

# What is Serverless?

Any service in AWS is serverless if the following apply:

## No servers

- There are no servers exposed that need to be directly administered.

## Elastic

- Service scales automatically and is highly available

## Pay as you go

You only pay for what you use

# What is Serverless?

- Serverless is a new paradigm in which the developers don't have to manage servers anymore
- Only deploy code i.e **Lambda Functions**
- Serverless was pioneered by AWS Lambda but now also includes anything that's **managed**: “databases, messaging, storage, etc.”
- ***Serverless does not mean there are no servers***, basically it means you just don't **manage / provision / see** them.
- Just like RDS, SNS.

# Serverless in AWS

- **AWS Lambda**
- **CloudWatch**
- **S3**
- **SNS**
- **Athena**
- **DynamoDB - NoSQL Key:Value DB**
- **API Gateway - Rest API**
- **SQS - Queue**
- **Kinesis - Streaming Data Processing**
- **Aurora Serverless ( RDS )**
- **Step Functions (serverless workflow)**

# Why Lambda?

- **EC2:**

Virtual Servers in the Cloud

Limited by RAM and CPU

Continuously running

Scaling means intervention to add / remove servers

- **Lambda:**

Virtual **functions** – no servers to manage!

Limited by time - **short executions ( 15 mins )**

Run **on-demand** ( **Not continuously running in background** )

**Supports Event Based and Scheduled based execution.**

**Scaling is automated!**



# AWS Lambda Language support

- **Python ( boto3 module )**
- Node.js (JavaScript)
- Java (Java 8 compatible , aws-java-sdk )
- C# (.NET Core)
- Go lang
- C# / Powershell

# Anatomy of Lambda Function in AWS



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

## Event Object

Data send during Lambda Function Invocation

## Context Object

Methods available to interact with runtime information ( log group, request id etc)

## lambda\_handler() Function

Function to be executed upon invocation

# Lambda Invoke Scenarios

## AWS Lambda - Permissions Model



Who can  
invoke this Lambda?

Whom can  
this Lambda invoke?

Which service is allowed to  
invoke a Lambda function?

Once invoked, which  
service is a Lambda  
function in turn allowed to  
invoke?

# Benefits of AWS Lambda

- **Easy Pricing :**

Charged based on the number of **requests** for your functions and the **duration** i.e the time it takes for your code to execute.

The AWS Lambda free usage tier includes **1Million free requests per month** and **400,000 GB-seconds of compute time per month**, [Lambda Pricing](#)

- Integrated with the whole AWS Stack ( SDKs for all services )
- Integrated with many programming languages.
- Easy monitoring through AWS CloudWatch.
- Easy to get more resources per functions (upto 10GB RAM)
- Increasing RAM will also improve CPU and network!

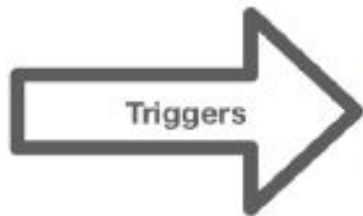


# Lambda Usage

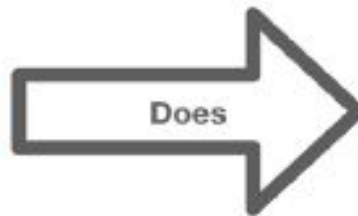
Event Source



AWS Services



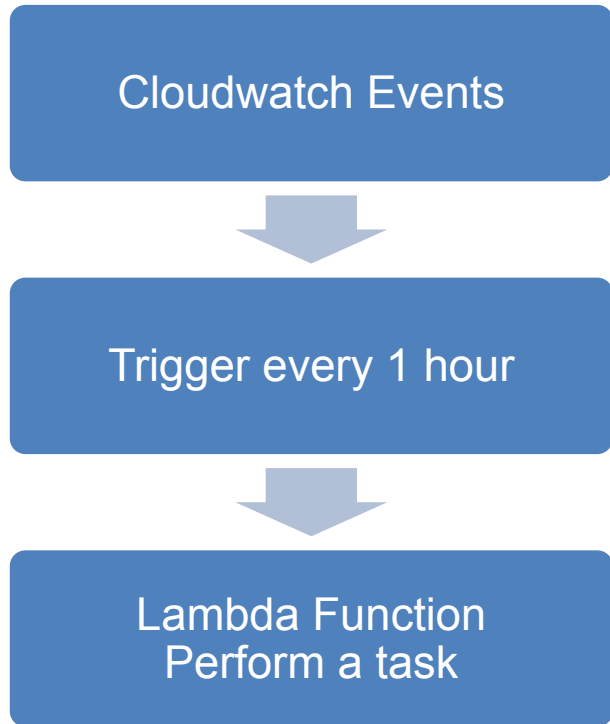
Lambda Function



Anything



# Example : Serverless Cron Job



# AWS Lambda Configuration

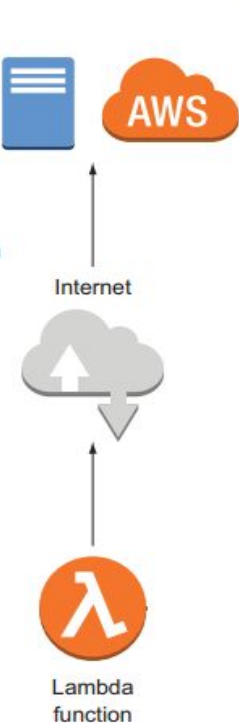
- Allocated memory (**128M** ( default) to **10G**)
- Timeout: default **3 seconds**, max of 300s (Note: **new limit 15 minutes**)
  - Timeout configuration should be greater than your execution duration.
- **IAM execution role** must be attached to the Lambda function
- Lambda Execution Logs will be stored in **CloudWatch Log Groups**.
- **Environment variables**
- Ability to deploy within a VPC + assign security groups

# AWS Lambda – Internet Access

Accessing any kind of resources is possible as long as they are reachable from the internet.

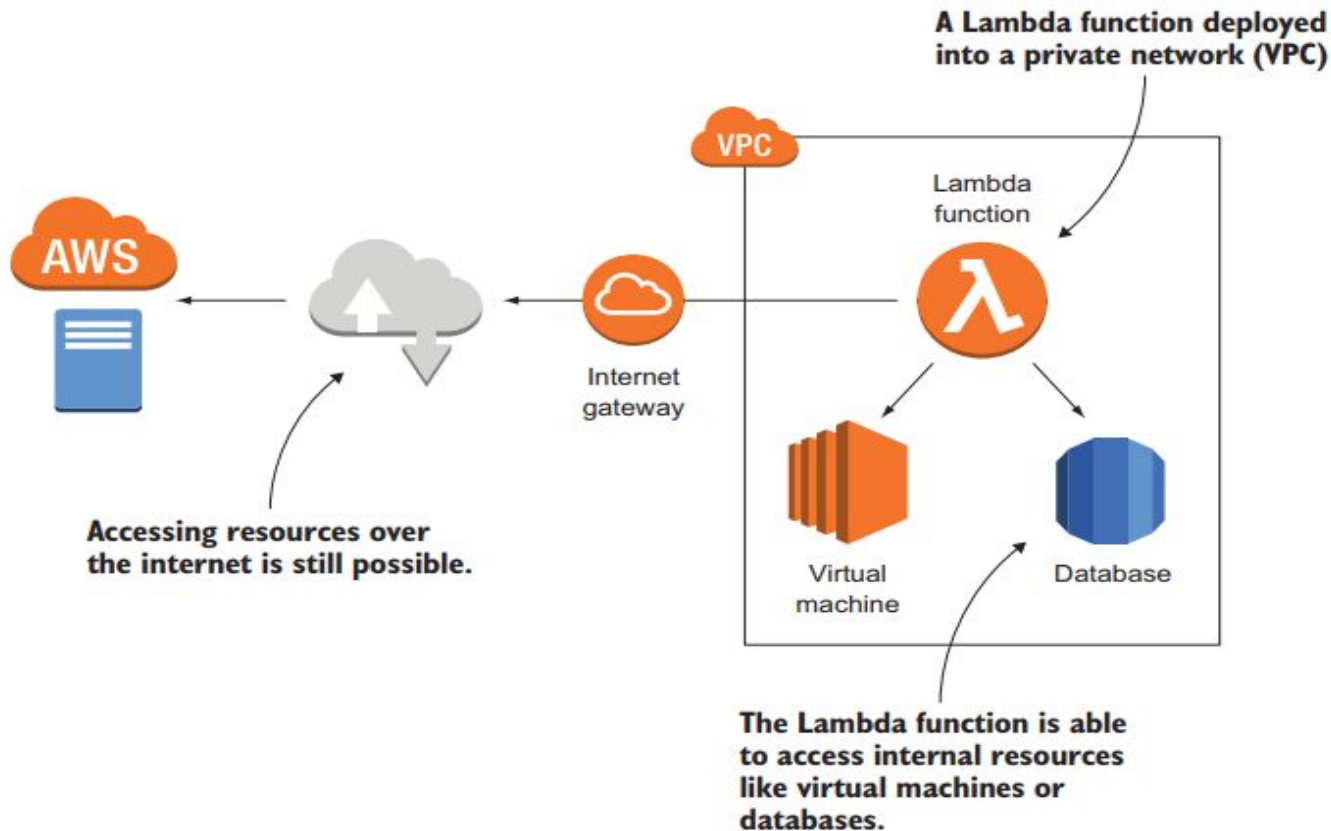
Some AWS services are accessible via internet (e.g., CloudWatch, DynamoDB).

The internet is accessible for each Lambda function by default.



By default a Lambda function is connected to the internet and running outside your VPCs.

# AWS Lambda – Within VPC



Deploying a Lambda function into your VPC allows you to access internal resources (such as database, virtual machines, S3 within AWS Private network etc).  
Lambda Function will use an ENI in your subnet



# AWS Lambda Logging, Monitoring

- **CloudWatch:**

- Lambda integrates with CloudWatch Logs and pushes all logs from your code to a CloudWatch Logs group associated with a Lambda function, which is named ***/aws/lambda/<function name>***
- AWS Lambda metrics are displayed in AWS CloudWatch Metrics

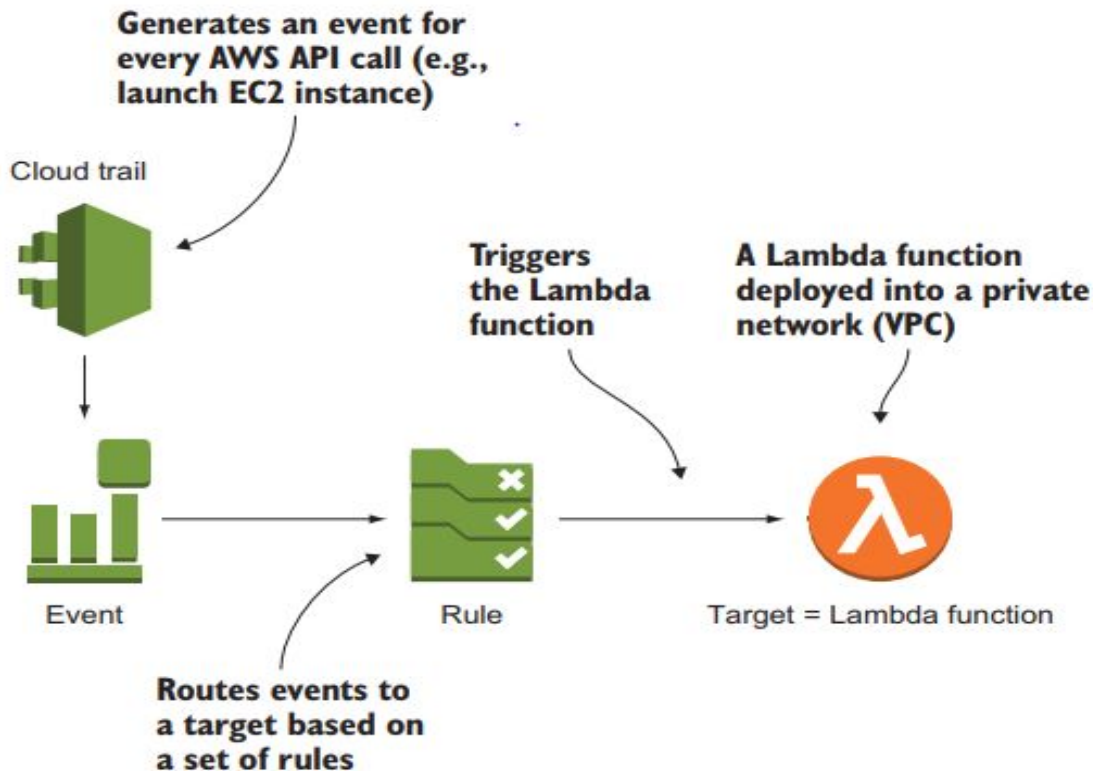
**Make sure your AWS Lambda function has an execution role with an IAM policy that authorizes writes to CloudWatch.**



# AWS Lambda Best Practices

- **Perform heavy-duty work outside of your function handler**
  - Connect to databases outside of your function handler
  - Initialize the AWS SDK outside of your function handler
  - Pull in dependencies/modules or datasets outside of your function handler
- **Do not hard code any AWS Resource Names/ARN that will change when you create lambda in different Environments ( dev/qa/prod ), different region, different AWS Account.**
- **Use environment variables for - Anything that is not sensitive**
  - Database Connection Strings, S3 bucket names, SNS Topic ARNs, etc... **do not put these values directly in your code.**
  - Passwords, sensitive values can be stored in **SSM Parameter Store ( Secure Strings ), Secrets Manager.**
- **Avoid using recursive code, never have a Lambda function call itself.**
- **Don't put your Lambda function in a VPC unless you have to.**

# Operational Tasks with Lambda



CloudTrail generates an event for every AWS API call, a rule routes the event to the Lambda function



# S3 Event Custom Email Notification

