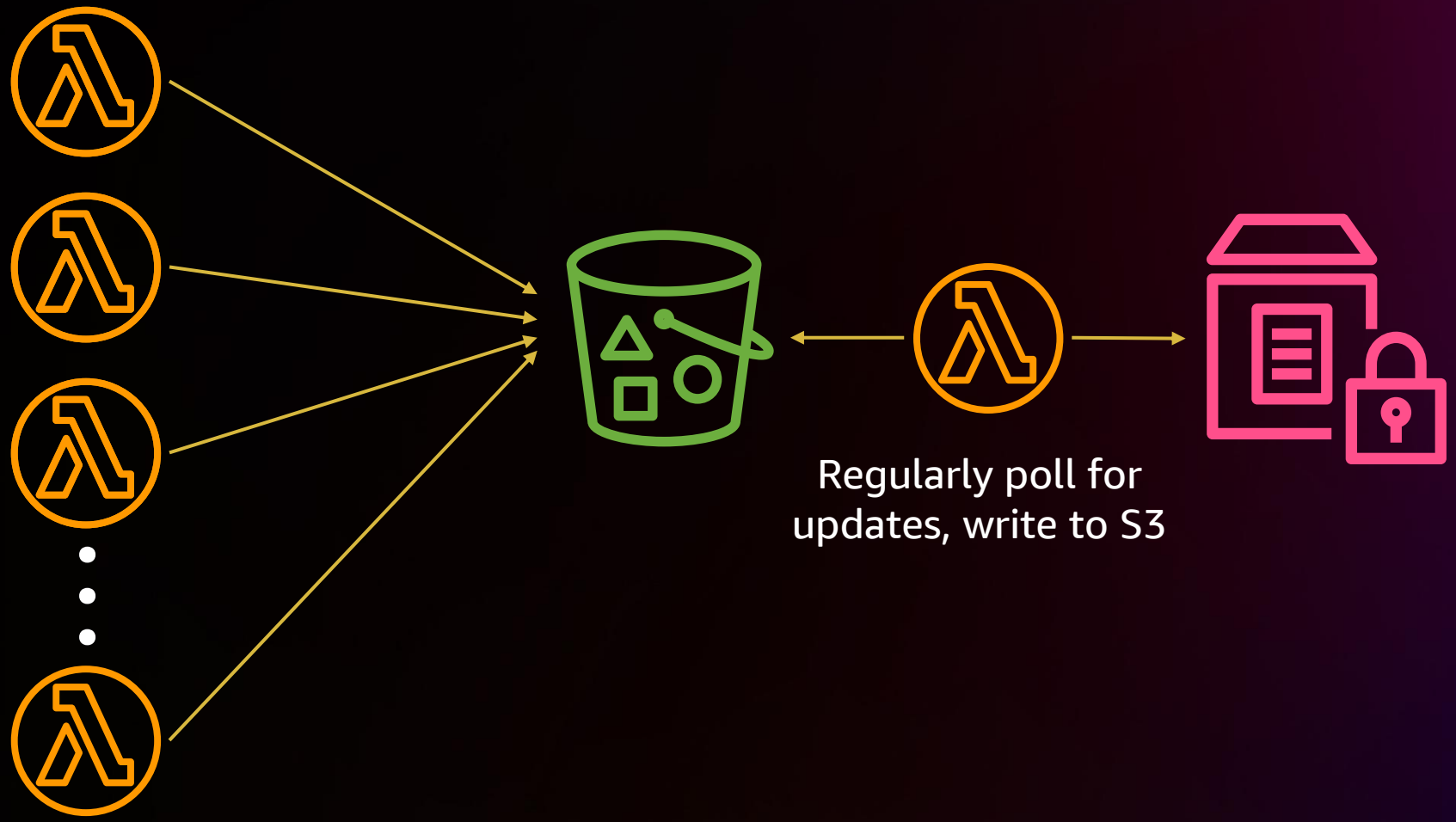
Downstream AWS services also have limits



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Managing scale mismatch

Amazon Builders' Library



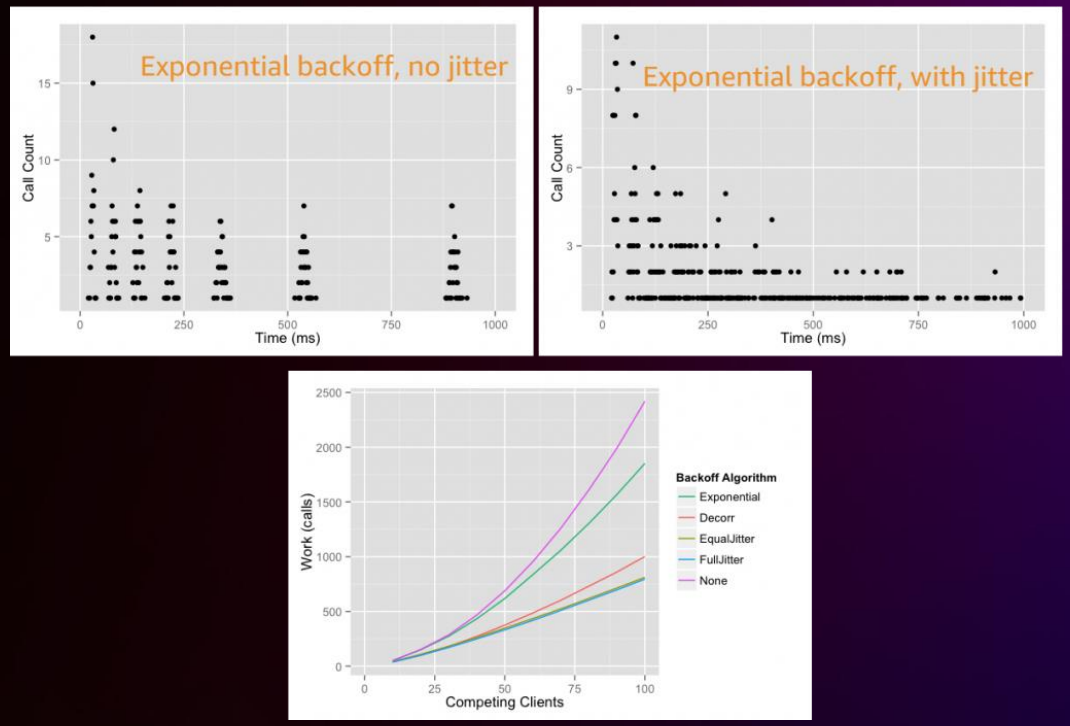
aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Implement retries with exponential backoff

- On error:
  - 5xx or throttling responses: retry the original request
  - 4xx: revise the request
- AWS SDKs include retry logic
  - Retries with exponential back-off and jitter to improve flow control
  - Configurable
- Timeout and/or adjust retry policy when appropriate

Amazon Builders' Library



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Why quotas/limits?

- Quotas and limits allow us to cheaply reject excess work
- Recommend implementing limits, particularly for synchronous workloads

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

“Load test your system.”

Someone, somewhere

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Stop guessing at capacity

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Connective tissue of modern applications



Event

A signal that
status has changed

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Event-driven architectures drive scalability and reliability



Event routers

Abstract producers and consumers from each other



Asynchronous events

Improve responsiveness and reduce dependencies



Event stores

Buffer messages until services are available to process

aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.



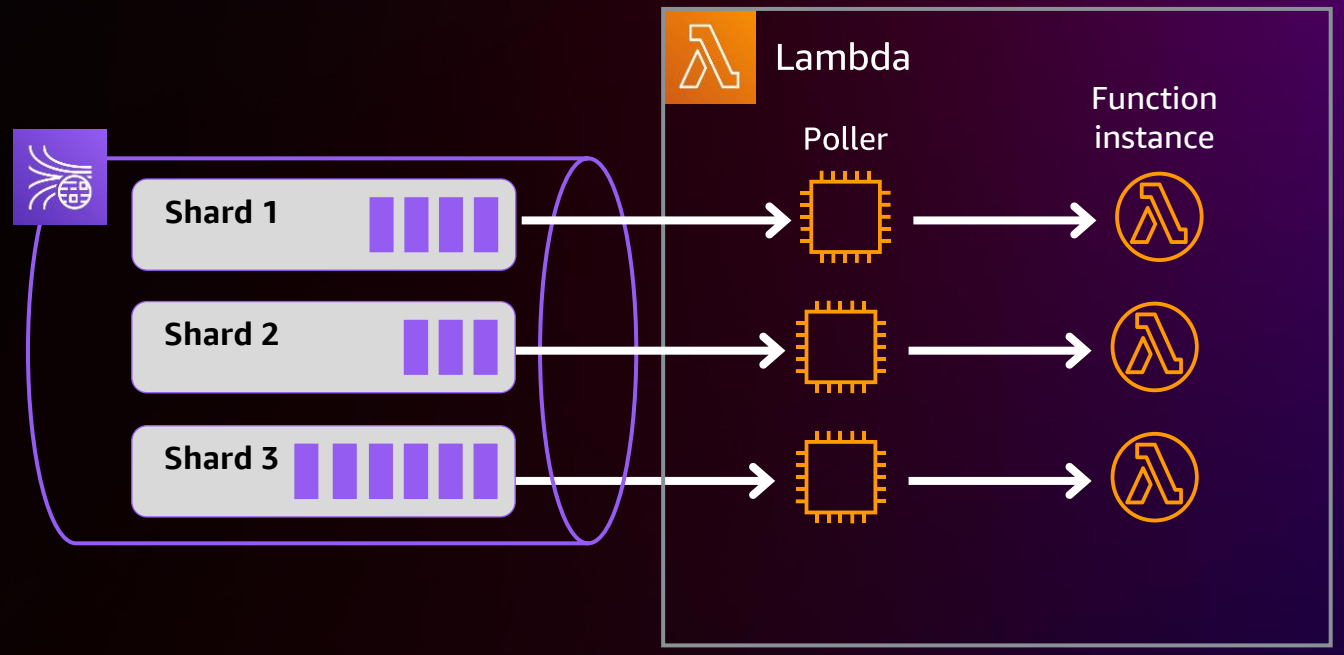
AWS Lambda

Event-driven
serverless compute

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Lambda function instances scale with Kinesis

- Function instance per shard by default
  - Maintains message ordering
  - Need more? Increase parallelization factor
- Lambda service manages polling, error handling of stream



aws
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Monitoring

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Take action with Amazon CloudWatch

AWS metrics

- AWS Lambda
  - Errors, throttles, iterator age, concurrency
- Amazon API Gateway
  - Success rate
  - Latency (tail p90, p95, p99)
- Amazon SQS
  - Message age

Custom metrics

- Use embedded metric format (EMF) to generate custom metrics:
  - Business, customer metrics
  - Operational metrics
  - Errors

aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

AWS X-Ray enables tracing of distributed applications

- Scales for microservice and serverless architectures
- Identifies the root cause of performance issues and errors
- Provides a cross-service view of requests made to application
  - Aggregates data across services into a trace (via a passed trace identifier)
  - Support for Lambda, API Gateway, SNS, Step Functions, DynamoDB, S3, etc.



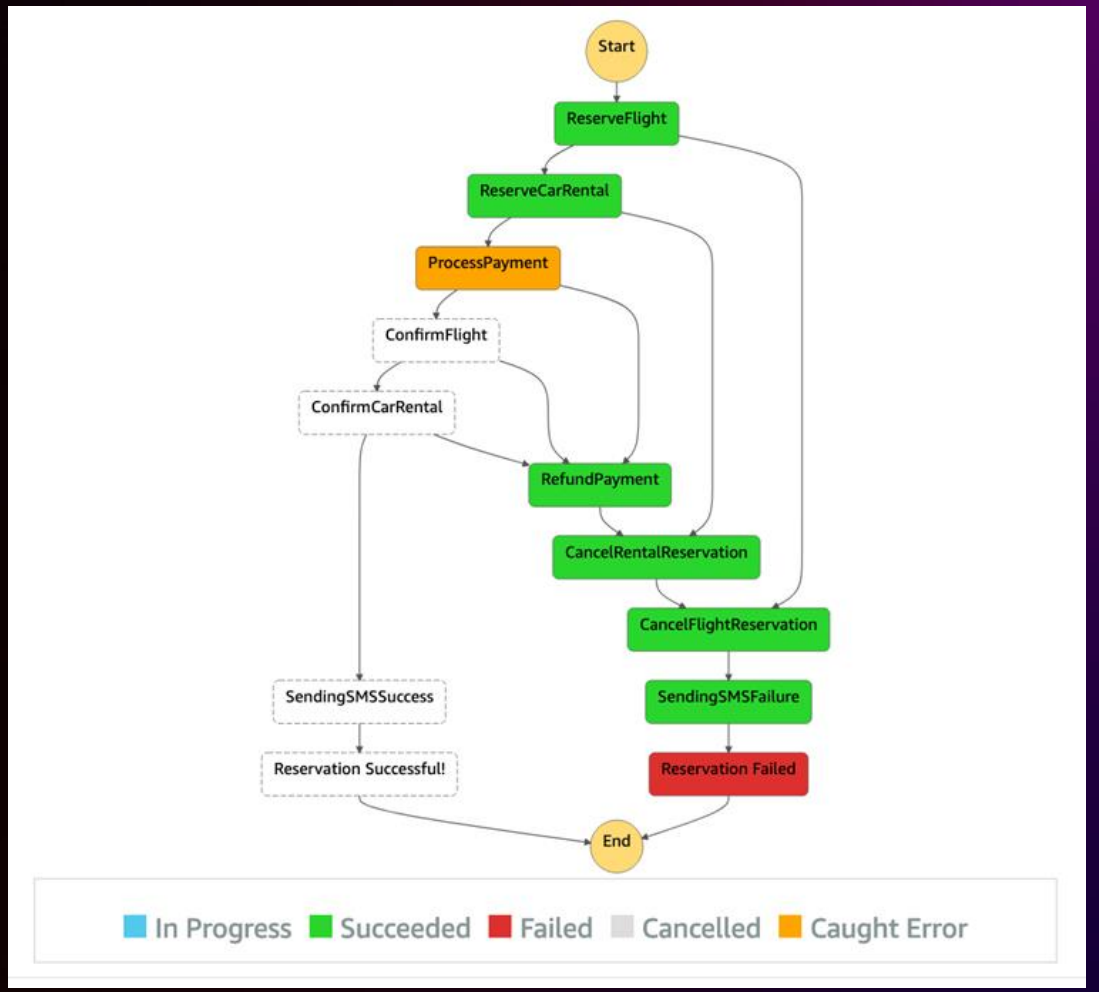
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Architectural patterns

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Manage rollback of distributed transactions

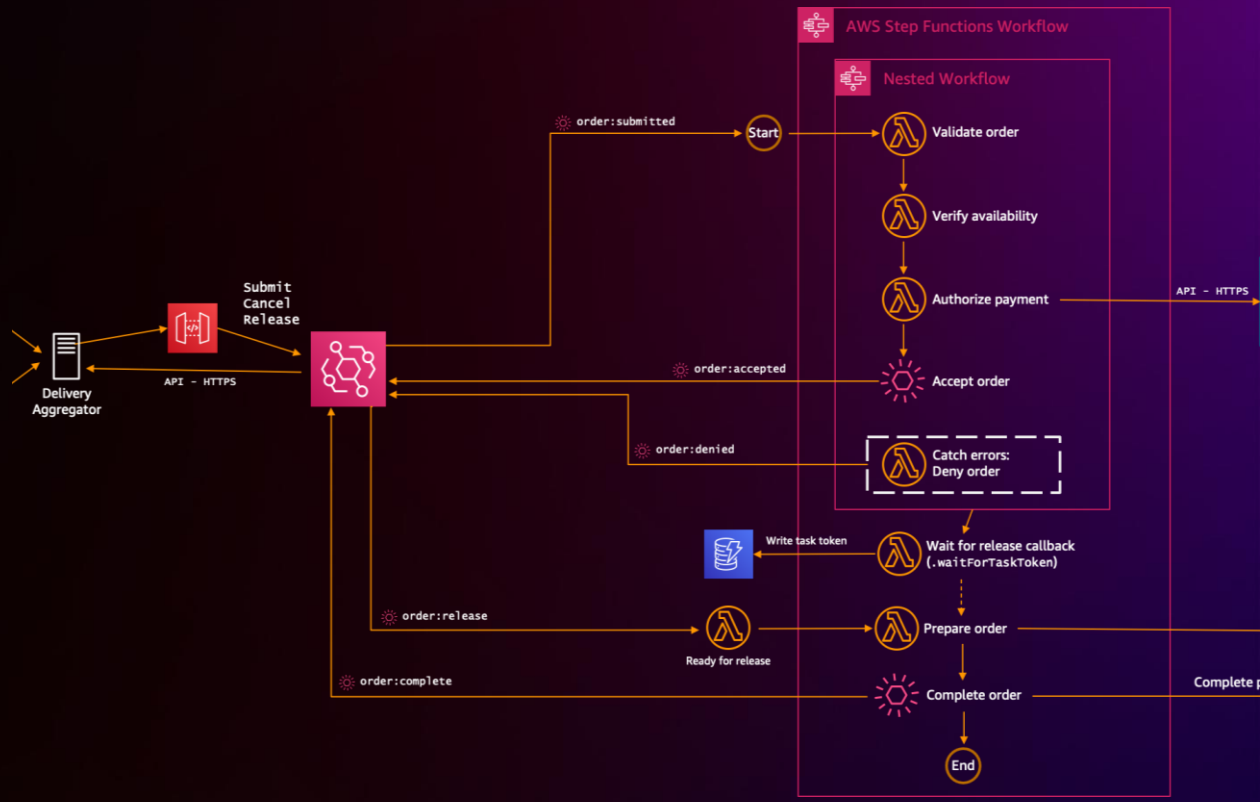
- Implement SAGA pattern using AWS Step Functions as coordinator
  - Reverse each action with compensating rollback



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Leverage event-driven architectures

- Enhance **scalability** and **reliability**:
  - **Orchestrate** processes within service boundaries
  - **Choreograph** cross-service interactions via events
- Review message delivery semantics:
  - Idempotence of transactions?

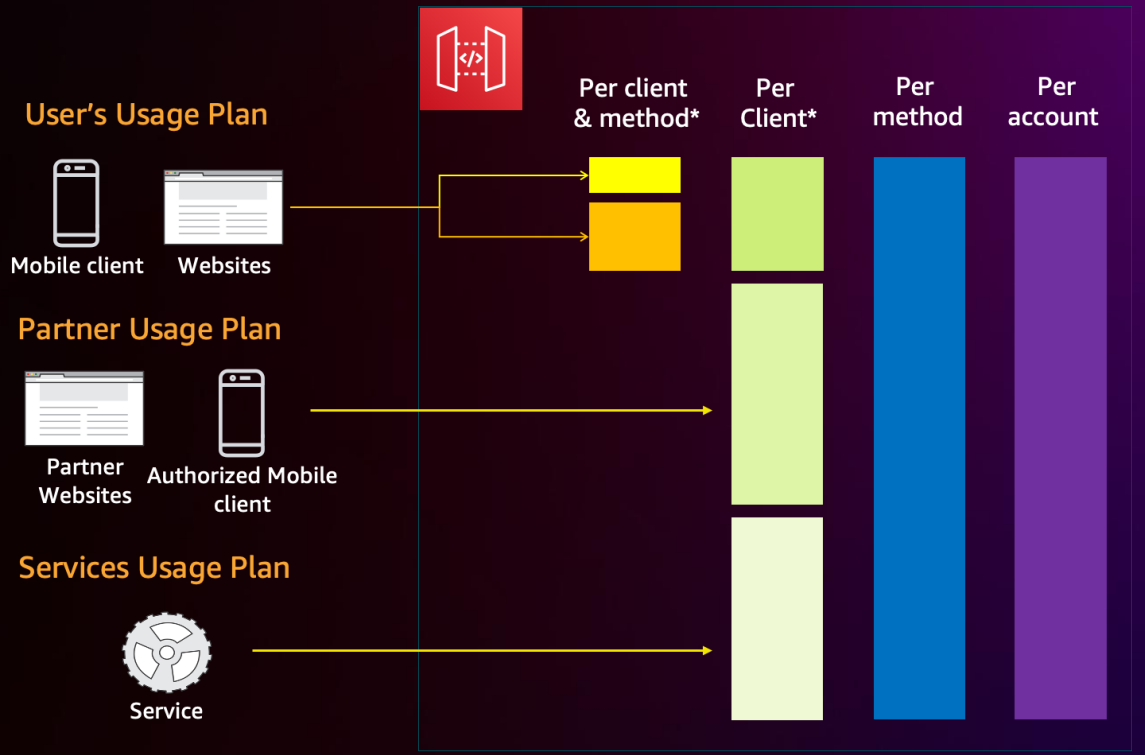


© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Reject excess work with API Gateway throttling

- **Protect backend systems**
- Allocate capacity across API consumers
- Rate-limit consumers

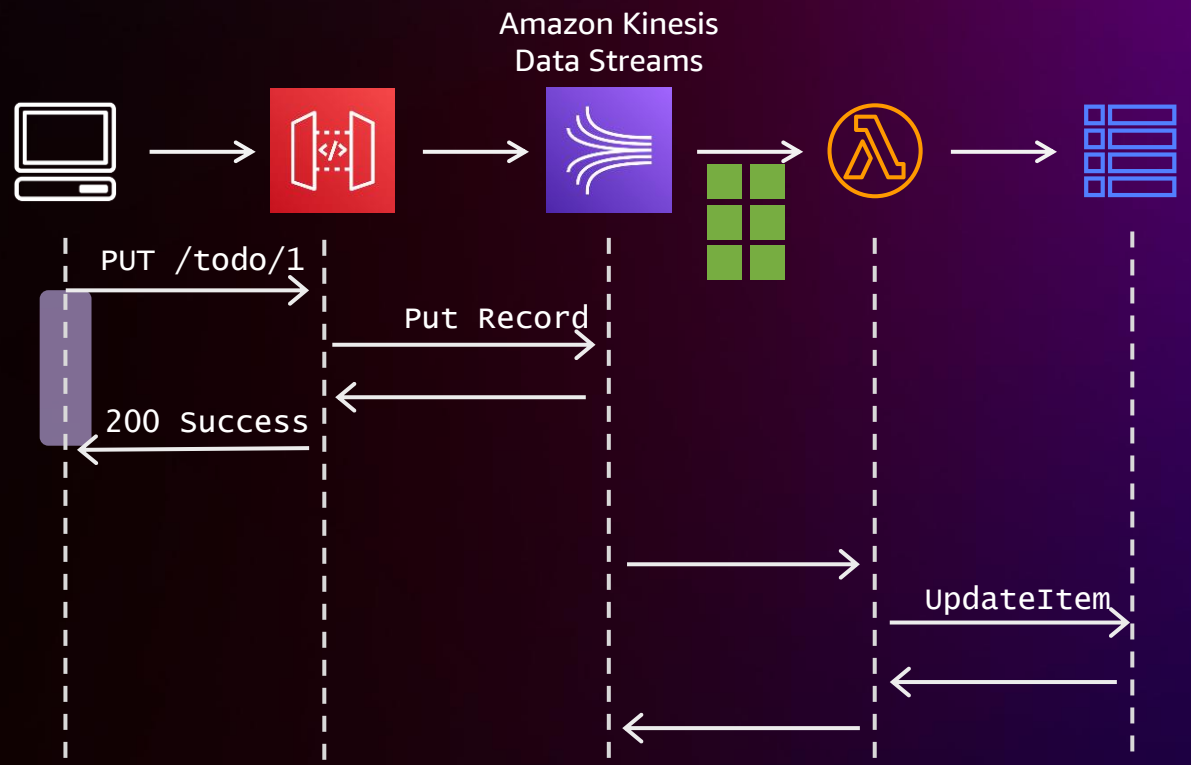
- Also consider **bounding work** by:
  - Validating input size
  - Paginating results



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Enhance durability with asynchronous patterns

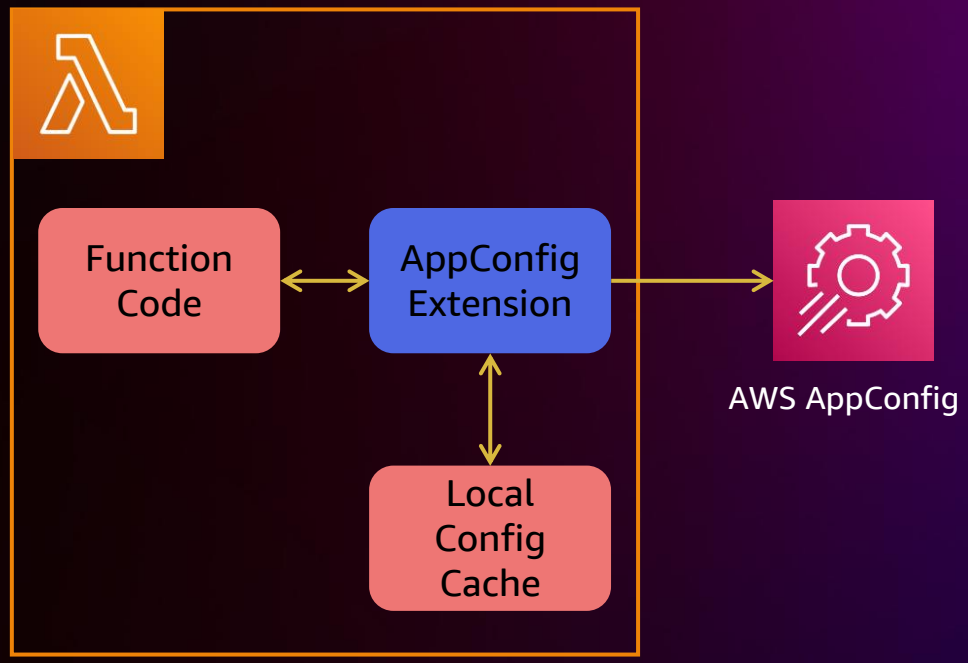
- Response to client is “message received”
  - Asynchronous processing
  - Process can be long running (beyond 30s)
  - Messages delivered in batches
  - Consider UI requirements
- Higher resiliency, durability
- Checkpoint streams
- Built-in retries, configurable to use case



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Disable features with feature flags (or a kill switch)

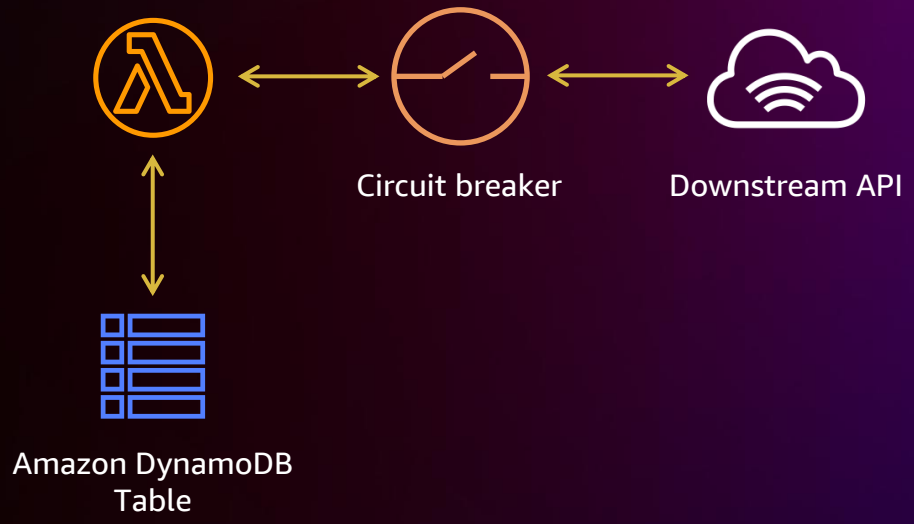
- Modify function behavior **without changing code**
- Safely disable breaking or unstable features without additional deployment
- **AWS AppConfig** allows you to create, manage, and quickly deploy application configurations



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Fail fast with circuit breaker

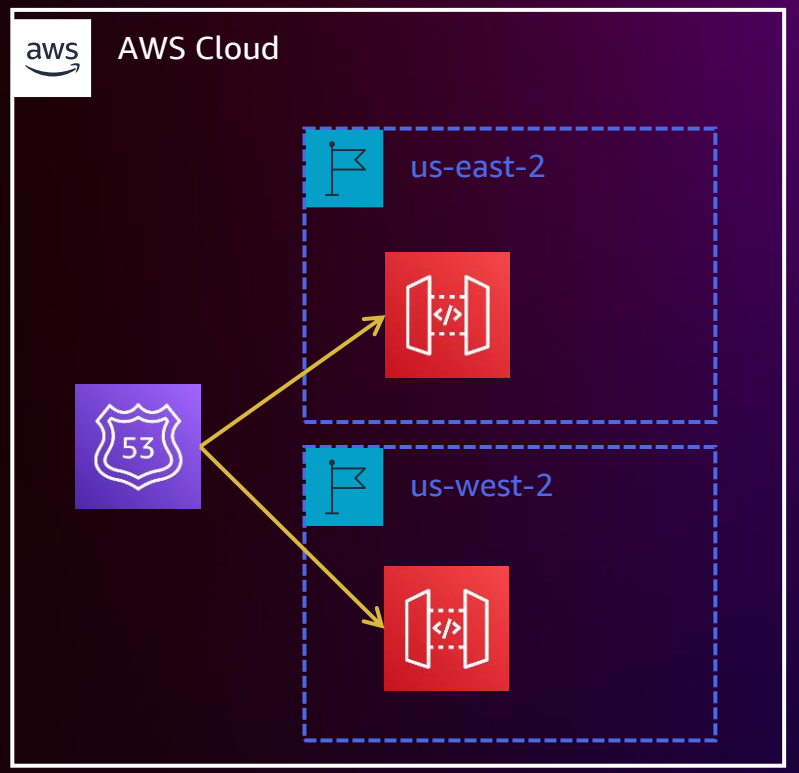
- Handle **slow** or **failing** remote calls
- Monitor remote calls for failures:
  - Trip (open) circuit at threshold
  - Further remote calls fail, no attempt
  - Periodically check dependency health to close
- Store breaker status in memory or persist in database (e.g., DynamoDB)
  - Centralization versus latency trade-off



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

What about multi-region?

- Customers choose multi-region for:
  - Increased **availability** (HA, DR)
  - Improved **global performance** for low latency workloads
  - **Regulatory** requirements
- **CAP theorem** applies
- With serverless, patterns differ by workload type
- Introduces **technical complexity**
  - Infrastructure failover and **health** checks
  - Data **consistency**, infrastructure redundancy
  - **Deployment** and **testing**



aws
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.