Hands-on Cloud Computing Services Lezione 4

Gabriele Russo Russo University of Rome Tor Vergata, Italy

A.A. 2021/22



Previous Lecture: Ansible

- Automated configuration management
- Command-line tool
- We also mentioned: Ansible Tower / AWX
- \blacktriangleright \rightarrow demo

Cloud Automation

We have already seen some tools for automation:

- Ansible (with dynamic inventories)
- AWS CLI
- AWS SDK (e.g., boto3)

Enough for infrastructure management?

Infrastructure-as-Code (IaC)

- Define and manage the infrastructure by means of a set of text files, instead of a user interface (CLI, Web, ...)
- Use simple text files to describe your resources (e.g., VMs, security groups, networks)
- Update the files to update the infrastructure
- Benefits:
 - Reduced costs
 - Reduced risks
 - Faster operations
 - Important to enable DevOps practices

Terraform

- Free tool for IaC
- Available on Linux, macOS, Windows (www.terraform.io)
- Can be used with several platforms (AWS, Azure, VMWare, CloudFlare, ...)
- Infrastructure defined using the HashiCorp Configuration Language (HCL)
- Key concepts: Providers + Resources

Terraform + AWS

Requirements:

- AWS CLI installed and configured
- Terraform installed (I am using Terraform 1.0.10)

We create example.tf and run:

- \$ terraform init
- \$ terraform validate # check syntax
- \$ terraform apply
- \$ terraform show
- \$ terraform apply # nothing to do

We now **update** example.tf adding a tag to the instance:

- Edit example.tf adding a tag to the instance
- \$ terraform apply
- Edit example.tf changing the instance type
- \$ terraform apply
- Let's destroy all the created resources: terraform destroy

Terraform: beyond this example

- Resource definitions not limited to EC2 instances!
- Remote storage for tf.state
- Versioning Terraform code (e.g., git repo)
- Variables to make code more reusable

AWS CloudFormation

- IaC code solution by AWS
- Stack + Template (YAML/JSON) + Resources
- https://docs.aws.amazon.com/AWSCloudFormation/latest/ UserGuide/GettingStarted.Walkthrough.html

AWS CloudFormation: Example

}

```
{
```

```
"AWSTemplateFormatVersion" : "2010-09-09",
"Description" : "A sample template",
"Resources" : {
  "MyEC2Instance" : {
    "Type" : "AWS::EC2::Instance",
    "Properties" : {
      "ImageId" : "ami-Off8a91507f77f867",
      "InstanceType" : "t2.micro",
      "KeyName" : "testkey",
      "BlockDeviceMappings" : [
        ſ
          "DeviceName" : "/dev/sdm",
          "Ebs" : {
            "VolumeType" : "io1",
```

Exercise from the Previous Lecture

Extend PhotoGallery with the following features:

- display pictures stored in a S3 bucket, along with their upload time
- users can upload pictures



- Source code: photogallery_v2
- How to provide credentials to boto3 to access the bucket?
 - Create a IAM Role for EC2
 - Attach the pre-defined S3FullAccess policy
 - Associate the EC2 instance(s) with the new role
- (Check alternative methods in the previous slides)

CloudFront

- CDN provided by AWS
- Easy to integrate with S3 buckets and ELBs

How to use it in Photogallery:

- Create a distribution for our S3 bucket
- Replace picture URLs as follows: http://bucketname.s3.amazonaws.com/imagename.jpg http://distributionname.cloudfront.net/imagename.jpg

Note: to delete a distribution, you need to Disable it first (and wait a couple of minutes)

- DNS service by AWS
- Register a domain name
- Fine-grained control over your DNS zone
- Extra features (e.g., latency-based query handling)

AWS: Database Services

AWS provides several database-oriented services. Among them:

- DynamoDB (Key-Value NoSQL tables)
- Aurora (relational DBMS)
- ElastiCache (in-memory databases: Memcached, Redis)
- Neptune (graph database)
- Timestream (for time series)
- RDS (Relational Database Service): easily deploy MariaDB, Aurora, PostgreSQL, ...

AWS: Database Services

AWS provides several database-oriented services. Among them:

- DynamoDB (Key-Value NoSQL tables)
- Aurora (relational DBMS)
- ElastiCache (in-memory databases: Memcached, Redis)
- Neptune (graph database)
- Timestream (for time series)
- RDS (Relational Database Service): easily deploy MariaDB, Aurora, PostgreSQL, ...

We'll use DynamoDB to store picture metadata in PhotoGallery

DynamoDB

- Schemaless
- Tables, Items, Attributes
- Primary Key + (optional) Sorting Key
- 2 pricing models:
 - provisioned capacity (default)
 - on-demand
- 2 consistency models:
 - eventual
 - strong

Example: dynamodb_example/

Photogallery + DynamoDB

Exercise: Use DynamoDB to store image tags

