AWS CLI 이용한 LAB 구축 & 삭제

엄기성 / awskrug CLI 소모임 / 2018.07.31

발표자 소개

- 이름: 엄기성(GiSeong Eom)
- 회사: 판교 B모 게임회사
- 업무: (Cloud) Infra. Ops
- 기타: https://github.com/giseongeom

Disclaimer

- 이 슬라이드의 내용은 전적으로 발표자 개인 의견입니다.
- 소속 부서, 고용주의 정책/의견과 무관함을 미리 밝혀 둡니다.

목차

- CLI 준비
- LAB 생성
- LAB 삭제
- 주의사항 & 팁

CLI 준비

작업 환경

- Ubuntu 16.04 (Windows Subsystem for Linux / Windows 10 1803)
- AWS CLI 1.15.67
- PowerShell Core 6.0.3 / Ubuntu 16.04 (x64)
- AWSPowerShell.NetCore 3.3.313.0

AWS CLI 간단 소개

- https://github.com/aws/aws-cli
- CLI : Command Line Interface
- Python-based - 설치방법이 다양함
- 이기종 지원 (Windows, Mac, Linux/UNIX)
- AWS Tools for PowerShell과 비교

AWS CLI 설치

- https://docs.aws.amazon.com/cli/latest/userguide/installing.html
- OS별 Package Manager
- Python PIP 추천
- Bundled Installer
- Amazon Linux

AWS CLI 구성

```
$ aws configure

AWS Access Key ID [None]: ABCDEFGH**********
AWS Secret Access Key [None]: abcdefgh**********
Default region name [None]: ap-northeast-2
Default output format [None]: json
```

IAM 계정 생성

```
$ aws iam create-user --user-name lab

# WebConsole Access
$ aws iam create-login-profile \
  --user-name lab --password '1SuperF@stx'

$ aws iam create-access-key --user-name lab
{
    "AccessKey": {
        "UserName": "lab",
        "Status": "Active",
        "CreateDate": "2018-07-31T02:05:10.744Z",
        "SecretAccessKey": "JLCFWS0z**********",
        "AccessKeyId": "AKIAIUDO442ZCK5WJUYA"
    }
}
```

- SecretAccessKey는 create-access-key 명령 실행할 때만 표시됨

IAM Group 생성

```
$ aws iam create-group --group-name HOLGroupUsers
$ aws iam attach-group-policy \
  --policy-arn arn:aws:iam::aws:policy/AdministratorAccess \
  --group-name HOLGroupUsers
$ aws iam list-attached-group-policies \
  --group-name HOLGroupUsers
{
  "AttachedPolicies": [
   {
     "PolicyName": "AdministratorAccess",
     "PolicyArn": "arn:aws:iam::aws:policy/AdministratorAccess"
   }
  ]
}
$ aws iam add-user-to-group \
  --group-name HOLGroupUsers
  --user-name lab
```

Profile 생성

```
$ aws configure --profile lab

AWS Access Key ID [None]: AKIAIUDO442ZCK5WJUYA
AWS Secret Access Key [None]: JLCFWS0z**********
Default region name [None]: ap-northeast-2
Default output format [None]: json

$ aws ec2 describe-regions --profile lab
$ aws iam get-user --profile lab --query.Arn

# AccountNumber - IAM Login URL
# https://${AccountNumber}.signin.aws.amazon.com/console
$ aws iam get-user --profile lab --query User.Arn | cut -d: -f5
```

Default profile 지정

```
$ export AWS_PROFILE=lab
$ aws iam get-user
{
    "User": {
        "UserName": "lab",
        "PasswordLastUsed": "2018-07-31T02:35:38Z",
        "CreateDate": "2018-07-31T01:19:45Z",
        "UserId": "AIDAJKMZ6OSCOUEB4KSGI",
        "Path": "/",
        "Arn": "arn:aws:iam::569142980887:user/lab"
    }
}
```

- https://docs.aws.amazon.com/cli/latest/userguide/cli-environment.html
- `AWS_PROFILE` 환경변수 사용해서 현재 세션에서만 적용

LAB 생성

LAB에서 만들 AWS 리소스

- 1 VPC with 2 subnet
- EC2 SSH Keypair
- 2 EC2 instances

VPC 생성

```
# VPC
$ aws ec2 create-vpc --cidr-block 10.0.0.0/16
$ aws ec2 describe-vpcs \
  --query "Vpcs[?CidrBlock == '10.0.0.0/16'].VpcId"
$ vpcid=$(aws ec2 describe-vpcs \
  --query "Vpcs[?CidrBlock == '10.0.0.0/16'].VpcId" --output text)

# Subnet
$ aws ec2 create-subnet --vpc-id $vpcid --cidr-block 10.0.0.0/24
$ aws ec2 create-subnet --vpc-id $vpcid --cidr-block 10.0.1.0/24
```

- AWS 설명서 - AWS CLI를 사용하여 IPv4 VPC 및 서브넷 생성

VPC 생성 (cont'd)

```
# Internet Gateway
$ aws ec2 create-internet-gateway
$ aws ec2 describe-internet-gateways \
  --query "InternetGateways[?Attachments[0].State == null]" \
  | jq .[0].InternetGatewayId

$ igwid=$(aws ec2 describe-internet-gateways \
  --query "InternetGateways[?Attachments[0].State == null]" \
  | jq -r .[0].InternetGatewayId)

# Attach Internet Gateway to VPC
$ aws ec2 attach-internet-gateway \
  --vpc-id $vpcid \
  --internet-gateway-id $igwid
```

VPC 생성 (cont'd)

```
# (Aleady-attached) Internet Gateway
$ aws ec2 describe-internet-gateways \
  --query "InternetGateways[?Attachments[0].VpcId=='$vpcid']" \
  | jq .[0].InternetGatewayId

$ igwid=$(aws ec2 describe-internet-gateways \
  --query "InternetGateways[?Attachments[0].VpcId=='$vpcid']" \
  | jq -r .[0].InternetGatewayId)

# Route Table
$ aws ec2 create-route-table --vpc-id $vpcid
$ aws ec2 describe-route-tables --query \
"RouteTables[?VpcId=='$vpcid'] | [?Associations[0].Main == null]" \
| jq .[].RouteTableId

$ user_rtableid=$(aws ec2 describe-route-tables --query \
"RouteTables[?VpcId=='$vpcid'] | [?Associations[0].Main == null]" \
| jq -r .[].RouteTableId)
```

VPC 생성 (cont'd)

```
$ aws ec2 create-route --route-table-id $user_rtableid \
  --destination-cidr-block 0.0.0.0/0 --gateway-id $igwid

# Subnet
$ aws ec2 describe-subnets --query \
"Subnets[?VpcId=='$vpcid'] | [?starts_with(CidrBlock, '10.0.1')]" \
| jq .[0].SubnetId

$ subnetid=$(aws ec2 describe-subnets --query \
  "Subnets[?VpcId=='$vpcid'] \
  | [?starts_with(CidrBlock, '10.0.1')]" \
  | jq -r .[0].SubnetId)
```

VPC 생성 (cont'd)

```
$ aws ec2 associate-route-table \
  --subnet-id $subnetid \
  --route-table-id $user_rtableid

$ aws ec2 modify-subnet-attribute \
  --subnet-id $subnetid --map-public-ip-on-launch

$ aws ec2 describe-route-tables --query \
    "RouteTables[?VpcId=='$vpcid'] | \
    [?Associations[0].SubnetId == '$subnetid']" \
    | jq .[0].Associations[0].RouteTableAssociationId

$ user_rtable_aid=$(aws ec2 describe-route-tables --query \
    "RouteTables[?VpcId=='$vpcid'] | \
    [?Associations[0].SubnetId == '$subnetid']" \
    | jq -r .[0].Associations[0].RouteTableAssociationId)
```

EC2 SSH Keypair 등록

```
$ aws ec2 import-key-pair --key-name vagrant \
  --region ap-northeast-2 \
  --public-key-material file://vagrant.pub

$ aws ec2 describe-key-pairs
{
  "KeyPairs": [
    {
      "KeyName": "vagrant",
    }
  ]
}
```

EC2 Launch

```
# Latest AmazonLinux 2.0 LTS ImageId
$ aws ec2 describe-images --owner amazon --query \
  "Images[?starts_with(Name, 'amzn2-ami-hvm-2.0')] \
  | sort_by([], &CreationDate) \
  | reverse([])[0].[Name,Description,ImageId,CreationDate]"

[
  "amzn2-ami-hvm-2.0.20180622.1-x86_64-gp2-dotnetcore-2018.07.11",
  "Amazon Linux 2 LTS AMI with .NET Core 2.1",
  "ami-bb0bbfd5",
  "2018-07-14T01:17:16.000Z"
]

$ ami_id=$(aws ec2 describe-images --owner amazon --query \
  "Images[?starts_with(Name, 'amzn2-ami-hvm-2.0')] \
  | sort_by([], &CreationDate) \
  | reverse([])" | jq -r .[0].ImageId)
```

EC2 Launch (cont'd)

```
$ aws ec2 create-security-group \
  --group-name allow-lab-access --description lab --vpc-id $vpcid

$ aws ec2 describe-security-groups --query \
  "SecurityGroups[?VpcId=='$vpcid'] \
  | [?contains(Description, 'lab')]"

$ sg_id=$(aws ec2 describe-security-groups --query \
    "SecurityGroups[?VpcId=='$vpcid'] \
    | [?contains(Description, 'lab')]" \
  | jq -r .[0].GroupId)

$ aws ec2 authorize-security-group-ingress \
  --group-id $sg_id --protocol tcp --port 22 --cidr 0.0.0.0/0
```

EC2 Launch (cont'd)

```
$ aws ec2 run-instances --image-id $ami_id \
  --count 2 --instance-type t2.micro \
  --key-name vagrant \
  --security-group-ids $sg_id \
  --subnet-id $subnetid

$ aws ec2 describe-instances --query \
  "Reservations[].Instances[].[PublicIpAddress,State.Name]" \
  --output table

---------------------------------
|      DescribeInstances        |
+-----------------+-------------+
|  52.78.240.178  |  running    |
|  13.125.239.197 |  running    |
+-----------------+-------------+
```

LAB 삭제

순서가 중요

- 리소스 생성할 때의 순서와 반대로 진행
- attach, add, associate 이런 키워드에 주의

EC2 Terminate

```
$ ids=$(aws ec2 describe-instances --query \
"Reservations[].Instances[?VpcId=='$vpcid'].InstanceId"\
--output text)

$ aws ec2 terminate-instances --instance-ids $ids
```

EC2 Security Group 삭제

```
$ sg_id=$(aws ec2 describe-security-groups --query \
    "SecurityGroups[?VpcId=='$vpcid'] \
    | [?contains(Description, 'lab')]" \
  | jq -r .[0].GroupId)

$ aws ec2 delete-security-group --group-id $sg_id
```

EC2 SSH Keypair 삭제

```
$ aws ec2 describe-key-pairs
$ aws ec2 delete-key-pair --key-name vagrant
```

VPC 삭제

```
$ user_rtableid=$(aws ec2 describe-route-tables --query \
    "RouteTables[?VpcId=='$vpcid'] | \
    [?Associations[0].SubnetId == '$subnetid']" \
  | jq -r .[].RouteTableId)

$ user_rtable_aid=$(aws ec2 describe-route-tables --query \
    "RouteTables[?VpcId=='$vpcid'] | \
    [?Associations[0].SubnetId == '$subnetid']" \
  | jq -r .[0].Associations[0].RouteTableAssociationId)

$ aws ec2 disassociate-route-table \
  --association-id $user_rtable_aid

$ aws ec2 delete-route \
  --route-table-id $user_rtableid \
  --destination-cidr-block 0.0.0.0/0

$ aws ec2 delete-route-table --route-table-id $user_rtableid
```

VPC 삭제 (cont'd)

```
$ igwid=$(aws ec2 describe-internet-gateways \
  --query "InternetGateways[?Attachments[0].VpcId=='$vpcid']" \
  | jq -r .[0].InternetGatewayId)

$ aws ec2 detach-internet-gateway \
  --vpc-id $vpcid \
  --internet-gateway-id $igwid

$ aws ec2 delete-internet-gateway \
  --internet-gateway-id $igwid
```

VPC 삭제 (cont'd)

```
$ subnet_ids=$(aws ec2 describe-subnets --query \
  "Subnets[?VpcId=='$vpcid'].SubnetId" --output text)

$ for mynet in $subnet_ids
> do aws ec2 delete-subnet --subnet-id $mynet
> done

$ aws ec2 delete-vpc --vpc-id $vpcid
```

IAM User/Group 삭제

```
$ export AWS_PROFILE=default

# 웹 로그인 차단
$ aws iam delete-login-profile --user-name lab

# 그룹에서 제거
$ aws iam remove-user-from-group \
  --group-name HOLGroupUsers --user-name lab


$ user_access_key=$(aws iam list-access-keys --user-name lab \
  --query AccessKeyMetadata[0].AccessKeyId --output text)

$ aws iam delete-access-key \
  --access-key-id $user_access_key --user-name lab
$ aws iam delete-user --user-name lab
```

IAM User/Group 삭제

```
$ aws iam detach-group-policy \
  --policy-arn arn:aws:iam::aws:policy/AdministratorAccess \
  --group-name HOLGroupUsers

$ aws iam delete-group --group-name HOLGroupUsers
```

주의사항 & 팁

Lessons from

- AWS CLI는 1회성 작업에 좋음
- 지속적인 유지보수가 필요하다면 다른 솔루션을 검토하자.
- `JMESPath` Query에 집착하지 말자.

Any Questions?

발표자료/예제소스 다운로드

References

- Advanced AWS CLI JMESPath Query Tricks
- JMESPath Tutorial
- jq Tutorial