Build and Orchestrate Serverless Applications on AWS with PowerShell

Trevor Sullivan
Solutions Architect
Amazon Web Services (AWS)
Agenda

• Serverless Foundations
• Serverless Architecture Patterns
  • Web Application
  • Automation
  • Stream Processing
  • Batch Processing
Spectrum of AWS offerings

- **“On EC2”**
  - Amazon EC2
  - EMR
  - Amazon ElastiCache
  - RDS
  - Redshift

- **Managed**
  - Amazon ES
  - S3
  - DynamoDB
  - API Gateway
  - AWS Step Functions
  - Cognito

- **Serverless**
  - Lambda
  - Kinesis
  - SQS
  - IoT

© 2018, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Serverless means...

- No servers to provision or manage
- Scales with usage
- Never pay for idle
- Availability and fault-tolerance built in
Serverless applications

EVENT SOURCE
- Changes in data state
- Requests to endpoints
- Changes in resource state

FUNCTION
- Node.js
- Python
- Java
- C#
- Go
- PowerShell

SERVICES (ANYTHING)
CUSTOMERS LOVE SERVERLESS

© 2018, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Using AWS Lambda

**Bring your own code**
- Node.js, Java, Python, C#, Go, PowerShell
- Bring your own libraries (even native ones)

**Authoring functions**
- WYSIWYG editor or upload packaged .zip
- Third-party plugins (Eclipse, Visual Studio)

**Simple resource model**
- Select power rating from 128 MB to 3 GB
- CPU and network allocated proportionately

© 2018, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Using AWS Lambda

**Stateless**
- Persist data using external storage
- No affinity or access to underlying infrastructure

**Programming model**
- Use processes, threads, /tmp, sockets normally
- AWS SDK built in (Python and Node.js)

**Flexible use**
- Synchronous or asynchronous
- Integrated with other AWS services
Event sources that trigger AWS Lambda

DATA STORES
- Amazon S3
- Amazon DynamoDB
- Amazon Kinesis
- Amazon Cognito

ENDPOINTS
- Amazon API Gateway
- AWS IoT
- AWS Step Functions
- Amazon Alexa

DEVELOPMENT AND MANAGEMENT TOOLS
- AWS CloudFormation
- AWS CloudTrail
- AWS CodeCommit
- Amazon CloudWatch

EVENT/MESSAGE SERVICES
- Amazon SES
- Amazon SNS
- Cron events

…and more!
AWS Lambda permissions model

Fine-grained security controls for both invocation and execution

Function policies:
• What can invoke the Lambda function

Execution role:
• What the Lambda function can access
Basic Function Code

```powershell
#require -module awspowershell.netcore

Write-Output -InputObject $LambdaContext, $LambdaInput
```
Create a unified API front end for multiple microservices

DDoS protection and throttling for your backend

Authenticate and authorize requests to a backend

Throttle, meter, and monetize API usage by third-party developers
AWS Step Functions

“Serverless” workflow management with zero administration:

- Coordinates distributed applications using visual workflows
- Automatically triggers, tracks, and logs each step
Security and Identity

AWS Identity and Access Management

- Fine-grained access control to AWS resources

Amazon Cognito

- User pools for secure, managed user directories
- Identity pools for federation and role-based access control
Logging and Monitoring

Amazon CloudWatch

AWS X-Ray
AWS Serverless Application Model (SAM)

- Extension of AWS CloudFormation
- Automate deployment via AWS CodePipeline
- Build, test, and debug locally via SAM CLI
- Search and deploy serverless apps on the AWS Serverless Application Repository
Build PCI- and HIPAA-compliant serverless applications!

Serverless platform services that can be used in both:

- AWS Lambda
- Amazon S3
- Amazon CloudFront
- Amazon DynamoDB
- Amazon Kinesis
- Amazon Cognito
- Amazon API Gateway
- Amazon SNS
Architecture Patterns
Common serverless use cases

**Web applications**
- Static websites
- Complex web apps
- Packages for Flask and Express

**Backends**
- Apps and services
- Mobile
- IoT

**Data processing**
- Real-time
- MapReduce
- Batch

**Chatbots**
- Powering chatbot logic

**Amazon Alexa**
- Powering voice-enabled apps
- Alexa Skills Kit

**IT automation**
- Policy engines
- Extending AWS services
- Infrastructure management
Web Application Characteristics

1. Static and Dynamic Content
2. Microservices
3. Scalable
4. Authenticate and Authorize
5. Globally available
Bustle Achieves 84% Cost Savings with AWS Lambda

With AWS Lambda, we eliminate the need to worry about operations

Tyler Love
CTO, Bustle

• Bustle had trouble scaling and maintaining high availability for its website without heavy management
• Moved to serverless architecture using AWS Lambda and Amazon API Gateway
• Experienced approximately 84% in cost savings
• Engineers are now focused on innovation

Bustle is a news, entertainment, lifestyle, and fashion website targeted towards women.
Multi-Region with API Gateway

Client

Amazon Route 53

api.mycorp.com

Custom Domain Name

Regional API Endpoint

API Gateway

Lambda

us-east-1

us-west-2

© 2018, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Automation characteristics

• Periodic jobs
• Event triggered workflows
• Enforce security policies
• Audit and notification
• Respond to alarms
• Extend AWS functionality
Enforce security policies

New Security Group ingress rule

RDP from 0.0.0.0/0

Amazon CloudWatch Events: Rule

AWS Lambda: Remediate and alert

Ingress rule deleted

AWS SNS: Email alert

CloudWatch Event Bus in another AWS Account
Autodesk - Tailor

Serverless AWS Account Provisioning and Management Service:
• Automates AWS Account creation,
• Configures AWS IAM, AWS CloudTrail, AWS Config, AWS Direct Connect, and Amazon VPC
• Enforces corporate standards
• Audit for compliance

Provisions new Accounts in 10 minutes vs 10 hours in earlier manual process

Open source and extensible: https://github.com/alanwill/aws-tailor
AWS Ops Automator

- Amazon CloudWatch: Time-based events
- AWS Lambda: Event handler
- AWS Lambda: Task executors
- Amazon CloudWatch: Logs
- AWS SNS: Error and warning notifications
- Amazon DynamoDB: Task configuration & tracking
- Amazon EC2
- Amazon Redshift
- Resources in multiple AWS Regions and Accounts

Image recognition and processing

Amazon Cognito:
User authentication

Amazon S3:
Image uploads

Start state machine execution

AWS Step Functions:
Workflow orchestration

Store meta-data and tags

Amazon DynamoDB:
Image meta-data & tags

Amazon Rekognition:
Object detection

Invoke Amazon Rekognition

Generate image thumbnail

https://github.com/awslabs/lambda-refarch-imagerecognition
Stream processing characteristics

- High ingest rate
- Near real-time processing (low latency from ingest to process)
- Spiky traffic (lots of devices with intermittent network connections)
- Message durability
- Message ordering
AWS Lambda + Amazon Kinesis

Real-time data processing:

1. Real-time event data sent to **Amazon Kinesis** allows multiple **AWS Lambda** functions to process the same events.

2. In **AWS Lambda**, Function 1 processes and aggregates data from incoming events, then stores result data in **Amazon DynamoDB**.

3. Lambda Function 1 also sends values to **Amazon CloudWatch** for simple monitoring of metrics.

4. In **AWS Lambda** function, Function 2 does data manipulation of incoming events and stores results in **Amazon S3**.

Sensor data collection

IoT Sensors

AWS IoT: Data collection

MQTT

IoT rules

IoT actions

Amazon S3: Raw records

Amazon Kinesis Streams: Real-time stream

Amazon Kinesis Firehose: Delivery stream

Real-time analytics applications

Amazon S3: Batched records
Batch processing

Amazon S3 Object -> AWS Lambda: Splitter -> AWS Lambda: Mappers -> Amazon DynamoDB: Mapper Results -> AWS Lambda: Reducer -> Amazon S3 Results

Analytics & Processing
Fannie Mae Serverless Financial Modeling

Financial Modeling is a Monte-Carlo simulation process to project future cash flows, which is used for managing the mortgage risk on a daily basis:

- Underwriting and valuation
- Risk management
- Financial reporting
- Loss mitigation and loan removal

- ~10 Quadrillion ($10 \times 10^{15}$) of cash flow projections each month in hundreds of economic scenarios.
- One simulation run of ~20 million mortgages takes 1.4 hours, >4 times faster than the existing process.
Serverless Computing and Applications

Build and run applications without thinking about servers

Get Started

Build Serverless Applications for Production

Serverless computing allows you to build and run applications and services without thinking about servers. Serverless applications don't require you to provision, scale, and manage any servers. You can build them for virtually any type of application or backend service, and everything required to run and scale your application with high availability is handled for you.

Building serverless applications means that your developers can focus on their core product instead of worrying about managing and operating servers or runtimes, either in the cloud or on-premises. This reduced overhead lets developers reclaim time and energy that can be spent on developing great products which scale and that are reliable.
Thank you!