Åbo Akademi University

Cloud Computing

Assignment 4

Luis Araújo(2004624)

May 7, 2021

Contents

Chapter 1

Introduction

To implement a web service able to automatically scale resources according to the load. These resources should adapt themselves according to the situation, increasing and decreasing as the demand for EC2 instances changes. In this assignment, we are going to export the **Auto Scaling** option from AWS services.

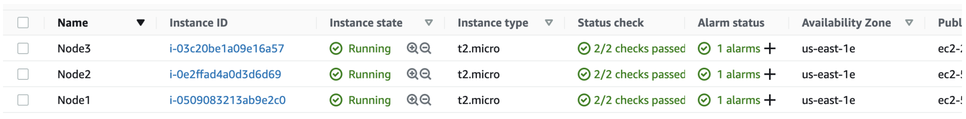
Chapter 2

Process

1) A short description of the steps you took to implement your elastic web service, and the used parameters

First, I started by following the same steps as the last assignment to create four different EC2 instances:

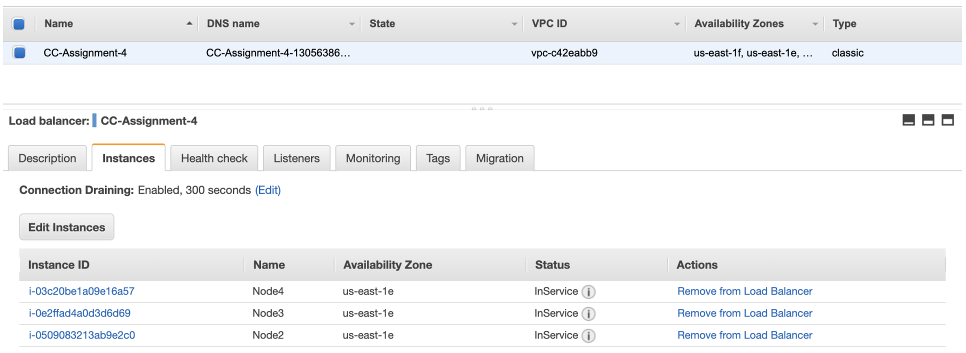
- Launch a EC2 instance
- Install PHP & Apache
- Modify the /etc/rc.local file, adding $sudo\ /usr/sbin/httpd\ -k\ start$ command
- Create an index.php file in the /var/www/html directory
- Generate a AMI image of this first instance
- Launch the other 3 virtual machine with the image created



	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Publ
☐	Node3	i-03c20be1a09e16a57	Running	t2.micro	2/2 checks passed	1 alarms	us-east-1e	ec2-:
☐	Node2	i-0e2ffad4a0d3d6d69	Running	t2.micro	2/2 checks passed	1 alarms	us-east-1e	ec2-!
☐	Node1	i-0509083213ab9e2c0	Running	t2.micro	2/2 checks passed	1 alarms	us-east-1e	ec2-!

Figure 2.1: Instances launched

Consequently, a load balancer was also launched.



	Name	DNS name	State	VPC ID	Availability Zones	Type
■	CC-Assignment-4	CC-Assignment-4-13056386…		vpc-c42eabb9	us-east-1f, us-east-1e, …	classic

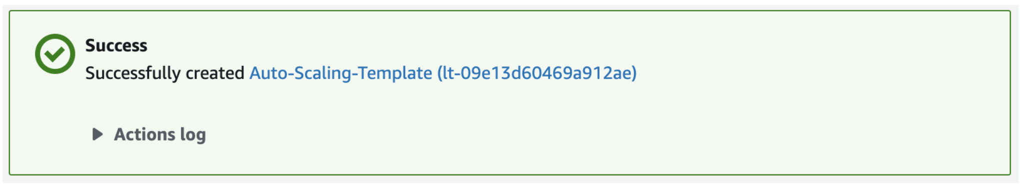
Load balancer: CC-Assignment-4

Connection Draining: Enabled, 300 seconds (Edit)

Instance ID	Name	Availability Zone	Status	Actions
i-03c20be1a09e16a57	Node4	us-east-1e	InService	Remove from Load Balancer
i-0e2ffad4a0d3d6d69	Node3	us-east-1e	InService	Remove from Load Balancer
i-0509083213ab9e2c0	Node2	us-east-1e	InService	Remove from Load Balancer

Figure 2.2: Load balancer

The next step was to create a Auto Scaling Template.

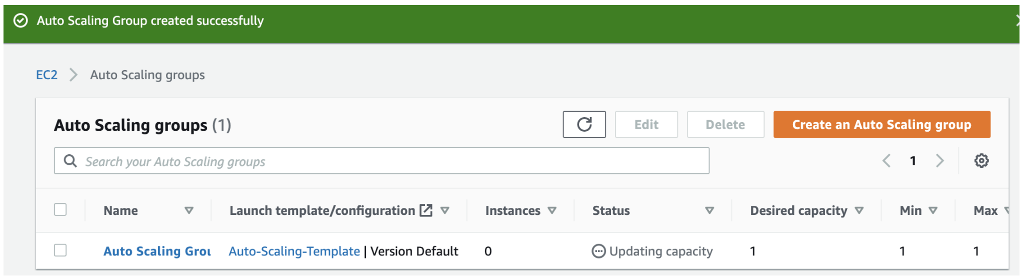


Success
Successfully created Auto-Scaling-Template (lt-09e13d60469a912ae)

▶ Actions log

Figure 2.3: Auto Scaling Template

Finally, we can create the Auto Scaling group.



Auto Scaling Group created successfully

EC2 > Auto Scaling groups

Auto Scaling groups (1)

	Name	Launch template/configuration	Instances	Status	Desired capacity	Min	Max
☐	Auto Scaling Grou	Auto-Scaling-Template | Version Default	0	Updating capacity	1	1	1

Figure 2.4: Auto Scaling Group

2) The used httperf command(s) to create load on the instances

httperf –server CC-Assignment-4-1305638622.us-east-1.elb.amazonaws.com –uri=/index.php –wsess=300,5,2 –rate 1 –timeout 5

3) Screenshots and link(s) to online video(s)* showing the CPU utilization across the instances over adequate time windows. You should annotate the graph(s) and/or videos with the corresponding events on their timelines (like start of httperf execution(s), instance creations, instance terminations, etc.).

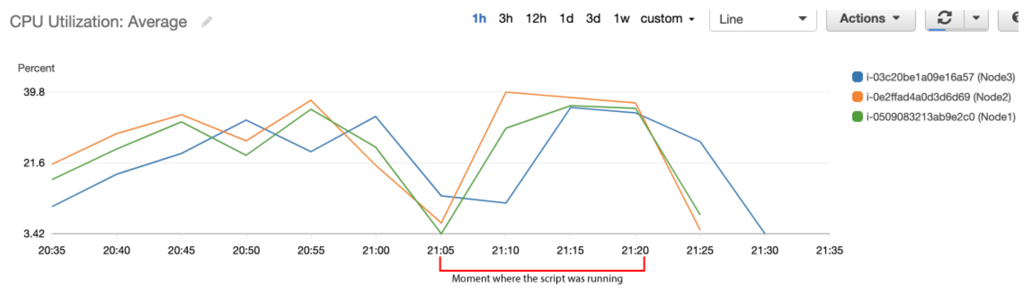


Figure 2.5: CPU Utilization

4) A detailed analysis of the obtained behavior and reactivity of your web service to load variations (increases and decreases of load). With the obtained behavior and reaction time, provide possible (concrete) web applications that could benefit from your auto-scaling mechanism for cloud resources.

When the httperf command is executed we can see a large increase of the load on one of the instances(Nodes), in this case, the Node 2. Since there is a high impact of the CPU Utilization on one of the nodes, and with the intend of stabilize and not overload the first node, the load balancer adds a second node (Node 1). After, at the peak of the CPU Utilization, the load balancer adds a third node (Node3). Finally, adding this third node the load starts to stabilize for a couple of minutes. Concluding the process, the httperf command ends and the load starts to decrease getting back to the normal CPU Utilization.
There are a couple of web applications that could benefit from this auto-scaling mechanism, mainly the applications where there is expected a large amount of request in a small amount of time. Examples of these web applications could be:

- A web site to sell tickets for movies
- A web site with sales
- Youtube, where the traffic in specific hours could have a high increase

Chapter 3

Conclusion

Once I have never worked with cloud computing, this assignment was pretty helpful to understand how the auto-scaling mechanism works and how it could be implemented in a web application. It was also satisfying seeing and analyzing the overload of the CPU when a lot of requests are made at the same period.