Getting Started with Serverless

AWS
User Groups

Speaker

Sanchit Jain

Lead Architect - AWS at Quantiphi

AWS APN Ambassador

FOLLOW ME

AWS
User Groups

Agenda

AWS Serverless Introduction	
S.NO.	AGENDA
1	Introduction
2	Demo
3	Lambda Code Walkthrough
4	SQS & SNS
5	Step Function
6	Demo

What is serverless?

What is Serverless?

a cloud-native platform

for

short-running, stateless computation

and

event-driven applications

which

scales up and down instantly and automatically

and

charges for actual usage at a millisecond granularity

Greater Agility

Less Overhead

Better Focus

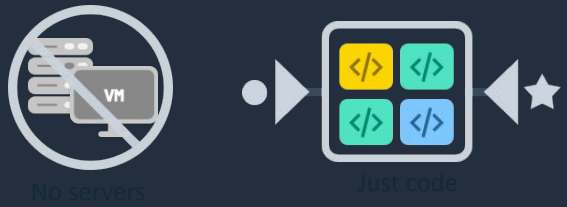
Increased Scale

More Flexibility

Faster Time To Market

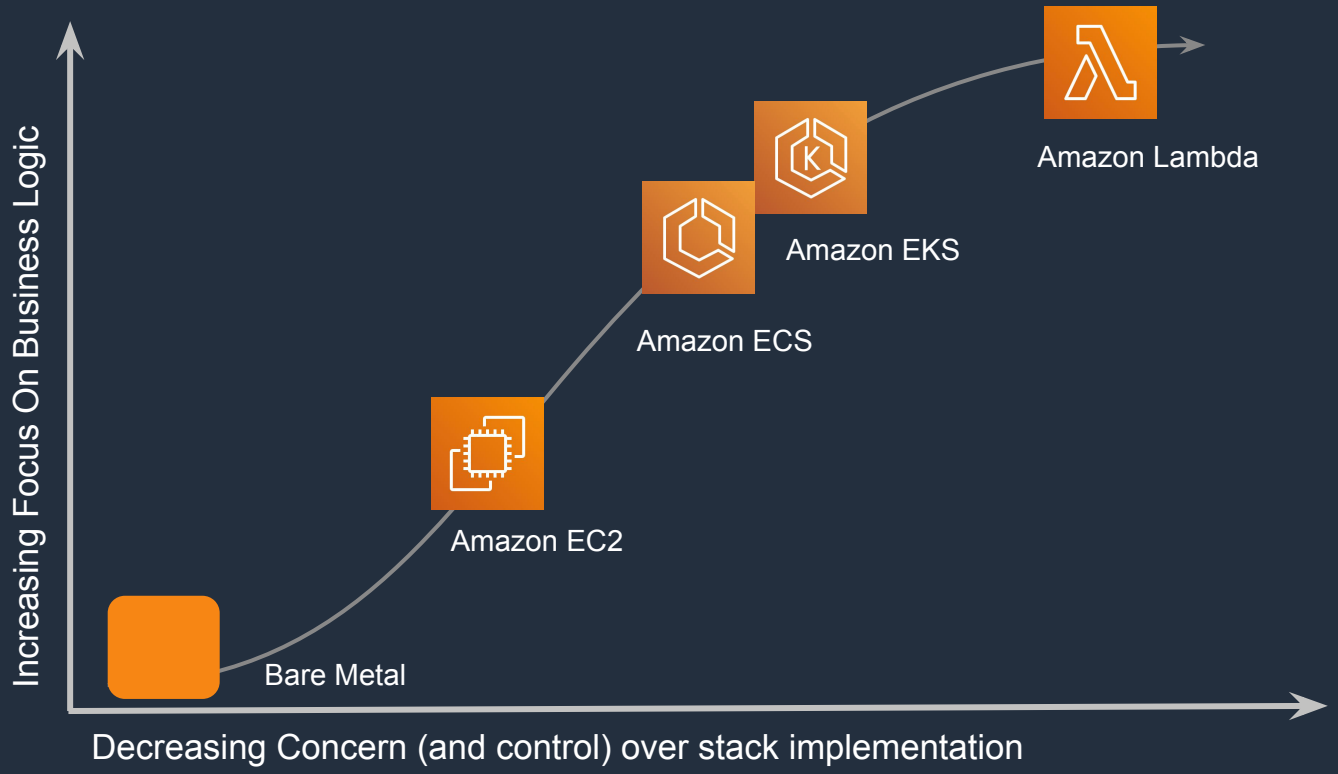
Why is Serverless attractive?

- Server-less means no servers? Or worry-less about servers?
- Runs code **only** on-demand on a per-request basis
- Making app development & ops dramatically faster, cheaper, easier
- Drives infrastructure cost savings



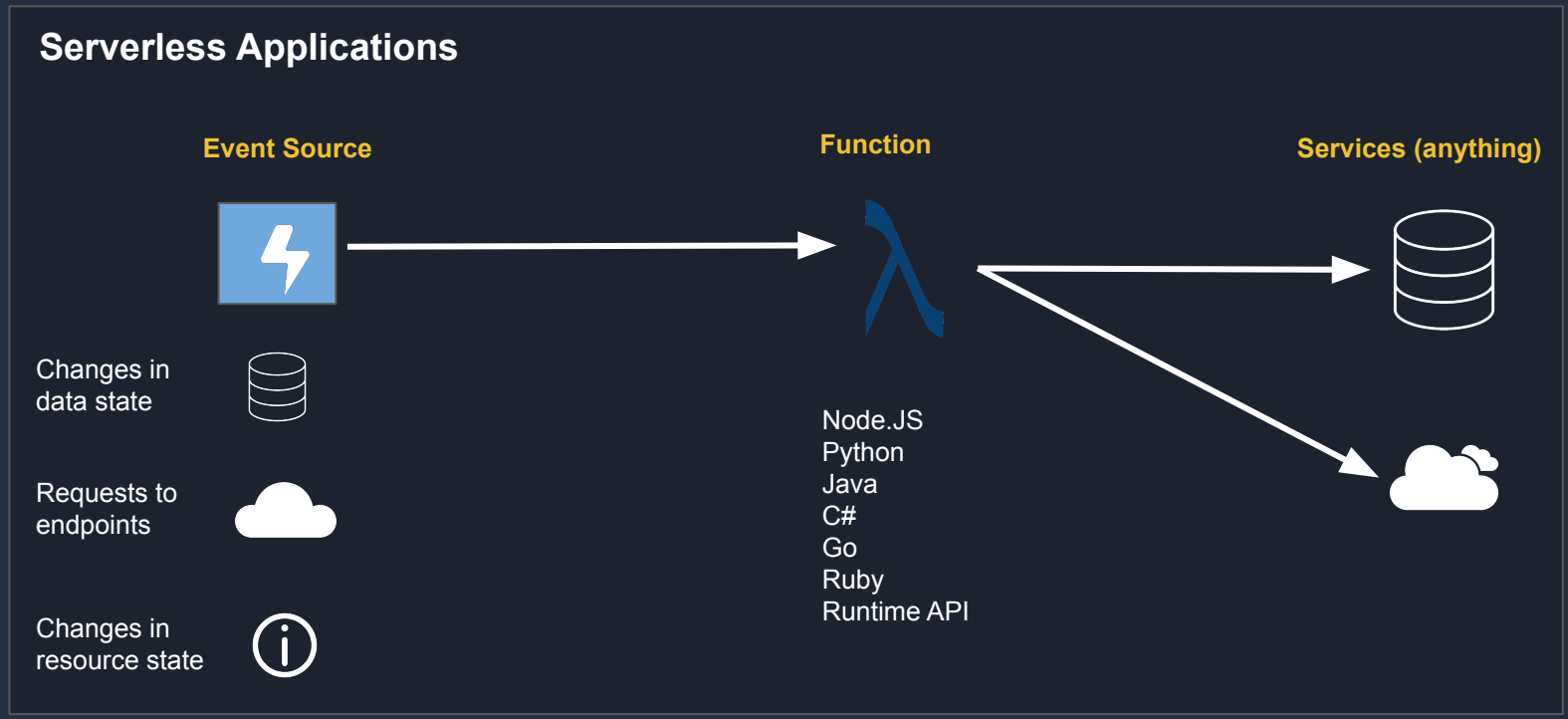
	On-prem	VMs	Containers	Serverless
Time to provision	Weeks-months	Minutes	Seconds-Minutes	Milliseconds
Utilization	Low	High	Higher	Highest
Charging granularity	CapEx	Hours	Minutes	Blocks of milliseconds

Where Serverless Stands?



What triggers code execution?

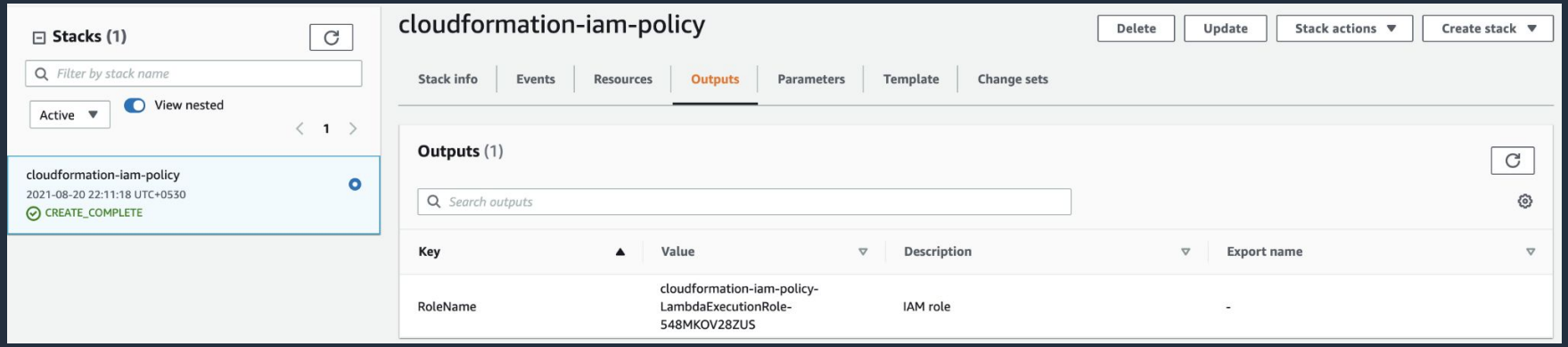
- Runs code in response to events
- Event-programming model



Demo

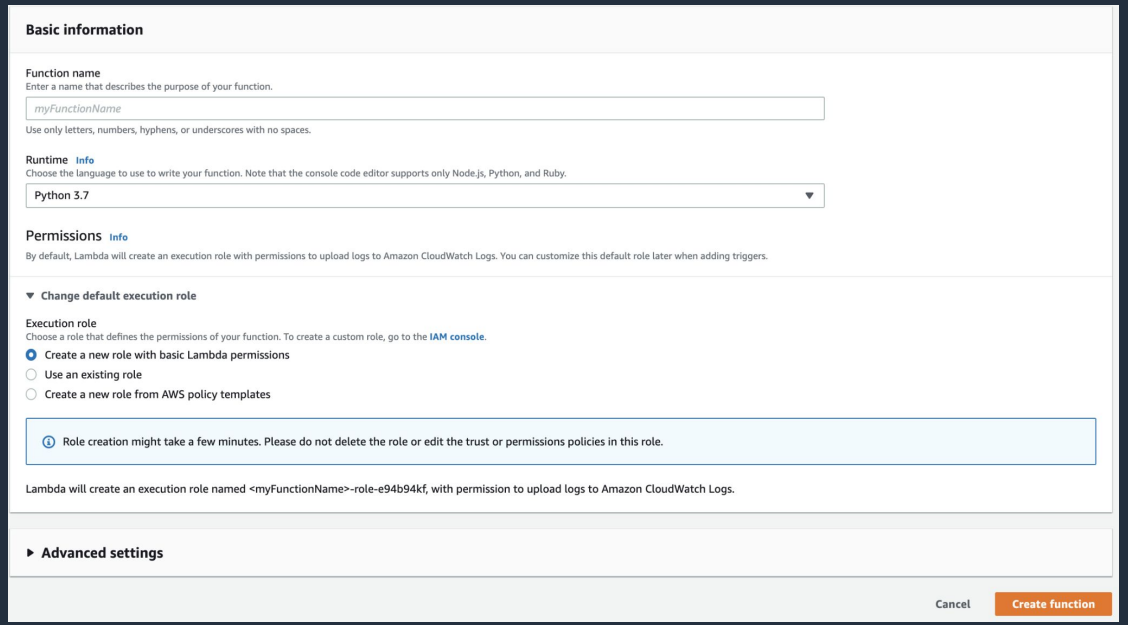
Prerequisites

- Download the cloudformation template from this link and Deploy it
- Once the Cloudformation stack is deployed successfully please capture the values for RoleName



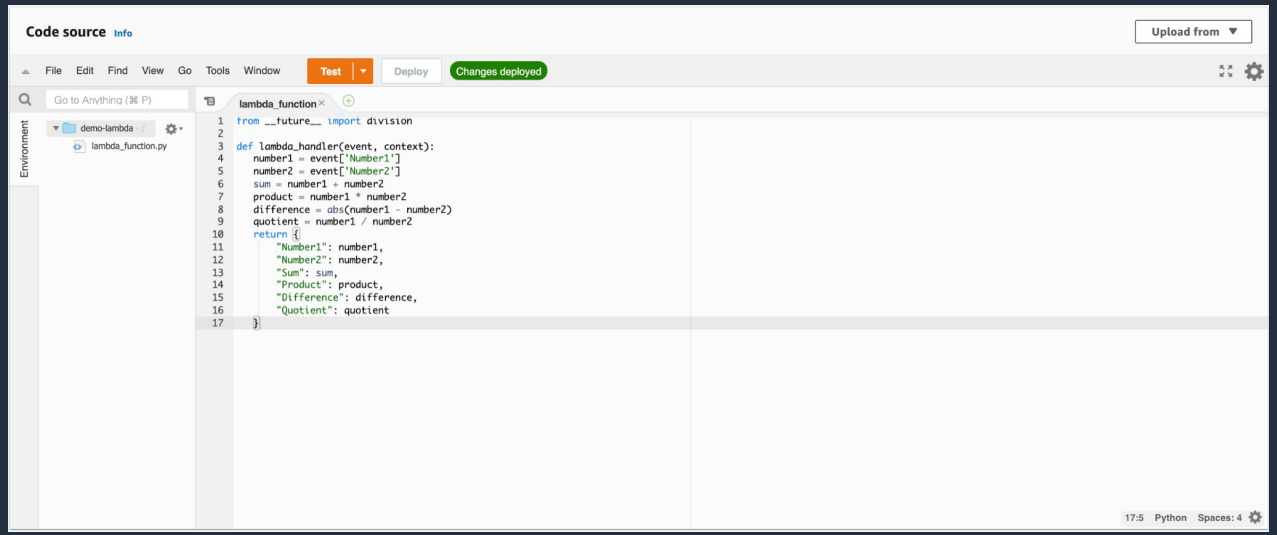
Create function

- Navigate to the AWS Lambda service
- On the Lambda console, Click Create function
- In the Create function, enter demo-lambda as the function name, runtime as Python 3.7
- In the execution role section, select the role cloudformation-iam-policy--xxxxx from the drop-down list
- Click Create function



Configure function

- Download the code from this link and paste it into code source
- Click Deploy, then click Test
- Copy the message { "Number1": 10, "Number2": 20 }, and paste it
- Click Test again, and capture the output



Lambda Code Walkthrough

Anatomy Of A Lambda Function

Handler() Function

Function to be executed upon invocation

Event Object

Data sent during Lambda Function invocation

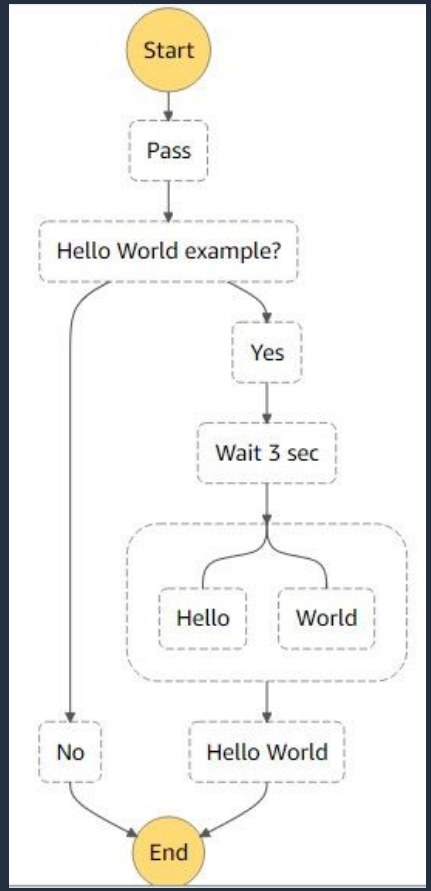
Context Object

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

```
def lambda_handler(event, context):
    return {
        "Hello World!"
}
```

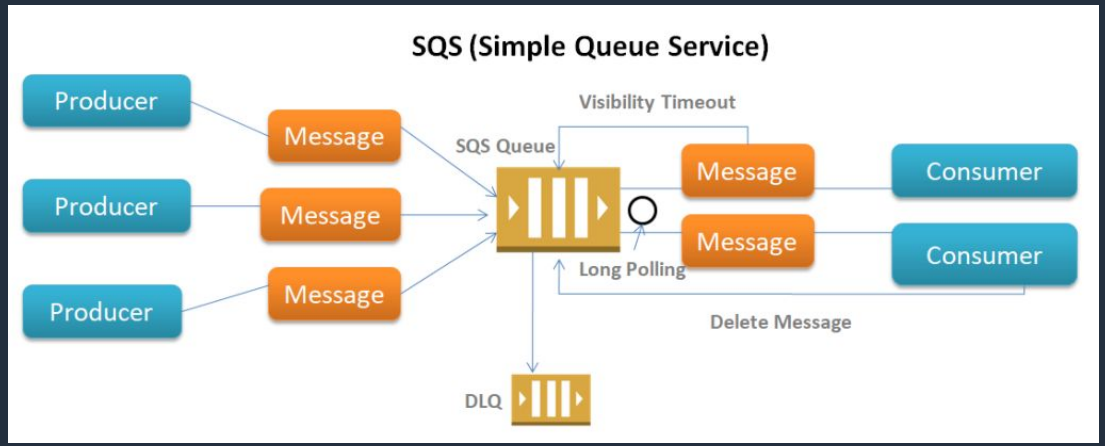
What is AWS Step Functions?

- AWS Step Function is a serverless orchestration service that allows integrating multiple AWS services to collate & design an enterprise-critical application or workflow with advance conditional branching and error handling
- ASL consist of three things
  - **State Machine Structure** - State machines are declared using JSON text and represents a structure consists of Comment, TimeoutSeconds, Version, StartAt, States
  - **Intrinsic functions** - Intrinsics are constructs like in programming languages, and can be leveraged to manipulate the data going to and from Task Resources
  - **Common State Fields** - Common State Fields consists of Comment, InputPath, OutputPath, Type, Next, End



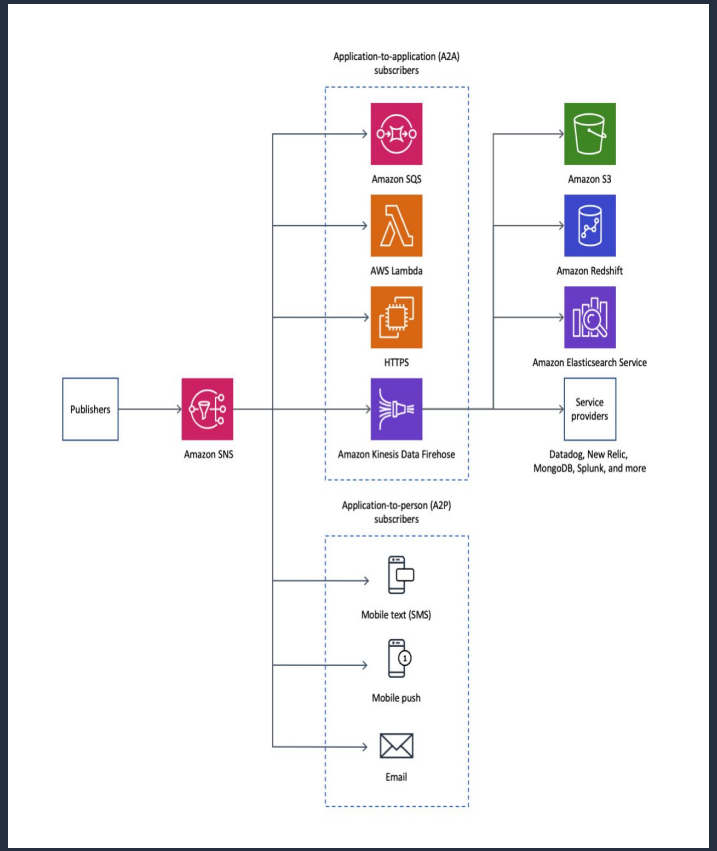
What is AWS SQS?

- Oldest AWS offering (over 10 years old)
- Fully managed service, use to decouple applications
- Scales from 1 message per second to 10,000s per second, and Low latency (<10 ms on publish and receive)
- Default retention of messages: 4 days, maximum of 14 days
- No limit to how many messages can be in the queue, and messages are deleted after they’re read by consumers
- Consumers share the work to read messages & scale horizontally

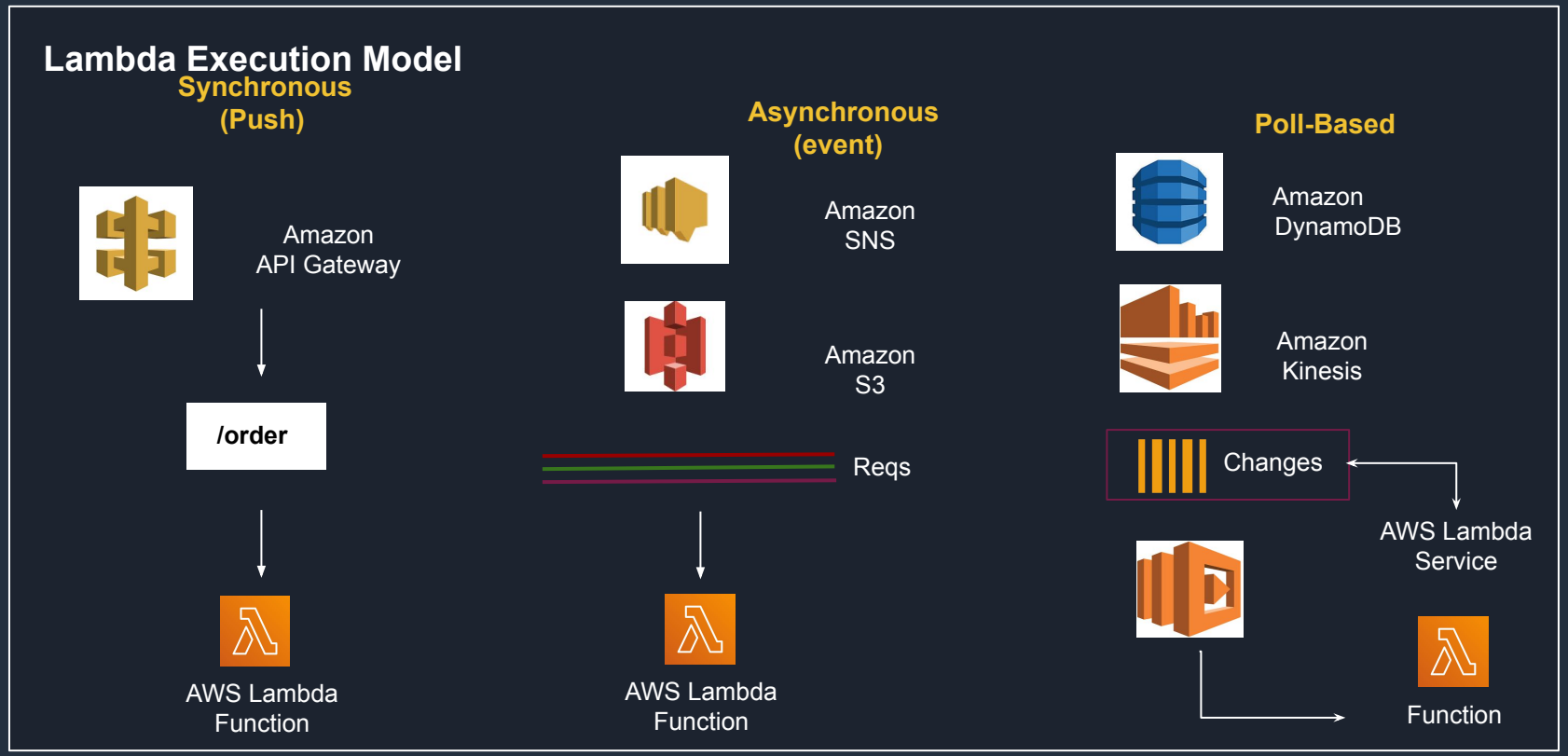


What is AWS SNS?

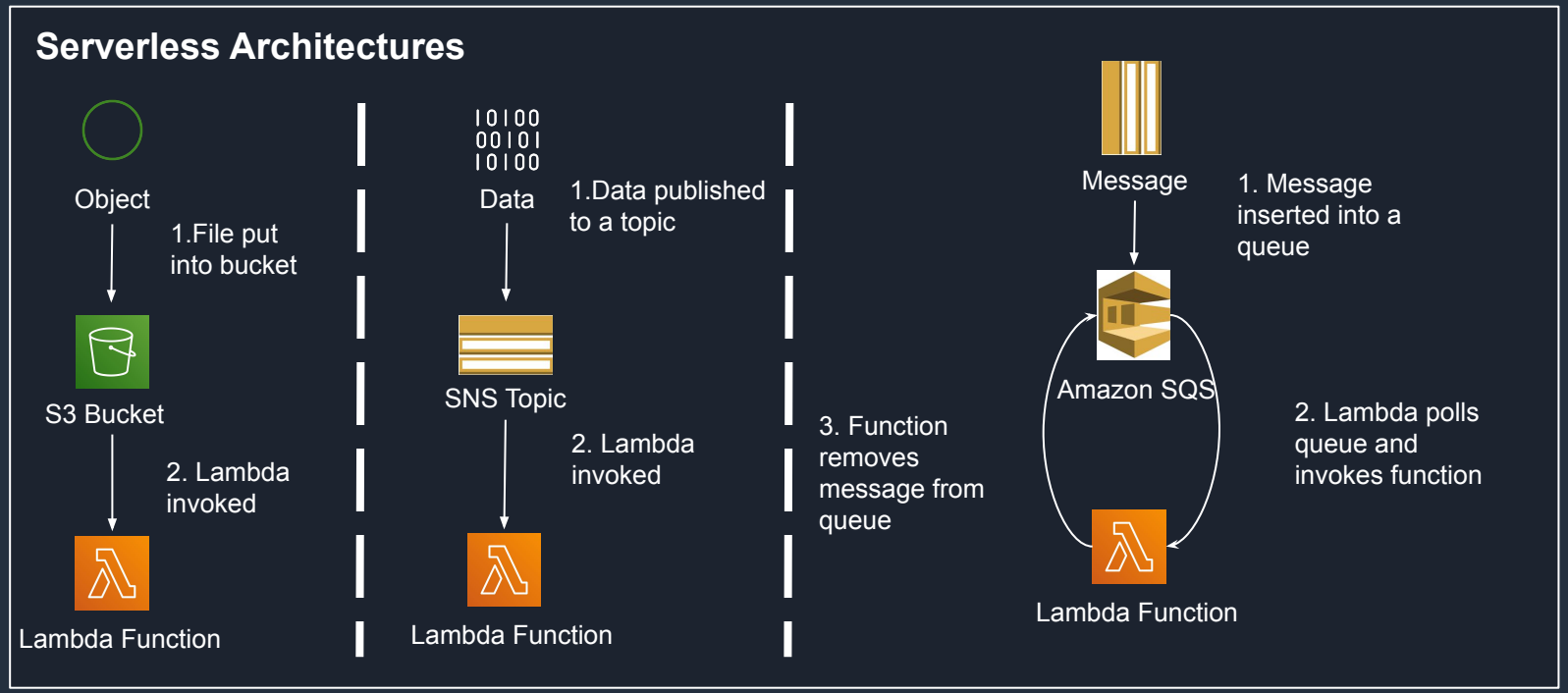
- Publishers sends message to a SNS topic and multiple subscribers listen to the SNS topic notifications
- Each subscriber to the topic will get all the messages
- Up to 10,000,000 subscriptions per topic, 100,000 topics limit
- SNS Subscribers can be:
  - HTTP / HTTPS (with delivery retries – how many times)
  - Emails,SMS messages,Mobile Notifications
  - SQS queues (fan-out pattern), Lambda Functions (write-your-own integration)



Lambda Invocation/Execution Model



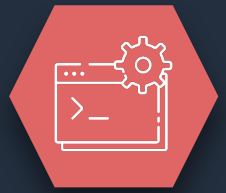
Different Serverless Architecture



Different Use cases

Web Applications

- Static Websites
- Complex Web Apps
- Packages for Flask and Express



Backends

- Apps and Services
- Mobile
- IoT

Data Processing

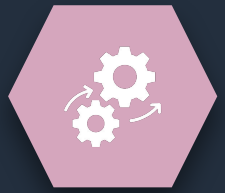
- Real time
- Map Reduce
- Batch

Chatbots

- Powering Chatbot Logic

Amazon Alexa

- Powering Voice Enabled Apps
- Alexa Skills Kit



IT Automation

- Policy Engines
- Extending AWS Services
- Infrastructure Management

What is Serverless good for?

Serverless is **good** for
short-running stateless
event-driven

- Microservices
- Mobile Backends
- Bots, ML Inferencing
- IoT
- Modest Stream Processing
- Service integration

Serverless is **not good** for
Long-running stateful
number crunching

- Databases
- Deep Learning Training
- Heavy-Duty Stream Analytics
- Numerical Simulation
- Video Streaming

Demo

Demo - Lambda invocation via SQS

- Create an Amazon SQS queue
  - Navigate to the SQS console
  - Choose Create a SQS, and then provide queue name
  - Capture the ARN (Amazon Resource Name) of the SQS
- Configure the event source
  - Under SQS console switch to details section, and select Lambda triggers
  - Click configure Lambda triggers, and select or provide Lambda ARN
- Now we will test the integration
  - Copy the message { "Number1": 10, "Number2": 20 }, and send it as a SQS message
  - Verify the lambda Cloudwatch logs and output for the same

Thank you!

AWS
User Groups