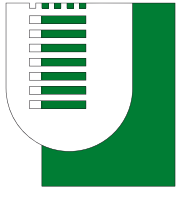
Hands-on Cloud Computing Services

Lecture 1

Gabriele Russo Russo

University of Rome Tor Vergata, Italy

A.A. 2023/24



TOR VERGATA

UNIVERSITÀ DEGLI STUDI DI ROMA

Overview

- Cloud Computing services in action
  - Amazon Web Services
- Tools for cloud automation
  - Ansible
  - Terraform
- Examples & Exercises

Course Material

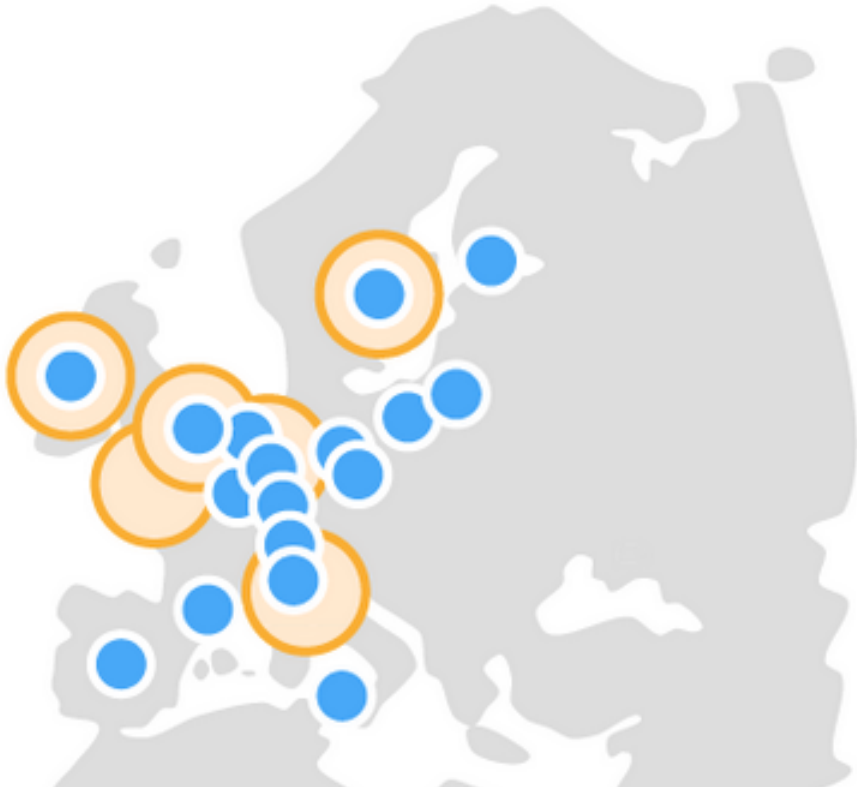
- Relevant slides and code will be published on the web page after the lectures
  - `http://www.ce.uniroma2.it/courses/sdcc2324/`
- Official AWS documentation
  - `https://docs.aws.amazon.com/`
  - most the content available in Italian as well
- If you are interested in a book that covers these topics (and much more): Arshdeep Bahga, Vijay Madisetti, "Cloud Computing Solutions Architect: A Hands-On Approach", 2019.

Amazon Web Services (AWS)

- One of the major public cloud providers (along with Google Cloud Platform, Microsoft Azure, …)
- 200+ different services
  - **Computation** (EC2, Lambda, …)
  - **Storage** (S3, …)
  - **Machine Learning** (Rekognition, Lex, …)
  - **Networking and Content Delivery** (CloudFront, Route 53, …)
  - **Security** (Cognito, …)
  - …

AWS Infrastructure

- 32 **regions** (+5 w.r.t. 2022)
  - in Italy: Milano (since 2020)
- 2+ **availability zones** in each region
- 500 Edge locations
  - In Italy: Milano, Roma, Palermo
- Numbers keep growing…



Accessing AWS

- General approach: register an AWS account (with an associated credit card)
  - It is very important to keep AWS credentials safe!
  - Enable Multi-Factor Authentication for your account
- For this course: AWS Academy Learner Lab

Interacting with AWS

- Web Console
- AWS CLI
- SDK for various languages (e.g., `Boto3`)
- (Infrastructure-as-Code tools)

Note: regardless of the chosen approach, you need to explictly select the AWS Region you want to use. Services activated in one region cannot be managed within a different one.

IAM users

- AWS Identity and Access Management (IAM)
- When you register to AWS, you get a root account
- IAM allows you to create and manage users within your account
- It's a good practice to use an IAM user (with full permissions) rather than the root account (if compromised, an IAM user can be deleted without losing the whole account)[1]

[1] https://docs.aws.amazon.com/IAM/latest/UserGuide/id_users_create.html

Amazon EC2

- Resizable compute capacity in the cloud
- Virtual machines = Instances
- Many different instance types
  - Different amount of vCPUs, memory
  - Different storage
  - Different processors (ARM vs x86 CPUs, GPUs, …)
  - **Different price!**
- On-demand / spot / reserved instances
- Starting from less than 0.01 $/hour

Example

- Let's create an EC2 instance
- **AMI**: we can use Amazon Linux
- **Instance type**: pick something cheap (e.g., `t3.micro/nano`)
- **Security group**: create a new SG
  - Allow SSH and HTTP inbound traffic
- Shutdown behavior: `Terminate`
- Create (if necessary) a new **key pair**
  - Store the private key in a safe place!
  - Public key automatically installed on the instance

Connecting to the new instance

```
$ ssh -i <file.pem> ec2-user@<Public IP/Public DNS>
```

Key pairs

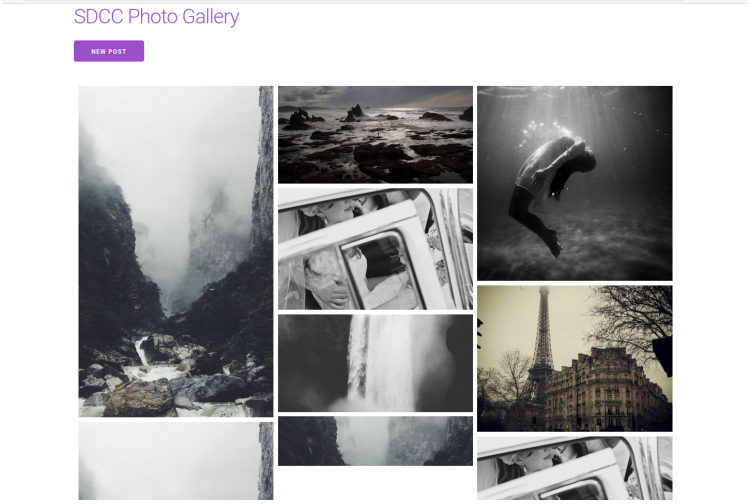
- Key pairs allow EC2 to verify your identity when connecting to an instance via SSH
- Two approaches to setup your key pair:
  1. Key pair created using EC2: you download the private key (`.pem` file) and save it to a **safe** location
  2. Key pair created on your PC (e.g., using `ssh-keygen`): you need to import the **public** key in EC2
- Instructions for both methods: `https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/create-key-pairs.html`

Monitor your Costs!

- The **Billing Dashboard** provides useful information to analyze paid and expected costs
- It is recommended to set an alarm so as to be notified whenever the expected monthly costs exceed a certain value: `https://docs.aws.amazon.com/AmazonCloudWatch/latest/monitoring/monitor_estimated_charges_with_cloudwatch.html`
- **Note**: estimated cost to reproduce steps from this lecture (EC2 + ELB) using services for a whole day: < 1 EUR

Example Application: Photogallery

- Web application, written in Python using Flask
- Just a toy: no security, no robustness, …
- We'll use and extend it during the lectures
- You may develop your own customized version



Roadmap:

Photogallery version 1: static set of images; upload not supported

…

Photogallery version N: image upload and tagging; image search; automatic resizing and tagging based on object recognition; …

Deploying Photogallery on EC2

Running Photogallery

```
$ export FLASK_APP=galleryApp.py
$ flask run -h 0.0.0.0 -p <numero di porta>
$ # Note: \-- requires root privileges for port 80
```

or, using the script `run.sh`:

```
$ bash run.sh
```

- Create a new EC2 instance to deploy the app
- Connect via SSH to the instance:

  ```
  $ ssh -i <file.pem> ec2-user@<Public IP/Public DNS>
  ```

Deploying Photogallery on EC2 (contd.)

- Install the required software:

```
$ sudo yum install pip
$ sudo pip3 install flask
```

- Copy the app files from your PC using `scp`:

```
$ scp -i <chiaveprivata.pem> -r <cartellalocale> \
  ec2-user@<istanza ec2>:/home/ec2-user/
```

- Start the application:

```
$ cd photogallery/
$ bash run.sh
```

- Open `http://EC2-PUBLIC-IP/` in a browser
- Test: what if we “close” port 80 in the security group?

Replicating App Instances

- Current configuration is neither scalable or fault-tolerant
- Let's run multiple replicas of the web server
- We need a **load balancer**