



ÅBO AKADEMI UNIVERSITY

CLOUD COMPUTING

Assignment 3



LUIS ARAÚJO(2004624)

APRIL 19, 2021

Contents

1	Introduction	3
2	Process	4
2.1	Creation of virtual machines	4
3	Questions	8
3.1	Question 1: Why is it recommended to reboot the instance before creating the image?	8
3.2	Question 2: Why does it make sense to assign the new instances in different availability zones?	8
3.3	Question 3: Why in this context (serving a basic php file) we do not need a more complex Application Load Balancer?	8
3.4	Question 4: What is going on? What happen if you terminate only one of the instances running apache?	9
4	Conclusion	10

Chapter 1

Introduction

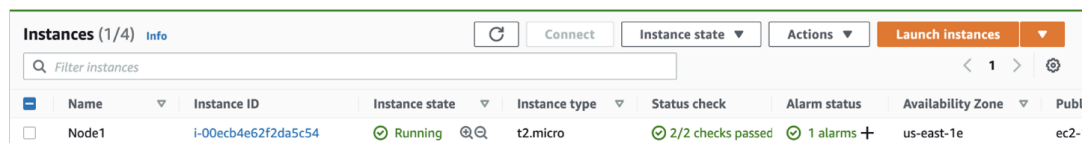
Intending to implement a very simple load-balanced service using AWS IaaS, in this assignment, the teacher proposed an exercise of launching virtual machines, installing and setting up PHP and the apache server, creating and testing an index.php file, and finally launching a load balance with all instances running.

Chapter 2

Process

2.1 Creation of virtual machines

The first step of this assignment was to create an instance of a virtual machine. To launch this virtual machine it was used the Amazon Linux AMI, and the t2.micro instance.



Instances (1/4) Info								
<input type="text" value="Filter instances"/> Launch instances								
	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Publ
<input type="checkbox"/>	Node1	i-00ecb4e62f2da5c54	Running	t2.micro	2/2 checks passed	1 alarms +	us-east-1e	ec2-

Figure 2.1: First Instance of a Virtual Machine

When the first instance was up and running, the next step was to install PHP and the apache server, following by its configuration. For the installation part, it was just needed to run the two command-line codes:

- `sudo yum install httpd`
- `sudo yum install php`

And for the configuration part, it was required to modify the `/etc/rc.local` file, adding `sudo /usr/sbin/httpd -k start` command, such as its demonstrated in the figure 2.2.

```
#!/bin/bash
# THIS FILE IS ADDED FOR COMPATIBILITY PURPOSES
#
# It is highly advisable to create own systemd services or udev rules
# to run scripts during boot instead of using this file.
#
# In contrast to previous versions due to parallel execution during boot
# this script will NOT be run after all other services.
#
# Please note that you must run 'chmod +x /etc/rc.d/rc.local' to ensure
# that this script will be executed during boot.

touch /var/lock/subsys/local
sudo /usr/sbin/httpd -k start
```

Figure 2.2: Configuration of apache server

Continuing to follow the assignment steps, the next procedure was to create an index.php file in /var/www/html directory, with the following piece of code:

```
<?php
// for debugging purpose only, will display all php errors and warnings
ini_set('display_errors', 1);
ini_set('display_startup_errors', 1);
error_reporting(E_ALL);
echo "Hello! My name is Luis_Araujo and my IP address is:
" . $_SERVER['SERVER_ADDR'];
?>
```

Figure 2.3: PHP Code

After, it was time to test if all steps were successfully concluded. By starting the server with the command `sudo /usr/sbin/httpd -k start` and by copying the public DNS of the VM to a web browser we obtain the following.

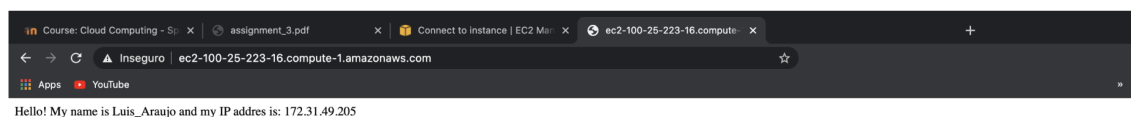


Figure 2.4: First test

Succeeding the first test, an image of this virtual machine could be created. Having this image facilitated the process of launch three more virtual machines, once we could avoid this whole procedure all over again, three more times.

Instances (1/4) Info								
<div> <input type="text" value="Filter instances"/> </div>								
<input type="checkbox"/>	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Publ
<input type="checkbox"/>	Node1	i-00ecb4e62f2da5c54	Running	t2.micro	2/2 checks passed	1 alarms	us-east-1e	ec2-
<input type="checkbox"/>	Node2	i-04f46830be55387c2	Running	t2.micro	Initializing	1/1 has	us-east-1b	ec2-
<input type="checkbox"/>	Nod3	i-0b630b7286b31f5b7	Running	t2.micro	Initializing	1/1 has	us-east-1b	ec2-
<input checked="" type="checkbox"/>	Node4	i-08f8a9304d7bdd5e3	Pending	t2.micro	-	No alarms	us-east-1b	ec2-

Figure 2.5: All virtual machines running

Next, and now having all virtual machines running, the load balance can be created. For this simple assignment, it was chosen the Classic Load Balancer, making sure that port 80 was opened.

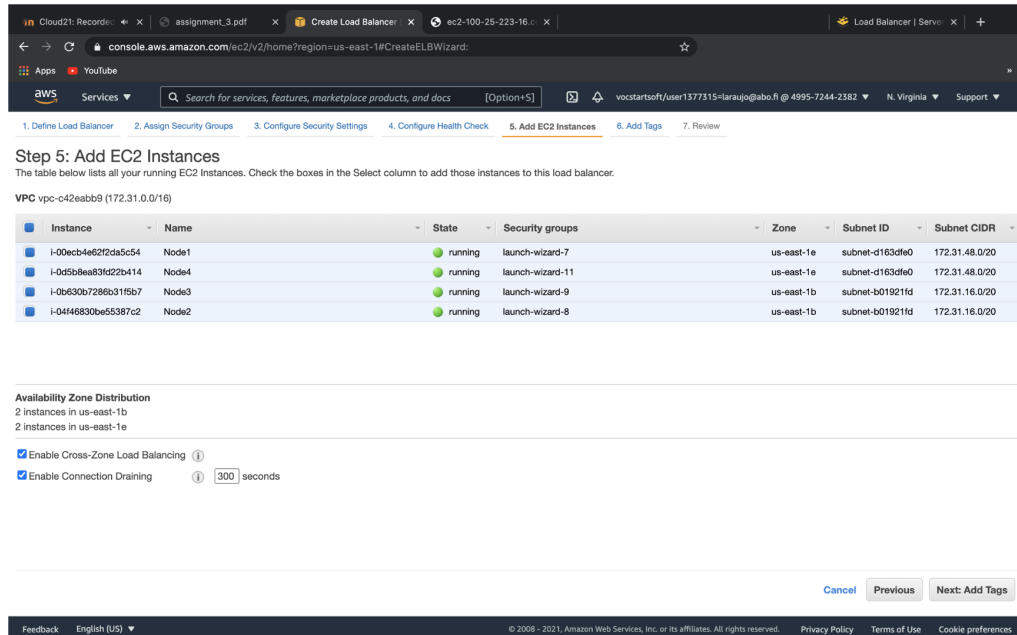


Figure 2.6: Creating Load Balance

After waiting a couple of times, the load balance was running in about 2 minutes.

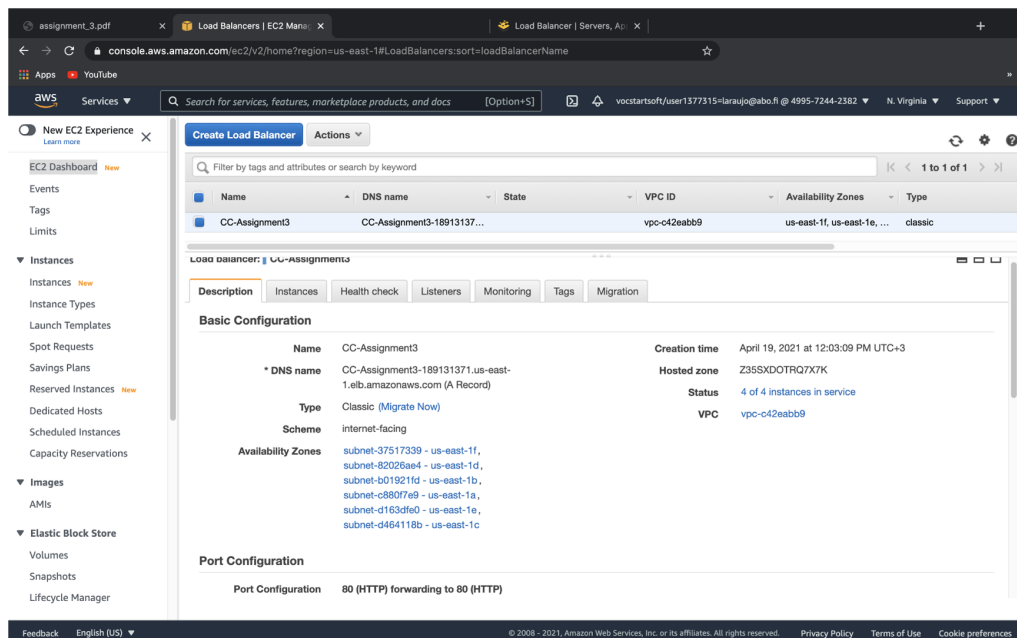


Figure 2.7: Load Balance Running

Finally, and by copying the DNS name to a browser we could test if everything is running correctly. For this test, a video was recorded to show that with the refresh of the page the IP of the virtual machine could change.

The video link of the test: <https://www.youtube.com/watch?v=s2mFdF3SSUw>

Chapter 3

Questions

3.1 Question 1: Why is it recommended to reboot the instance before creating the image?

It is recommended to reboot the instance before creating the image because if the instance is running while creating the image there are no guarantees that the file system integrity will be stable. Rebooting the instance avoids state inconsistencies of the disk. Nothing is in buffer or "moving around" while the image is being created.

3.2 Question 2: Why does it make sense to assign the new instances in different availability zones?

It could make sense to assign the new instances in different availability zones, because if you distribute your instances across multiple Availability Zones and one instance fails, you can design your application so that an instance in another Availability Zone can handle requests. This gives the application more flexibility to launch in production and resources that are highly available, resilient/fault-tolerant, and scalable as compared to using a single data center.

3.3 Question 3: Why in this context (serving a basic php file) we do not need a more complex Application Load Balancer?

In this context, we do not need a more complex Application Load Balancer, because this one works at the request level and routes traffic to targets, IP addresses, Lambda functions, and containers based on content in the request. It can provide sophisticated request routing oriented towards delivering modern application architectures such as container-based and microservices applications. And, with this example, we are dealing with basic load balance functionalities throughout different Amazon EC2 instances.

3.4 Question 4: What is going on? What happen if you terminate only one of the instances running apache?

So, with a Load Balancer the requests are being distributed across different servers, which is a way of increasing the capacity and the reliability of the application, for instance, if we have thousands of requests, a single server will not handle well this substantial amount of requests, so, for that, having a system to slip this requests in different servers is a way of increasing the capacity of the application. Also, if one server fails the Load Balancer will increase the jobs of the other servers, but the application will still be running, which increases the reliability of the application.

Chapter 4

Conclusion

Reflecting on this assignment, what surprised me the most is how easy it is, nowadays, to launch a load balancer, and how satisfying is to see everything running. It also made me realize how these types of systems work.