IAC - Python Boto3

-- Chidanand

Introduction to IAC

IAC Tools

- Cloud Specific
  - AWS & Azure => Terraform (https://www.terraform.io)
  - AWS => CloudFormation & OpsCode
  - Python Boto3
- Bare Metal
  - Puppet, Chef, ANSIBLE, SaltStack

Python Boto3

- AWS SDK for Python developers
- Write Python code for managing AWS Components & Services
- Easy to Use, Object Oriented API

Boto3 Installation

- Python ver 3.6.3 or above (https://www.python.org/downloads/windows/)
- Check python path by running command line
  >python –version
- Install Boto3 using pip command

  >pip install boto3

Boto3 Configuration

- Create a new user ‘botouser’(programmatic access) in AWS and let us specify EC2 and S3 Full permission

- AmazonEC2FullAccess
- AmazonS3FullAccess
- Download & Save Access Key ID & Secret Access Key

Simple Storage Service - S3

- Object based storage in the Cloud
- Independent of any server and is accessed over internet
- Files are stored in Buckets (simple flat folder without any hierarchy)
- Bucket name must be unique across all AWS Account (like a domain name)
- Buckets are region specific

Simple Storage Service

- Unlimited capacity but one object can be up to 5TB (multi-part upload for larger files)
- HTTP API/REST Interface
  - Create : HTTP PUT(POST)
  - Read : HTTP GET
  - Delete : HTTP DELETE
  - Update : HTTP POST (PUT)

S3 – Other Attributes

- Versioning
- Encryption
- Replication across Regions
- MFA for delete operations
- Various Storage Class (Pricing difference)
  - S3 Standard
  - S3 – IA (Infrequently Accessed)
  - S3 One Zone

S3 – Lab Exercise

- Create 2 Buckets in any one specific Region
- Unique Bucket Names
- Upload few files to it manually

Boto3 S3

- Set up Boto3 to use AWS user credentials that we created earlier
- One way to do it is by manually creating **~/.aws/credentials** with following content

```
[default]
aws_access_key_id = YOUR_ACCESS_KEY
aws_secret_access_key = YOUR_SECRET_KEY
```

Boto3 S3

- You can also set up a default region if you need or this can also be passed as a parameter dynamically - **~/.aws/config**

```
[default]
region=us-east-1
```

- **For our exercise, please choose which ever region where-in you have manually created your Buckets**

Lab – Print All Buckets in Region

```
import boto3

# Let's use Amazon S3
s3 = boto3.resource('s3')

# Print out bucket names
for bucket in s3.buckets.all():
    print(bucket.name)
```

Lab – Upload Multi-part file to S3

```
import boto3

# Create an S3 client
s3 = boto3.client('s3')

filename = 'file2.txt'
bucket_name = 'chid-bucket22'

# Uploads the given file using a managed uploader, which will split up large
# files automatically and upload parts in parallel.

try:
    response = s3.upload_file(filename, bucket_name, filename)
    print (response)
except Exception as error:
    print (error)
```

Download File from S3

- First Upload a JPEG or PNG file to your choice of S3 Bucket

```
s3.Bucket(BUCKET_NAME).download_file(KEY,
'my_local_image.jpg')
```

AWS – Elastic Compute Cloud (EC2)

- Scalable Computing Capacity Service
- Instances – Virtual computing environment
- Amazon Machine Images (AMIs) – preconfigured templates for your instances
- Various Configuration of CPUS, Memory, Storage, networking capacity
- Key Pairs for secure access of instances
- Default users : **ec2-user (**Amazon AMI) & **ubuntu**

AWS – Elastic Compute Cloud (EC2)

- Across multiple physical locations
- Security Groups to control access – ports, protocols, IP range (ACLs)
- Elastic IP Address
- Tags to monitor & audit instances

EC2 Compute Basics

- Instance Types
  - Virtual CPUs (vCPUs)
  - Memory
  - Storage (size and type)
  - Network performance
  - C4 : Compute Optimized (Cpu)
  - R3 : Memory Optimized (Ram)
  - i2 : Storage Optimized (Iops)
  - g2 : GPU-based instance (Gpu)
  - t2 : General Purpose

EC2 – Demo & Exercise

- Create an EC2 Instance – Free Tier Eligible one – Amazon Linux AMI – t2 micro
- Add lots of tags to Identify the instance
  - Name : My-Web-EC2-Server
  - Department : ACCENTURE-IT-TEAM
  - Team : Web-DevTeam
  - StaffID : 9412

EC2 – Demo & Exercise

- Create a Security Group – MyWebDMZ – and allow SSH, HTTP & HTTPS traffic
- New pair of SSH Keys (**ACCENTURE_KEYS**)
- Download SSH Keys (PEM file)
- Use PuttyGen to Convert PEM to PPK
- Username : ec2-user@IP-ADDRESS
- SSH to EC2 instance

EC2 DEMO

ALWAYS SHUTDOWN WHEN YOU DO NOT NEED THE INSTANCE!!!!

Boto3 – Create Security Group

```
ec2.create_security_group
(GroupName='ACCENTURE_SEC_GROUP',
  Description='Created using Boto3',
  VpcId=vpc_id)
```

Boto3 – Spin EC2 Instances

- Choose the same Region
- Use '**ACCENTURE_KEYS'**
- Use 'ACCENTURE_SEC_GROUP'
- Pick Amazon AMI - ami-cfe4b2b0

Boto3 – Spin EC2 Instances

- ec2.create_instances(

MinCount, MaxCount, ImageId, InstanceType,

SecurityGroups, KeyName,……)

- instance.wait_until_running()
- instance.id, instance.state, instance.public_dns_name

Questions?