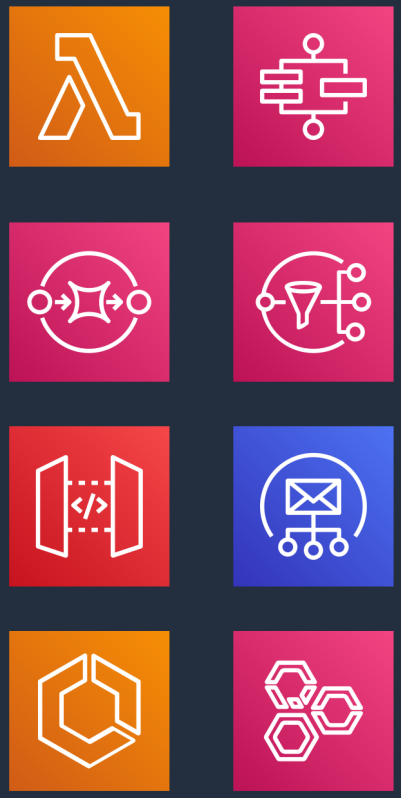
Serverless

AWS Certified Solutions Architect – Associate

Harun Ur Rashid – Cloud Infrastructure Architect
27th of August 2021

Agenda

- What is *Serverless*?
- AWS Lambda and AWS Step Functions
- Amazon Simple Queue Service (SQS)
- Amazon Simple Notification Service (SNS)
- Amazon API Gateway
- Amazon Simple Email Service (SES)
- Amazon Elastic Container Service (ECS)
- AWS Well Architected Framework



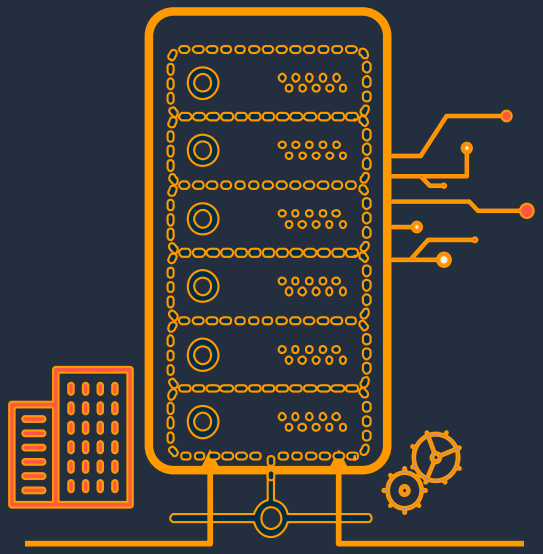
aws

What is Serverless?

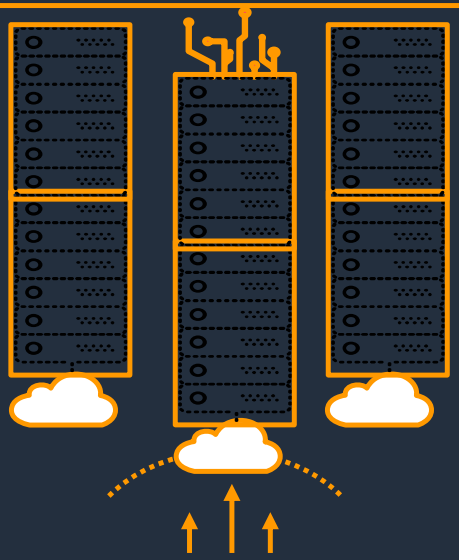
Evolution of computing

Physical Servers in Datacenters

Virtual Servers in Datacenters



Virtual Servers in the Cloud



aws

What is *Serverless*?

No infrastructure provisioning, no management

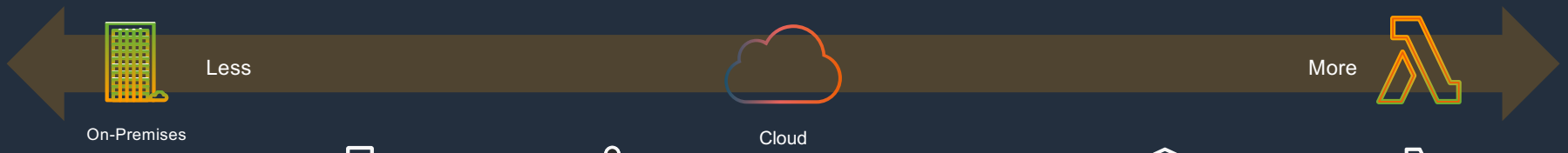
Automatic scaling

Pay for value

Highly available and secure

aws

AWS operational responsibility models



Compute	Virtual Machine	EC2	Elastic Beanstalk		Fargate	AWS Lambda
Databases	MySQL	MySQL on EC2	RDS MySQL	RDS Aurora	Aurora Serverless	DynamoDB
Storage	Storage					S3
Messaging	ESBs		Amazon MQ	Kinesis		SQS / SNS
Analytics	Hadoop	Hadoop on EC2	EMR	Elasticsearch Service		Athena

aws

AWS Lambda

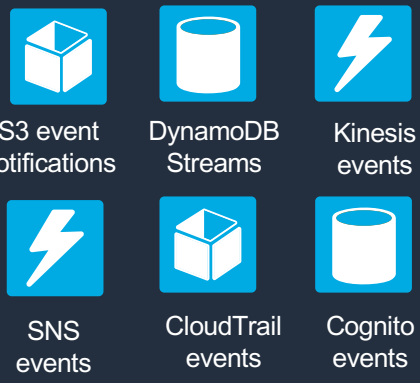
Introduction to AWS Lambda

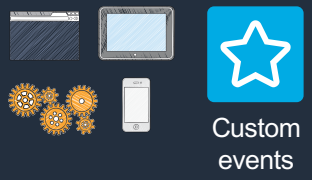
- "Function-as-a-Service"
- Run code without provisioning or managing servers
  - Java, Go, PowerShell, Node.js, C#, Python, and Ruby
  - Runtime API which supports additional programming languages
- Pay only for the compute time you consume
- Automatically runs your code with high availability
- Scale with usage

aws

How Lambda works

Invoked in response to events
- **Changes in data**
- **Changes in state**





Author in familiar language using any libraries; Execute only when needed, automatic scale

"Lambda functions"

Access any service, including your own

Any custom

Any AWS

Such as…

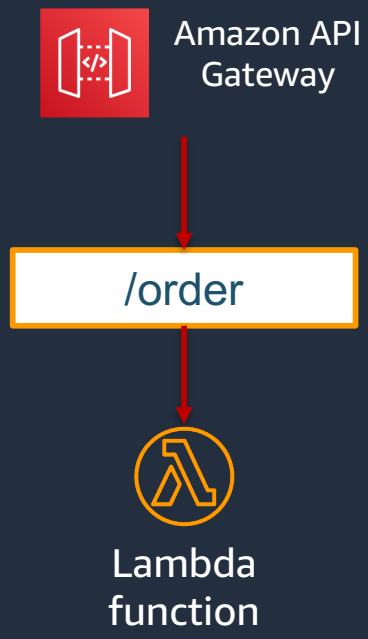
SNS DynamoDB Lambda

Redshift Kinesis S3

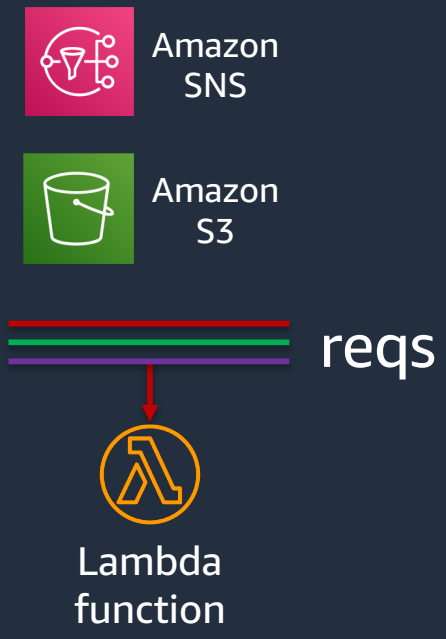
aws

Lambda execution models

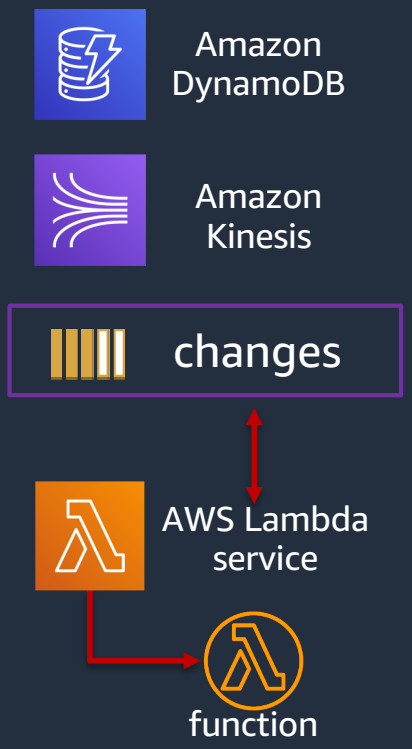
Synchronous (push)



Asynchronous (event)



Stream (Poll-based)



aws

AWS event sources that can trigger Lambda

Synchronous (push):
Elastic Load Balancing (Application Load Balancer)
Amazon Cognito
Amazon Lex
Amazon Alexa
Amazon API Gateway
Amazon CloudFront (Lambda@Edge)
Amazon Kinesis Data Firehose
Amazon Simple Storage Service Batch

Stream (poll):
Amazon DynamoDB
Amazon Kinesis
Amazon Managed Streaming for Apache Kafka
Amazon Simple Queue Service

Asynchronous (event):
Amazon Simple Storage Service
Amazon Simple Notification Service
Amazon Simple Email Service
AWS CloudFormation
Amazon CloudWatch Logs
Amazon CloudWatch Events
AWS CodeCommit
AWS Config
AWS IoT
AWS IoT Events
AWS CodePipeline

aws

Anatomy of a Lambda function

Handler function

Function to be executed upon invocation

Event object

Data sent during Lambda function invocation

Context object

Methods available to interact with runtime information (request ID, log group, more)

```
import json

def lambda_handler(event, context):
    # TODO implement
    return {
        'statusCode': 200,
        'body': json.dumps('Hello World!')
    }
```


aws

Anatomy of a Lambda function – Example

Input

```
{
  "my_number": 5
}
```

Response

```
Response:
{
  "statusCode": 200,
  "body": "25"
}
```

```
import json

def lambda_handler(event, context):
    my_number = event["my_number"]
    my_number_squared = my_number * my_number
    return {
        'statusCode': 200,
        'body': json.dumps(my_number_squared)
    }
```


aws

Lambda Layers

Lets functions easily share code: Upload layer once, reference within any function

Promote separation of responsibilities, lets developers iterate faster on writing business logic

Built in support for secure sharing by ecosystem

aws

Additional Lambda configurations

- Assignable memory in 64MB increments, from 128MB to 3GB
- Configured to run up to 15 minutes per execution
- Environment variables
- Reserved and provisioned concurrency
- Amazon VPC, Security Groups, and RDS Proxy
- Stateless /tmp storage, and stateful filesystem using Amazon EFS
- Publish new versions of functions, and alias them
- AWS Serverless Application Model (AWS SAM)
- Lambda @ Edge

aws

Poll#1

- Which of the following services can invoke a Lambda function synchronously (with functionality built-in with the invoking service)?
- A: IAM
- B: Amazon Lex
- C: Amazon Kinesis Data Firehose
- D: API Gateway
- E: S3

aws

Poll#2

- What is the maximum amount of memory you can allocate to an AWS Lambda function?
- A: 128MB
- B: 512MB
- C: 1GB
- D: 3GB

aws

AWS Step Functions

State machine

Describes a collection of computational steps split into discrete states

Has one starting state and always one active state (while executing)

The active state receives input, takes some action, and generates output

Transitions between states are based on state outputs and rules that we define

aws

AWS Step Functions

Fully-managed state machines on AWS

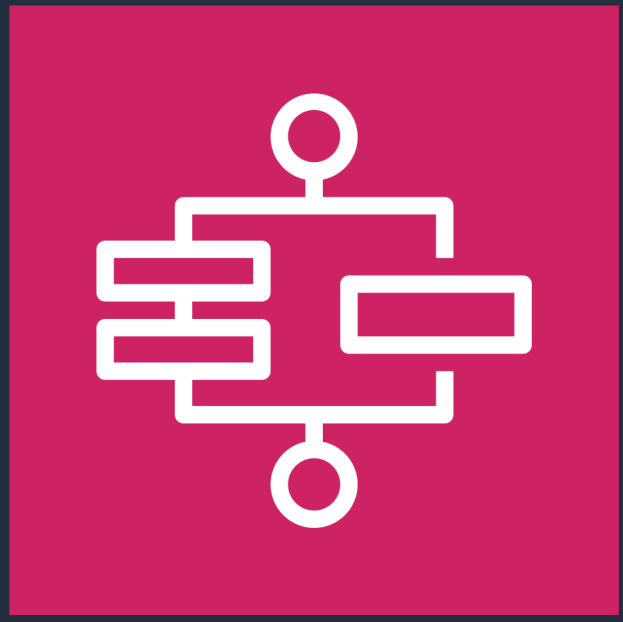
Resilient workflow automation

Built-in error handling

Powerful AWS service integration

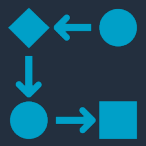
First-class support for integrating with your own services

Auditable execution history and visual monitoring



aws

How AWS Step Functions work



The workflows you build with Step Functions are called state machines, and each step of your workflow is called a state.

When you execute your state machine, each move from one state to the next is called a state transition.

You can reuse components, easily edit the sequence of steps or swap out the code called by task states as your needs change.

aws

Amazon State Language

https://states-language.net/spec.html

```
{
    "Comment": "A simple minimal example",
    "StartAt": "Hello World",
    "States": {
     "Hello World": {
       "Type": "Task",
       "Resource": "arn:aws:lambda...HelloWorld",
       "End": true
     },
     [. . .]
  }
}
```

aws

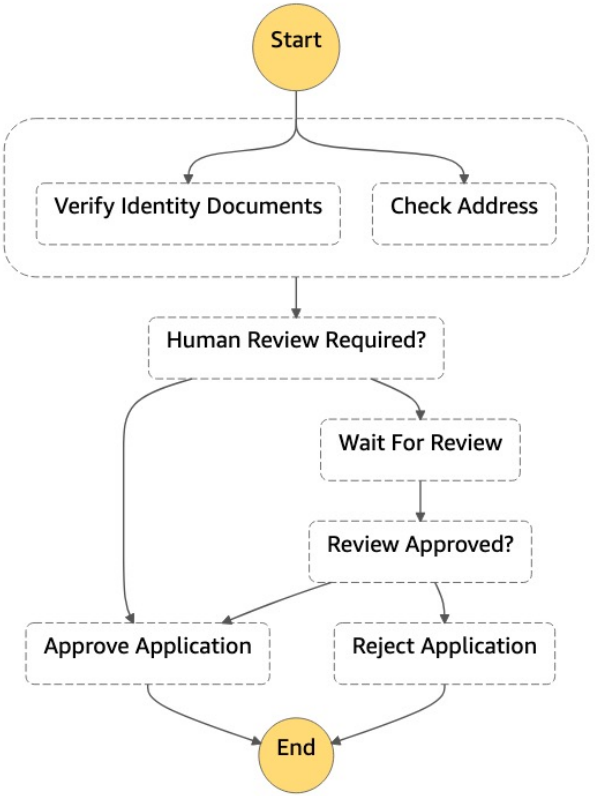
Example workflow: opening an account

Tasks

Parallel Steps

Branching Choice

Wait for a callback

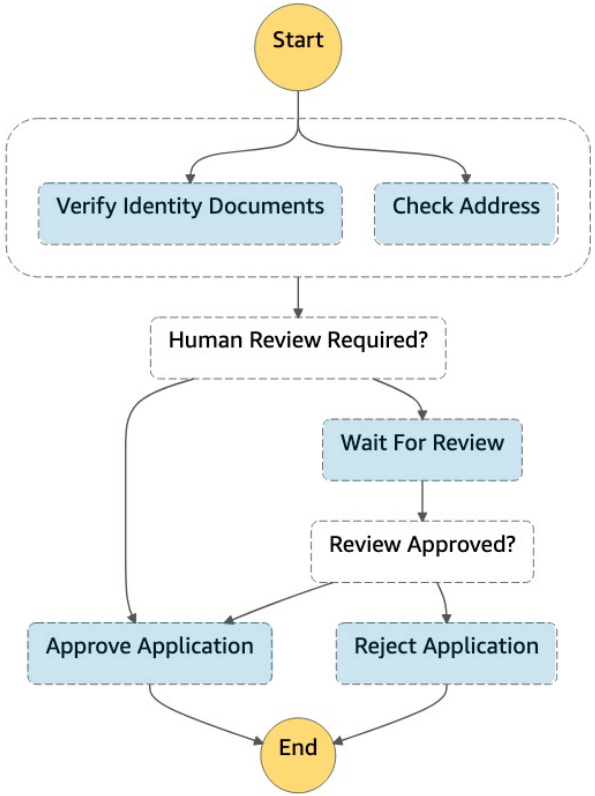


Performing a task

Call an AWS Lambda Function

Wait for a polling worker to perform an activity

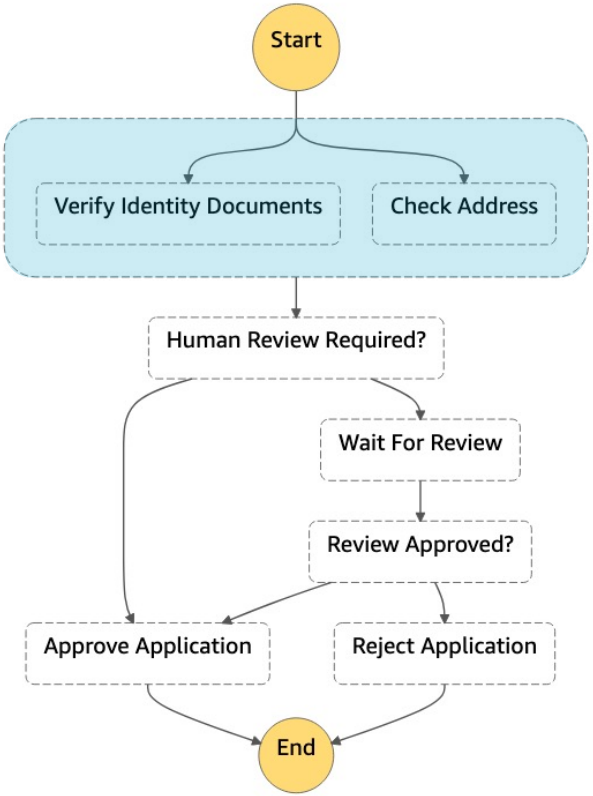
Pass parameters to an API of an integrated AWS Service



Executing branches in parallel

Contains an array of state machines branches to execute in parallel

Outputs an array of outputs from each state machine in its branches

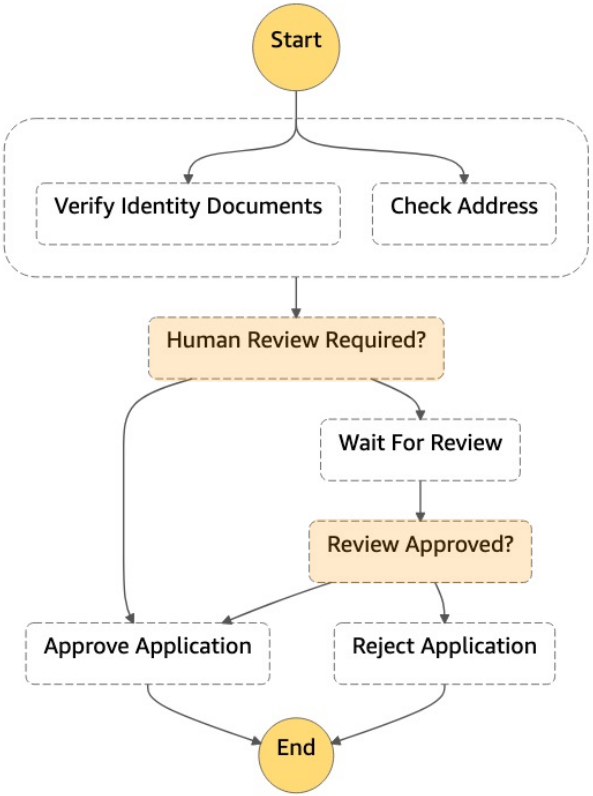


Making a choice

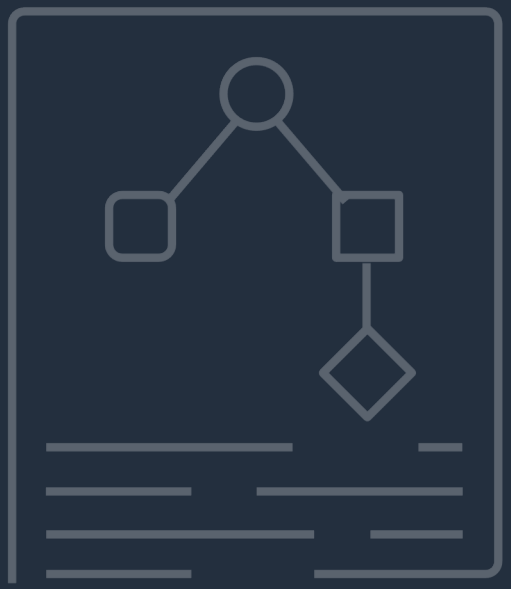
Like a switch statement in programming

Inspects an array of *choice* expressions, comparing variables to values

Determines which state to transition to next



State types



Task	Execute work
Choice	Add branching logic
Wait	Add a timed delay
Parallel	Execute branches in parallel
Map	Process each of an input array's items with a state machine
Succeed	Signal a successful execution and stop
Fail	Signal a failed execution and stop
Pass	Pass input to output

aws

AWS Step Functions service integrations

AWS
Lambda

Amazon
Elastic Container Service

AWS
Batch

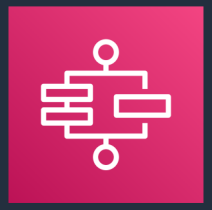
AWS CodeBuild

Amazon
DynamoDB

AWS
Glue

Amazon
SageMaker

Amazon EMR



AWS
Step Functions

Amazon
Simple Notification Service

Amazon
Simple Queue Service

Amazon Athena

aws

Poll#3

- AWS Step Functions can orchestrate multiple AWS Glue Jobs – True or False?
- A: True
- B: False

Amazon Simple Queue Service (SQS)

What is a “queue”?

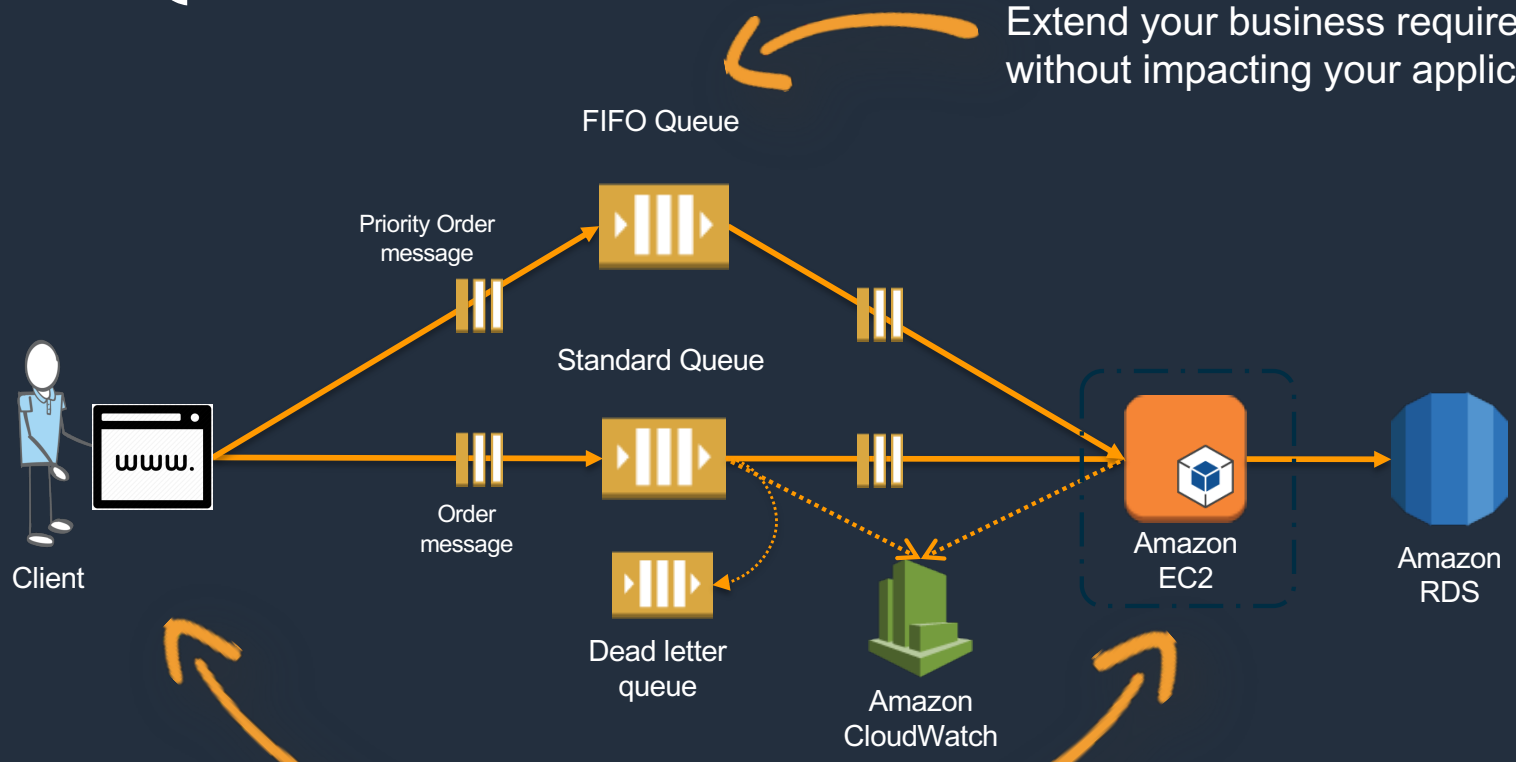
aws

Amazon SQS overview

- Fast, reliable, scalable, fully managed queue service
- Send, store, and receive messages between components
- Consumer polls for messages
- AWS SDK or Java Message Service (JMS) APIs
- Pay for what you use

aws

Amazon SQS



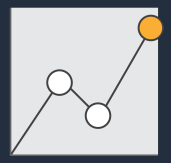
Key features

- Persistent message queue with high durability and availability
- Messages are stored across multiple AZs

- Messages retained until deleted— up to 14 days
- Nearly unlimited throughput

aws

Key features



- Amazon CloudWatch metrics and alerts

- Message payloads up to 256 KB (without Extended Client Library for Java)

- Message batching to increase throughput and reduce cost

- Secure: uses AWS Identity & Access Management (IAM) and HTTPS/TLS

aws

SQS queue types

Standard Queues

- At least once delivery
- Best-effort ordering
- Nearly unlimited transaction rate

FIFO Queues

- First-in First-out delivery preserving message ordering
- Exactly once processing
- 300 transactions per second
- Messages groups for multiple ordered streams

aws

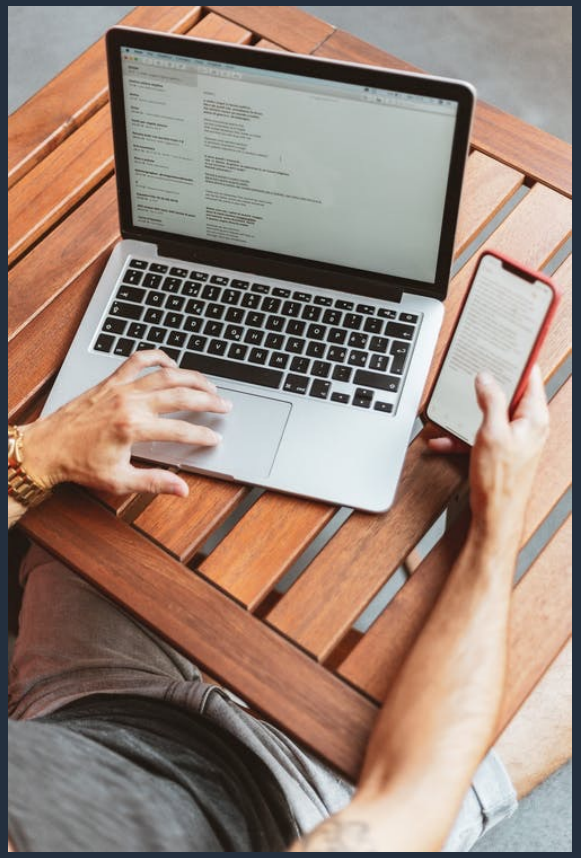
Poll#4

- Which of the following SQS Queue preserve String Ordering in message processing?
- A: Standard Queues
- B: FIFO Queues

aws

Amazon Simple Notification Service (SNS)

What is “pub-sub”?



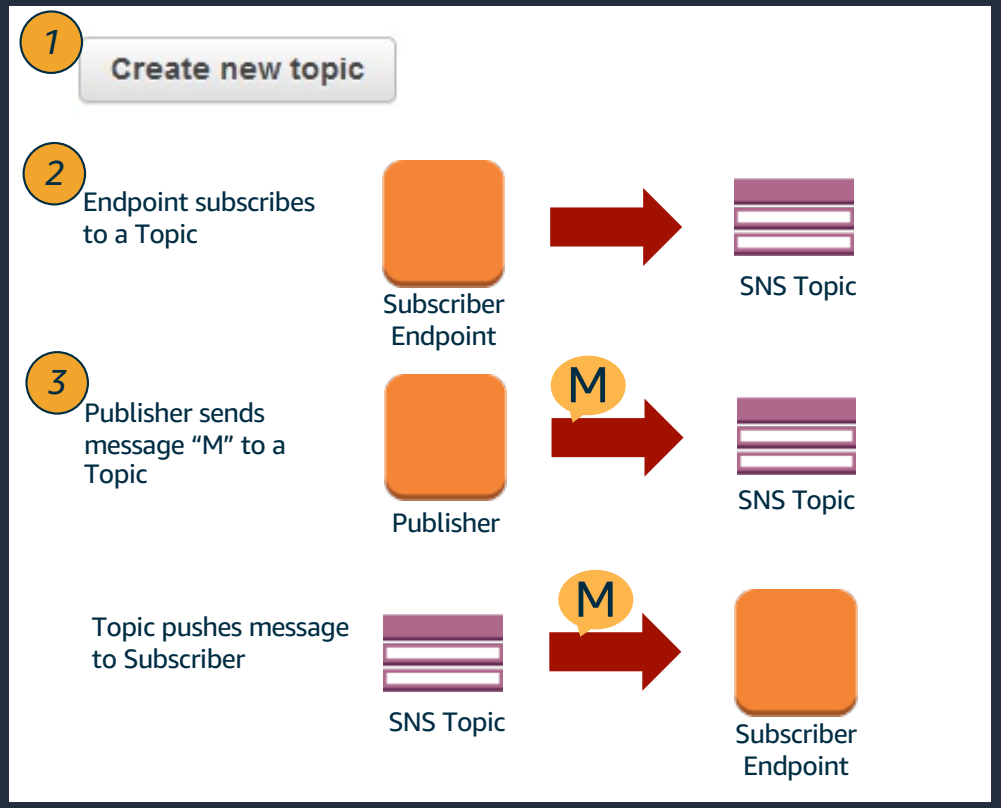
What is Amazon SNS?

- Amazon SNS is a web service that coordinates and manages the delivery or sending of messages to subscribing endpoints or clients
- Two types of clients—publishers and subscribers—also referred to as producers and consumers
- Publishers communicate asynchronously with subscribers by producing and sending a message to a topic
- Subscribers like web servers, email addresses, Amazon SQS queues, AWS Lambda functions consume or receive the message or notification through:
  - Amazon SQS, HTTP/S, email, SMS, Lambda

aws

How Amazon SNS works

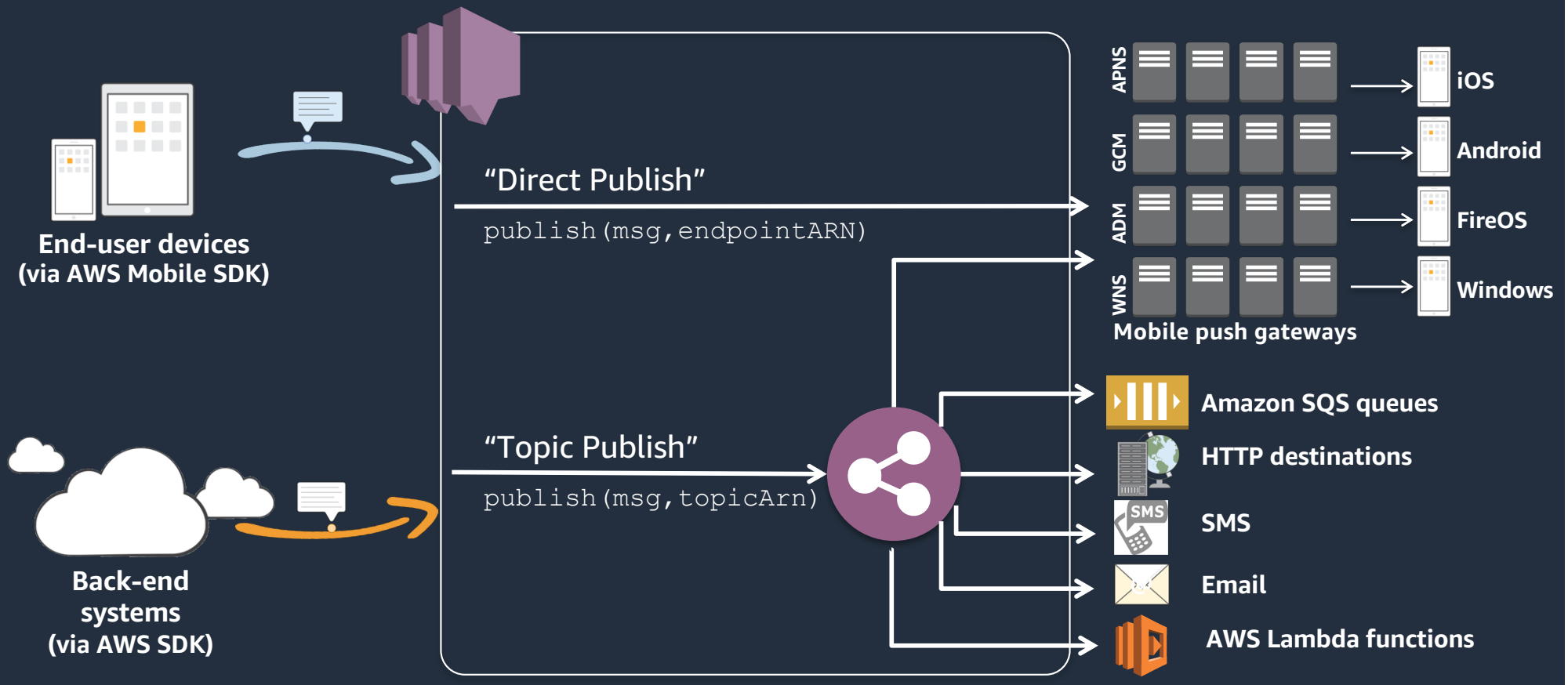
1) Create a Topic
2) Subscribe to a Topic
3) Publish to a Topic



See full API list:
http://docs.aws.amazon.com/sns/latest/api/Welcome.html

aws

One API - Multiple destination types



SQS vs SNS

SQS

- Queue
- Pull
- Decoupling two applications
- Messages are persisted in the queue for a fixed duration

SNS

- Topic (Pub/Sub)
- Push
- Fanout from one to many applications
- No persistence (but can retry)

aws

Poll#5

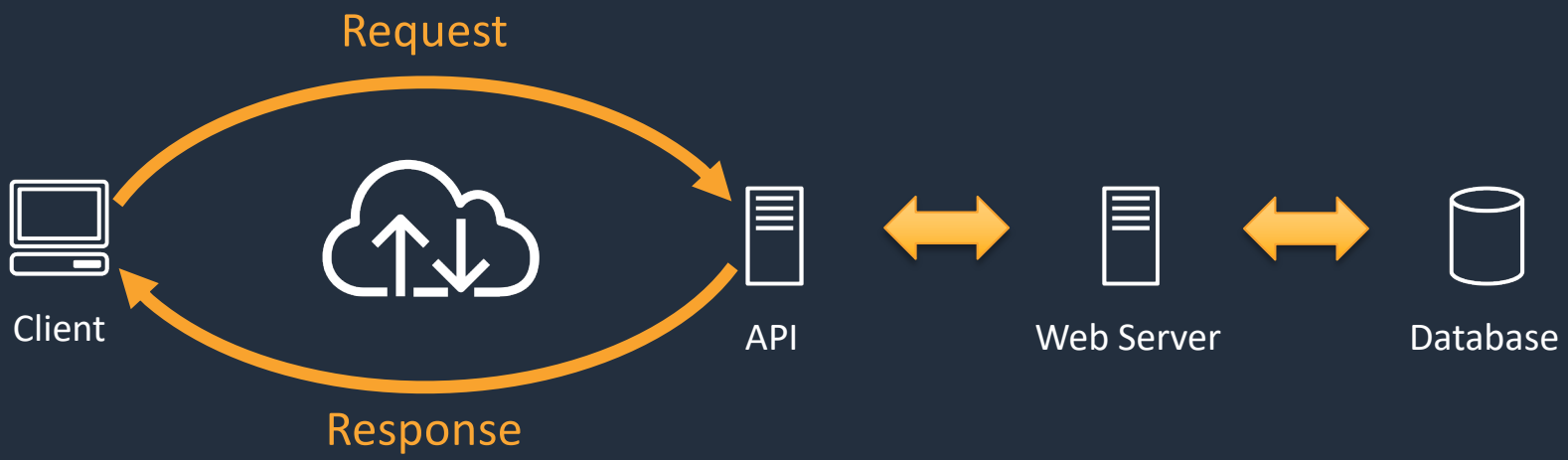
- Which delivery mechanism is NOT supported by Amazon Simple Notification Service (SNS)?
- A: Email
- B: HTTP(S)
- C: Short Message Service (SMS)
- D: Really Simple Syndication (RSS)

aws

Amazon API Gateway

Application Programming Interface (API)

> In building applications, an API simplifies programming by abstracting the underlying implementation and only exposing objects or actions the developer needs.



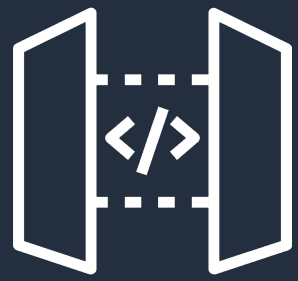
Web-based companies and services offer APIs for developers to use, such as:

- Social Networks – Facebook, Twitter, etc
- Payment Processing – Amazon Pay, PayPal, etc

https://en.wikipedia.org/wiki/Application_programming_interface

aws

Amazon API Gateway



API Gateway is a fully managed service that makes it easy for developers to create, publish, maintain, monitor, and secure APIs at any scale. It frees you from the operational burden of implementation, offers reliable network protection, and centralizes authorization decisions within policies so bugs and code concerns are minimized.

It also enables you to:

- Host multiple versions and stages of your APIs
- Create and distribute API Keys to developers
- Throttle and monitor requests to protect your backend
- Leverage signature version 4 to authorize access to APIs
- Perform Request / Response data transformation and API mocking
- Reduce latency and DDoS protection through CloudFront
- Store API responses through managed caches
- Generate SDKs for Java, JavaScript, Java for Android, Objective-C or Swift for iOS, and Ruby

aws

Supported protocols

RESTful: HTTP APIs & REST APIs

WebSocket APIs

- Request / Response
- HTTP Methods like GET, POST, etc
- Short-lived communication
- Stateless

- Serverless WebSocket
- 2 way communication channel
- Long-lived communication
- Stateful

aws

Types of APIs

RESTful APIs

HTTP APIs are the cheapest, fastest, best choice for building APIs that only require API proxy functionality. For APIs that require API proxy functionality and management features in a single solution, API Gateway also offers REST APIs.

WebSocket APIs

WebSocket APIs allow you to build real-time two-way communication applications, such as chat apps and streaming dashboards. API Gateway maintains a persistent connection to handle message transfer between a backend service and its clients.

Regional *(Available with all types)*

- Recommended API type for general use cases
- Designed for building APIs for clients in the same region

Edge-Optimized *(Available with REST APIs)*

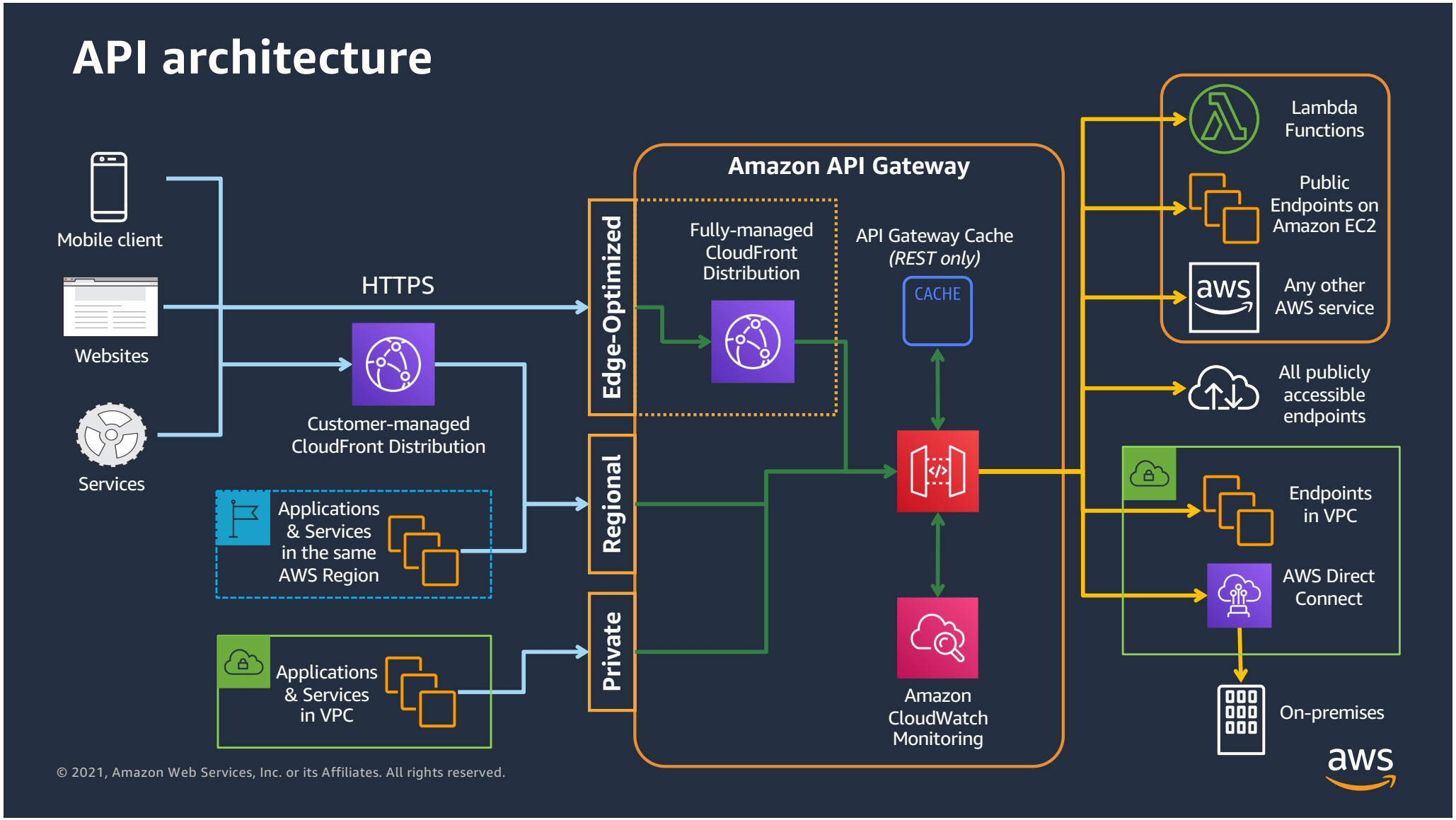
- Uses CloudFront to reduce TLS connection overhead (reduces roundtrip time)
- Designed for a globally distributed clients

Private *(Available with REST APIs)*

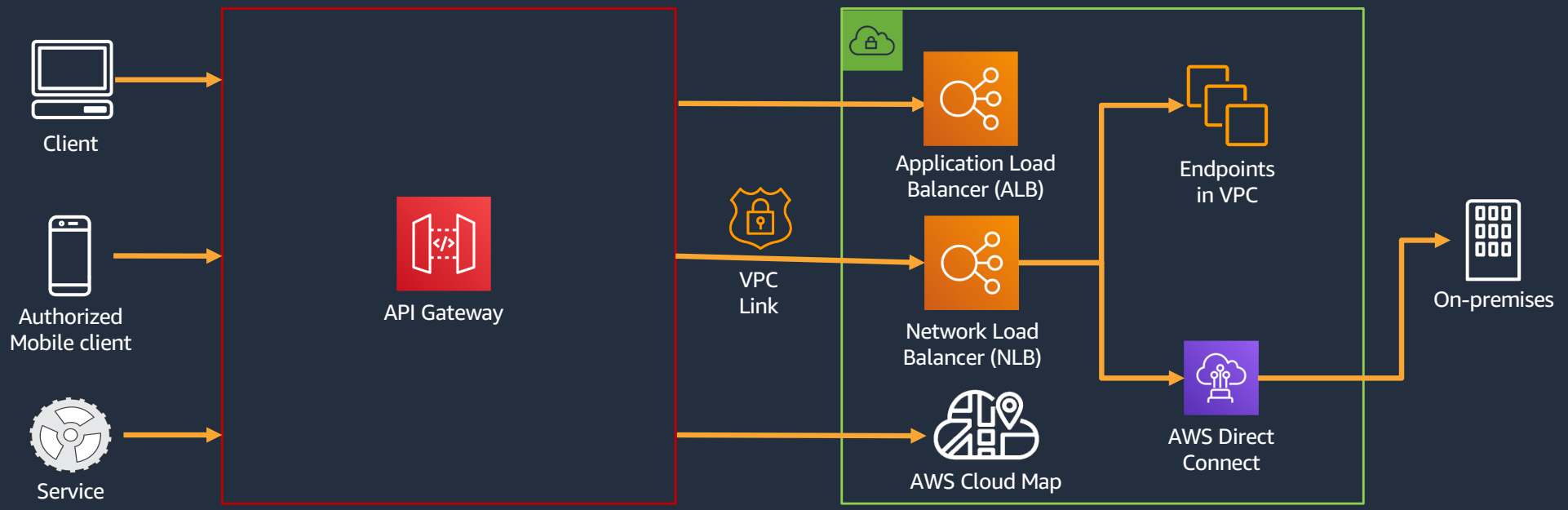
- Only accessible from within VPC (and networks connected to VPC)
- Designed for building APIs used internally or by private microservices

aws

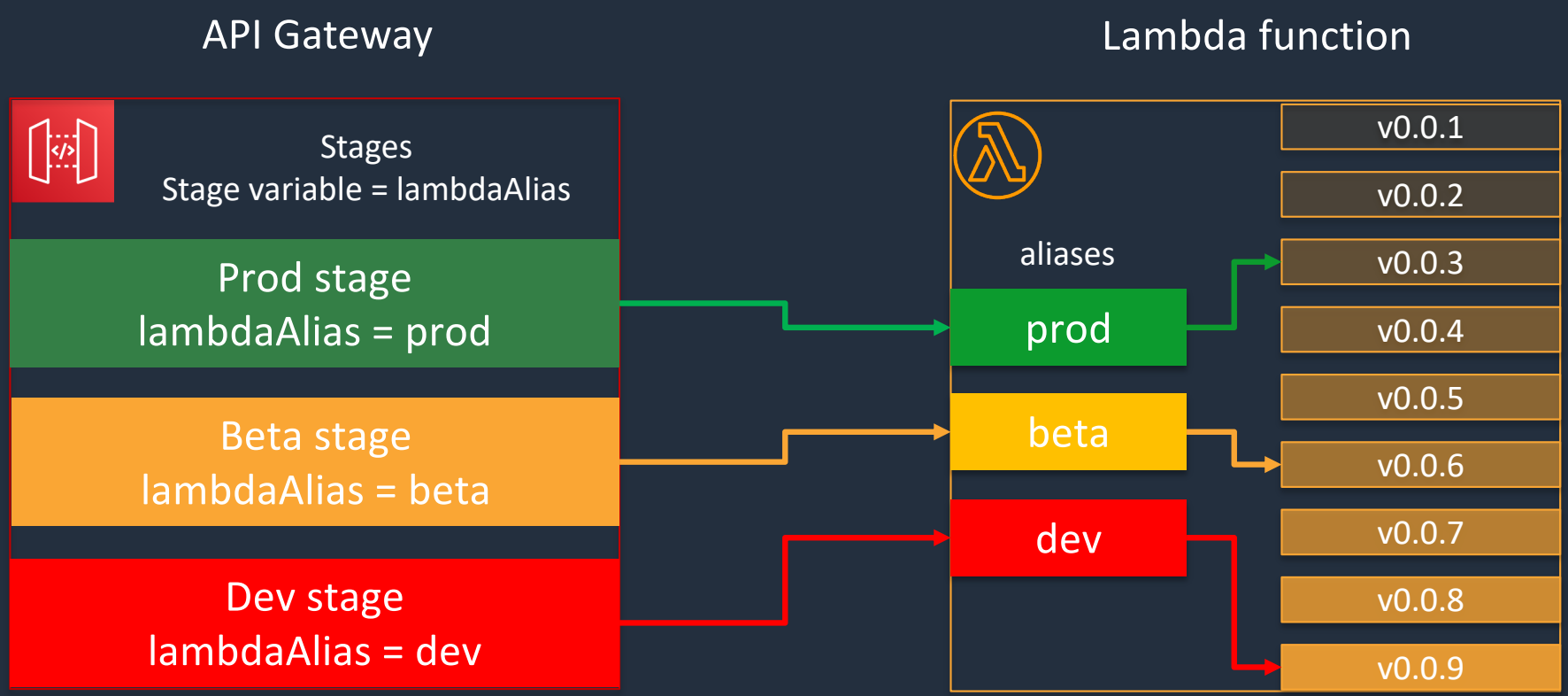
API architecture



VPC Links (Private Integrations)



Staging



Custom domains

~~https://12345.execute-api.us-east-1.amazonaws.com/prod~~

https://*mydomain.com/api-one*

- Supports HTTP, REST, and WebSocket APIs
- SSL Certs managed through ACM
- Supports multiple domains through base path mapping

aws

Poll#6

- Which of the following Component/Feature of API Gateway enabled Private Integration with AWS Backend Resources?
- A: Stage
- B: Custom Domain
- C: Method
- D: Resource
- E: VPC Links
- F: APIKEY

aws

Amazon Simple Email Service

Amazon SES is a flexible, affordable, and highly-scalable email sending and receiving service for businesses and developers

High performance

High deliverability

Scalable

Reliable

Configurable

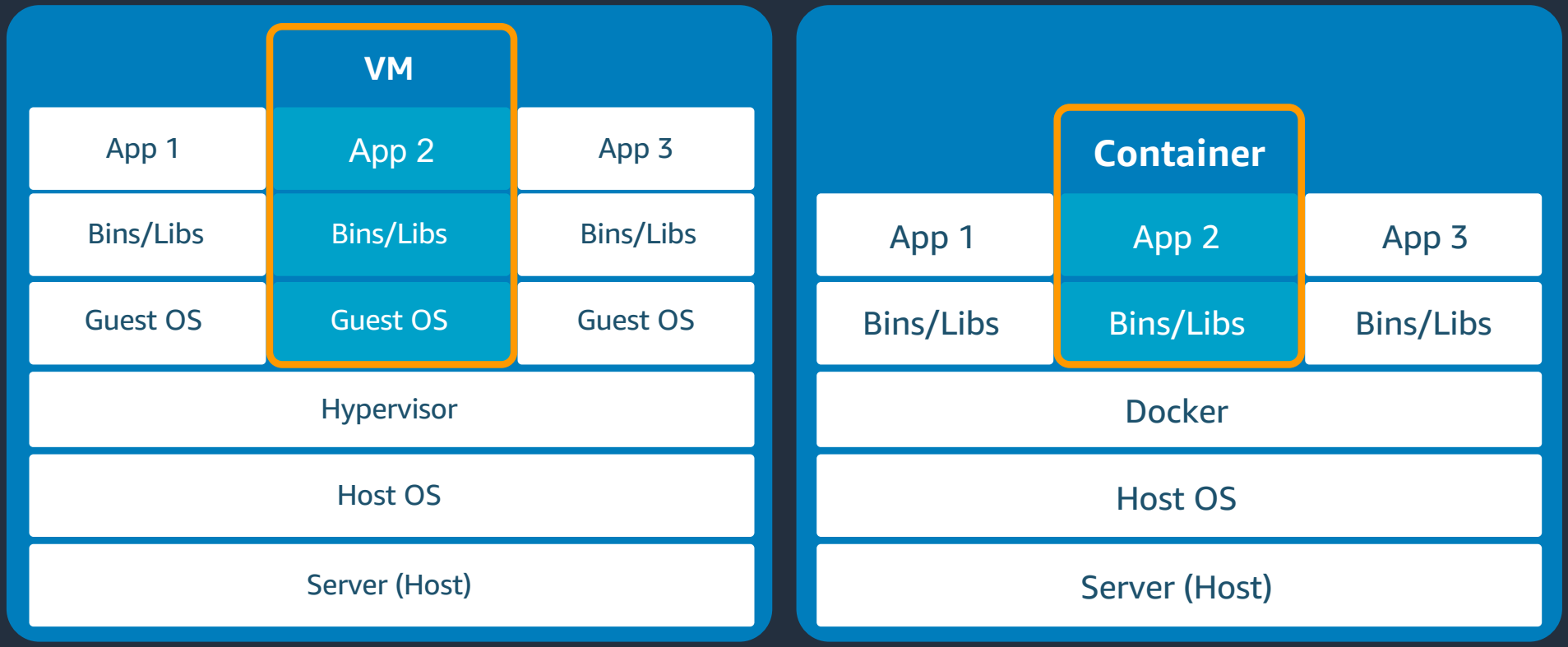
aws

Amazon Elastic Container Service (ECS)

aws

What are containers?

What are containers?



aws

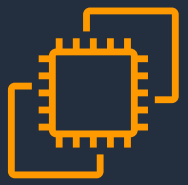
AWS container services landscape

Management
Deployment, Scheduling, Scaling & Management of containerized applications

Amazon Elastic Container Service

Amazon Elastic Kubernetes service

Hosting
Where the containers run



Amazon EC2

AWS Fargate

Image Registry
Container Image Repository

Amazon Elastic Container Registry

aws

Poll#7

- Which of the following AWS Container as a Service Platform supports Serverless Architecture/Deployment
- A: Amazon Elastic Container Service
- B: AWS Fargate
- C: Amazon Elastic Kubernetes Service

aws

Well-Architected Framework

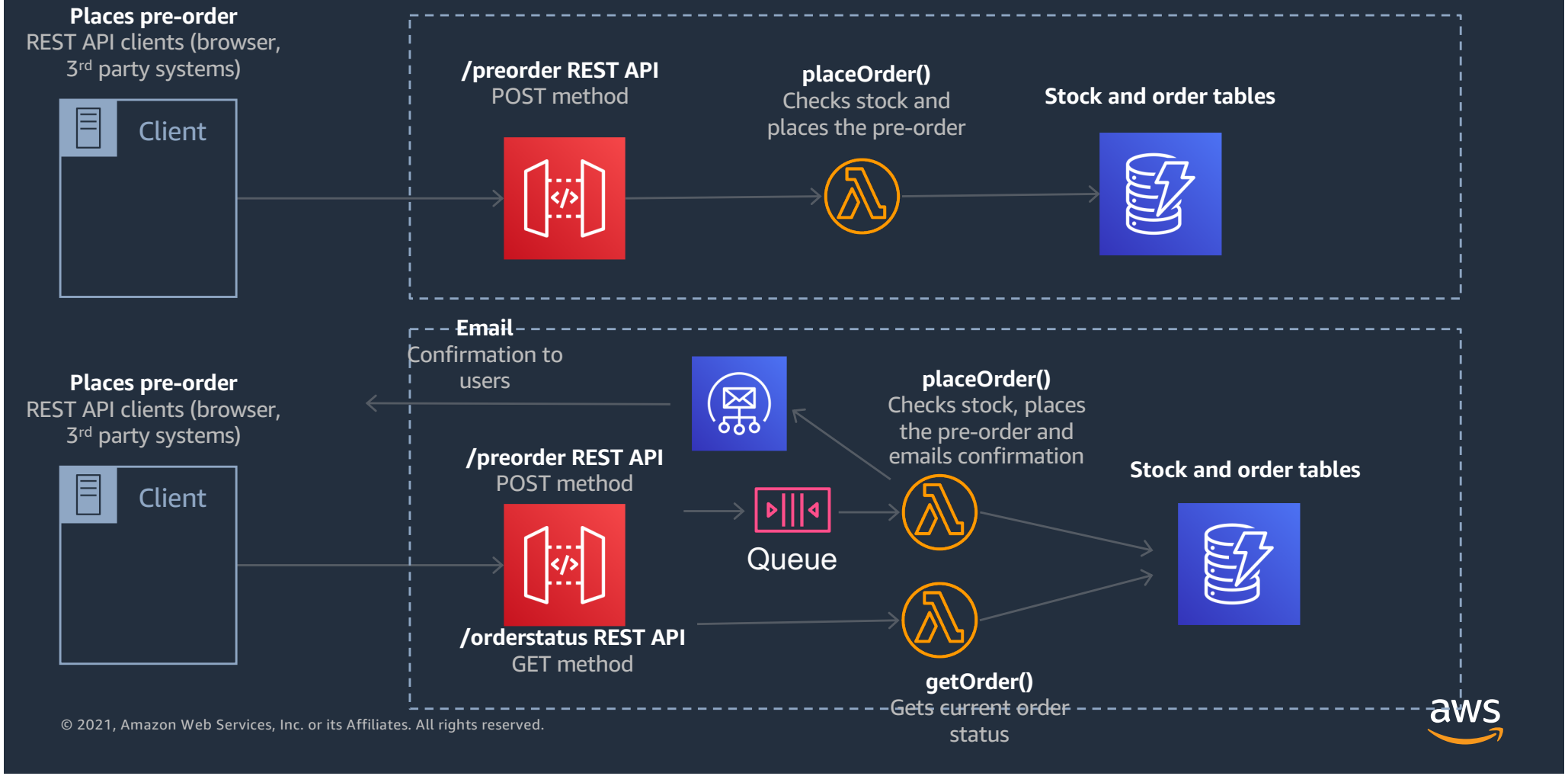
Pillars

General principles

- Stop guessing your capacity needs
- Test systems at production scale
- Automate to make architectural experimentation easier
- Allow for evolutionary architectures
- Drive architectures using data
- Improve through game days

An example serverless application

Example serverless implementation



Is it well-architected?

aws

Poll#8

- Which is NOT a pillar of the AWS Well-Architected Framework?
- A: Operational Excellence
- B: User Experience
- C: Security
- D: Cost Optimization

- B

aws

Poll#9

- Which service is a push-based, "pub-sub" messaging service, used to "fan out" applications?

  A: Amazon Simple Queue Service (SQS)
- B: Amazon API Gateway
- C: Amazon Simple Notification Service (SNS)
- D: AWS Step Functions

aws

Poll#10

- Your customer is building an internal application that serves as a repository for images uploaded by a couple of users. Whenever a user uploads an image, it would be sent to Kinesis for processing before it is stored in an S3 bucket. Afterwards, if the upload was successful, the application will return a prompt telling the user that the upload is successful. The entire processing typically takes about 5 minutes to finish.

  Which of the following options will allow you to asynchronously process the request to the application from upload request to Kinesis, S3, and return reply, in the most cost-effective manner? (SELECT ONE)

  A) Use a combination of SQS to queue the requests and then asynchronously process them using On-Demand EC2 instances

  B) Use a combination of SNS to buffer the requests and then asynchronously process them using On-Demand EC2 instance

  C) Create a Lambda function that will asynchronously process the request

  D) Use a combination of Lambda and Step Functions to orchestrate service components and asynchronously process the requests

Q&A

Further reading

- AWS Lambda FAQs
- AWS Step Functions FAQs
- Amazon SQS FAQs
- Amazon SNS FAQs
- Amazon API Gateway FAQs
- Amazon SES FAQs
- Amazon ECS FAQs
- AWS Well-Architected Framework Whitepaper

aws