Basics of Cloud Computing – Lecture 7

More AWS & Serverless Computing

Pelle Jakovits
Satish Srirama
Outline

• Overview of Amazon Web Services

• Serverless computing
  – FaaS Model
  – Apache OpenWhisk
  – Advantages and disadvantages
Cloud Providers and Services we have discussed

• Amazon Web Services
  – **Compute:** EC2
  – **Storage:** S3, EBS
  – **Scaling:** Elastic Load Balancing, Auto Scale, CloudWatch
• Eucalyptus
• OpenStack
• Management providers
  – AWS Management Console
  – OpenStack Horizon
  – RightScale
• PaaS
  – Google AppEngine
  – Windows Azure
  – Elastic MapReduce
AWS services we will discuss

- Management Console
- Identity and Access Management
- CloudFormation
- Data Services
- Data Pipelines
- Data migration services
AWS Management Console

• Hope some of you have started using Amazon accounts
• You can manage your complete Amazon account with management console
  – AMI Management
  – Instance Management
  – Security Group Management
  – Elastic IP Management
  – Elastic Block Store
  – Key Pair management
  – etc.
• Have different pages for different services
AWS EC2 DashBoard

Resources

You are using the following Amazon EC2 resources in the EU West (Ireland) region:

- 0 Running Instances
- 0 Dedicated Hosts
- 0 Volumes
- 17 Key Pairs
- 0 Placement Groups
- 0 Elastic IPs
- 1 Snapshots
- 0 Load Balancers
- 32 Security Groups

Learn more about the latest in AWS Compute from AWS re:Invent by viewing the EC2 Videos.

Create Instance

To start using Amazon EC2 you will want to launch a virtual server, known as an Amazon EC2 instance.

Launch Instance

Note: Your instances will launch in the EU West (Ireland) region.

Service Health

Service Status:
- EU West (Ireland):

Availability Zone Status:
- eu-west-1a:
  - Availability zone is operating normally
- eu-west-1b:
  - Availability zone is operating normally
- eu-west-1c:
  - Availability zone is operating normally

Scheduled Events

EU West (Ireland):
No events
AWS Identity and Access Management (IAM)

• How can an enterprise or group of users use one credit card?
• Manage IAM users
  – Create new users and manage them
  – Create groups
• Manage credentials
  – Create and assign temporary security credentials
• Manage permissions
  – Creating policies for specific services and users
  – Can use very fine-grained granularity
IAM policy example

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ec2:DescribeInstances", "ec2:DescribeImages",
        "ec2:DescribeKeyPairs", "ec2:DescribeVpcs",
        "ec2:DescribeSubnets", "ec2:DescribeSecurityGroups"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "ec2:RunInstances",
      "Resource": [
        "arn:aws:ec2:sa-east-1:111122223333:key-pair/**",
        "arn:aws:ec2:sa-east-1:111122223333:subnet/subnet-1a2b3c4d"
      ]
    }
  ]
}
```
IAM policy example

```json
{
    "Effect": "Allow",
    "Action": "ec2:RunInstances",
    "Resource": [
        "arn:aws:ec2:sa-east-1:11112222333:instance/*"
    ],
    "Condition": {
        "StringEquals": {
            "ec2:InstanceType": "m1.small"
        }
    }
},
{
    "Effect": "Allow",
    "Action": "ec2:RunInstances",
    "Resource": [
        "arn:aws:ec2:sa-east-1::image/ami-*"
    ],
    "Condition": {
        "StringEquals": {
            "ec2:Owner": "amazon"
        }
    }
}
AWS CloudFormation

• Provides an easy way to create and manage a collection of related AWS resources, provisioning and updating them in an orderly and predictable fashion
• It is based on templates model
  – Templates describe the AWS resources, the associated dependencies, and runtime parameters to run an app.
  – The templates describe stacks, which are set of software and hardware resources.
  – Something similar to CloudML and RightScale server templates
• Hides several details
  – How the AWS services need to be provisioned
  – Subtleties of how to make those dependencies work.
Resources:

**Ec2Instance:**
Type: 'AWS::EC2::Instance'
Properties:
  SecurityGroups:
    - !Ref InstanceSecurityGroup
    - MyExistingSecurityGroup
  KeyName: mykey
  InstanceType: t2.micro
  ImageId: ami-7a11e213

**InstanceSecurityGroup:**
Type: 'AWS::EC2::SecurityGroup'
Properties:
  GroupDescription: Enable SSH access via port 22
  SecurityGroupIngress:
    - IpProtocol: tcp
      FromPort: '22'
      ToPort: '22'
      CidrIp: 0.0.0.0/0
AWS CloudFormation

• Amazon provides several pre-built templates to start common apps, such as:
  – WordPress (blog)
  – LAMP stack (Linux, Apache, MySQL, and PHP)
  – Gollum (wiki used by GitHub)

• There is no additional charge for AWS CloudFormation.

• You pay for the utilized AWS resources (e.g. EC2 instances, Elastic Load Balancers, etc.)

AWS Data Services

- **Amazon Relational Database Service**
  - Scalable and re-sizeable SQL DB service
  - Supports most of the familiar database engines
    - MySQL, PostgreSQL, Oracle, Microsoft SQL Server

- **Amazon Aurora**
  - High Performance SQL service (MySQL and PostgreSQL compatible)
  - Distributed, fault-tolerant, self-healing storage

- **NoSQL databases**
  - DocumentDB - Managed Document DB (MongoDB compatible)
  - DynamoDB – Managed NoSQL database
  - Neptune – Managed Graph Database
AWS Data Pipeline

- Service for orchestrating data movement and processing tasks inside AWS

- **DataNode** - Location of the data source or destination. (SqlDataNode, S3DataNode, ...)

- **Activity** - Operation to perform on data (CopyActivity, EmrActivity, ...)

- **Schedule** - When data pipelines activities are initiated (On-demand, CRON, ...)

- **Precondition** - Conditions for when pipeline tasks can be executed

- **Resource** - EC2 resources or other AWS services Activities depend on

- Open Source alternative: **Apache NiFi**
AWS Snowball

- Device for secure and physical Data Migration
- Storage capacity: 50 – 80TB
- Migrate Big Data: analytics data, genomics data, video libraries, image repositories, backups, etc.
AWS Snowball Edge

• Device for Data Migration together with Onboard pre-processing capability.

• Onboard compute service options:
  – AWS Lambda, EC2 AMIs
  – 26 or 52 vCPUs
  – **Optional GPU** for machine learning and real time video analysis

• Storage capacity: 100TB

• Clustering: Up to 20 nodes
AWS Snowmobile

- Petabyte- to Exabyte-scale data transfer service for moving extremely large amounts of data to AWS
- 100 PB of Data => as much as 1,250 AWS Snowball devices
- Pricing:
  - $0.005/GB per month
  - $5243/PB per month
Other AWS Services

- **Analytics** (EMR, Athena serverless queries, Kinesis video analytics)
- **Machine Learning** (DeepLens, SageMaker, Lex)
- **Application Integration** (Message Queue, Step Functions)
- **Internet of Things**
- **Media Services** (Transcoding pipelines, stream management)
- **Networking** (VPC, Domains)
- **Augmented & Virtual Reality**
- **RoboMaker** (Cloud services for Robot Operating System)
- **Security** (User management, Firewalls, Data privacy tools)
- **BlockChain**
A BIT MORE ON SERVERLESS COMPUTING
What is Serverless?

- Abstraction of servers
- Event-driven/instant scale
- Sub-second billing
Serverless computing - continued

• Newer workloads are a better fit for event driven programming
  – Execute application logic in response to database triggers
  – Execute app logic in response to sensor data
  – Execute app logic in response to scheduled tasks etc.

• Applications are charged by compute time (millisecond) rather than by reserved resources

• Greater linkage between cloud resources used and business operations executed

• Serverless in a nutshell
  – Event-action platforms to execute code in response to events
Current platforms for serverless

- OpenWhisk
- AWS Lambda
- Azure Functions
- Google Functions
- fission
- OpenLambda
- Red-Hat

3/27/2019
FaaS in Public Clouds

• **AWS Lambda**
  – Run code in AWS without managing infrastructure or software
  – Java, Go, PowerShell, Node.js, C#, Python, and Ruby code
  – Pricing is based on number of **requests** and **GB-Sec "Memory-Duration"**
  – Free: 1M **requests** a month. After: $0.20 per 1M
  – Free: 400,000 **GB-Sec**. After: $0.000017 per 1 **GB-Sec**

• **IBM BlueMix Cloud Function**
  – Based on OpenWhisk - Open Source Serverless cloud platform
  – Event, trigger & rule based execution
  – Supports any language*
  – Free: 400,000 **GB-Sec**. After: $0.000017 per 1 **GB-Sec**
Apache OpenWhisk

- Initiated by IBM but now an Apache project
- Open source cloud platform
- Serverless deployment and operations model
- Optimal utilization and granular pricing
- Scales on a per-request basis
- Supports JS, Swift, Python, Java, Docker
Triggers, actions, rules (and packages)

• Services or data sources define the events they emit as **triggers**
• Developers associate the **actions** to handle the events via **rules**
• **Packages** are a shared collection of Triggers and Actions
Triggers and Rules

• Trigger examples
  – changes to database records
  – IoT sensor readings that exceed a certain temperature
  – new code commits to a GitHub repository
  – simple HTTP requests from web or mobile apps

• Rule is an association of a trigger to an action
  – Many to many mapping
Actions

• They can be
  – small snippets of JavaScript or Swift code
  – custom binary code embedded in a Docker container

• Instantly deployed and executed whenever a trigger fires

• It is also possible to directly invoke an action by using the OpenWhisk API, CLI, or iOS SDK

• A set of actions can be chained

CLI – Command line interface
# Hello world as an OpenWhisk action.

def myFunction(args):
    name = args['name']
    greeting = "Hello " + name + '!
    return {"greeting": greeting}

• Create an action:
  wsk action create myfunction Hello-Python.py

• Invoke an action:
  wsk action invoke myfunction --result --param name World

• Update an action:
  wsk action update myfunction Hello-Python.py
System overview

[Diagram of System Overview]

CRUD triggers, actions, and rules
Invoke actions

REST

Package Feed
Package Feed
Package Feed
Package Feed

Trigger
Rule
Action NodeJS
Rule
Action Swift
Rule
Action Docker
Rule
Action NodeJS
Rule
Action Docker

Invoke

Service ecosystem
- Bluemix services
- 3rd party services
- Self-enabled services

https://github.com/apache/incubator-openwhisk/blob/master/docs/about.md
The internal flow of processing

For more information read https://github.com/apache/incubator-openwhisk/blob/master/docs/about.md
Advantages of Serverless/FaaS

- Very simple and "cheap" to scale
- Rapid prototyping
- Easy to modify serverless functions
- Pay only for the execution time, not for idle computation time
- Can create applications by composing functions written in different languages
Disadvantages of Serverless

• Harder to avoid vendor lock-in
  – Depend heavily on built in triggers and rules
• Lack of monitoring and debugging tools
• Composition and architecture complexity
• Slow cold-start
• What about stateful computations?
• Harder to predict costs
Next labs

• This week Lab
  – Continue working with Google AppEngine

• Next week Lab
  – Cloud Functions in IBM Bluemix (Managed OpenWhisk service)
Next Lecture

• Overview of Mobile & Cloud Lab research
• Cloud computing challenges

• NB! Opportunity to ask additional questions about the exam
References

• Check Amazon videos and webinars at http://aws.amazon.com/resources/webinars/
• Apache OpenWhisk - https://github.com/apache/incubator-openwhisk/blob/master/docs/about.md