Exercise 3

Configuring a Load Balancer
Stress Testing

Prior Knowledge
Unix Command Line Shell
Exercise 2: Auto Scaling groups and Launch Configurations

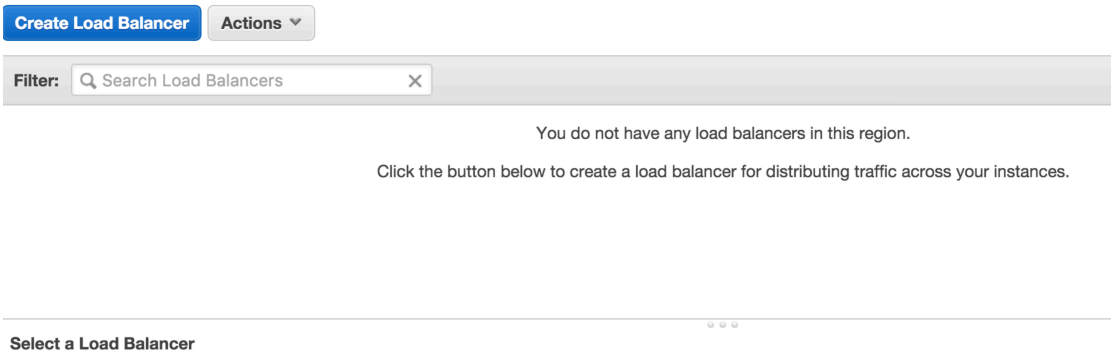
Learning Objectives
Creating an elastically scaled system in the cloud
How to stress test using Linux siege command

Software Requirements
Browser and AWS account, previous configuration from Exercise 2

Part A: Setting up a Load Balancer and ELB Auto Scale Group

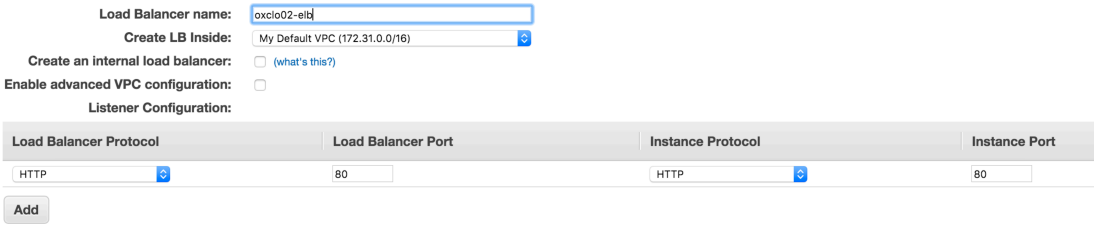
1. Go to the AWS Console and then the EC2 Console.

2. Near the bottom of the left hand menu, find Load Balancers and Click on it. You will see something like this (although other students may have created load balancers that will show up).



3. Click **Create Load Balancer**

4. In the screen following:
   a. Set name to *userid*-elb (e.g. oxclo02-elb)
   b. Leave the Load Balancer protocol as HTTP, etc, except change the **Instance Port** to 8080.
   This will mean that traffic coming to the LB will be sent to port 8080 on the instance servers.

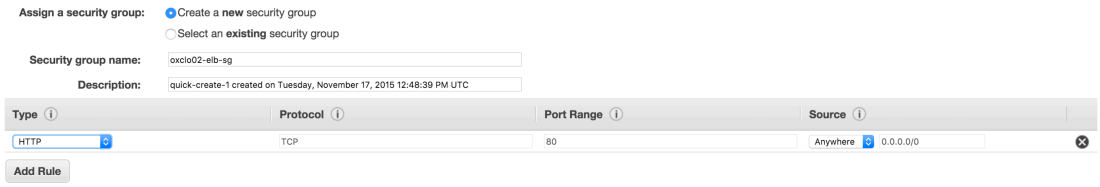


5. Click **Next: Assign Security Groups**

6. Select **Create a New Security Group**

7. Give it the name *userid*-elb-sg (e.g. oxclo02-elb-sg)

8. Make sure the rule says:
   HTTP TCP 80 Anywhere 0.0.0.0/0

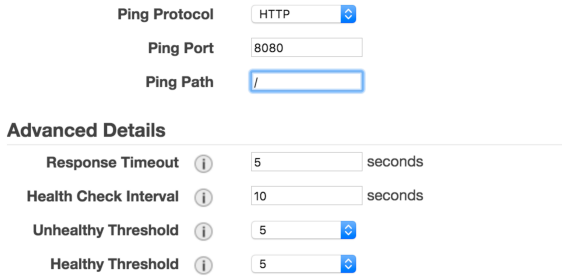


9. Click **Next: Configure Security Settings**

10. Ignore the warning and click: **Next: Configure Health Check**

11. Change the settings as follows:

    a. Ping Protocol: HTTP
    b. Ping Port: 8080
    c. Ping Path: /
    d. Response Timeout: 5
    e. Health Check interval: 10
    f. Unhealthy threshold: 5
    g. Healthy threshold: 5



12. Click **Next: Add EC2 Instances**

13. Do NOT add any instances! Click **Next: Add Tags**

14. Add the tag with Key/Value: Name / *userid*-asi

15. Click **Review and Create** then **Create**

16. Click **Close**

17. Now let's create our AutoScaling Group

18. Go back to creating an Auto Scale Group like last time. **(Auto Scaling Groups -> Create Auto Scaling Group)**

19. Create from an existing Launch Configuration and choose your own launch config that you previously created. Click **Next Step**

20. On the following screen:

    a. Give it a group name of *userid*-asg (e.g. oxclo02-asg)

    b. Add one or more subnets as before

    c. Expand the **Advanced Details**

    d. Click **Receive Traffic from Elastic Load Balancers**

e. Select your own Load Balancer from the options

f. Change the Health Check type to ELB

g. Leave the Grace period as 300 seconds

h. Click **Next: Configure Scaling Policies**

21. On the following screen

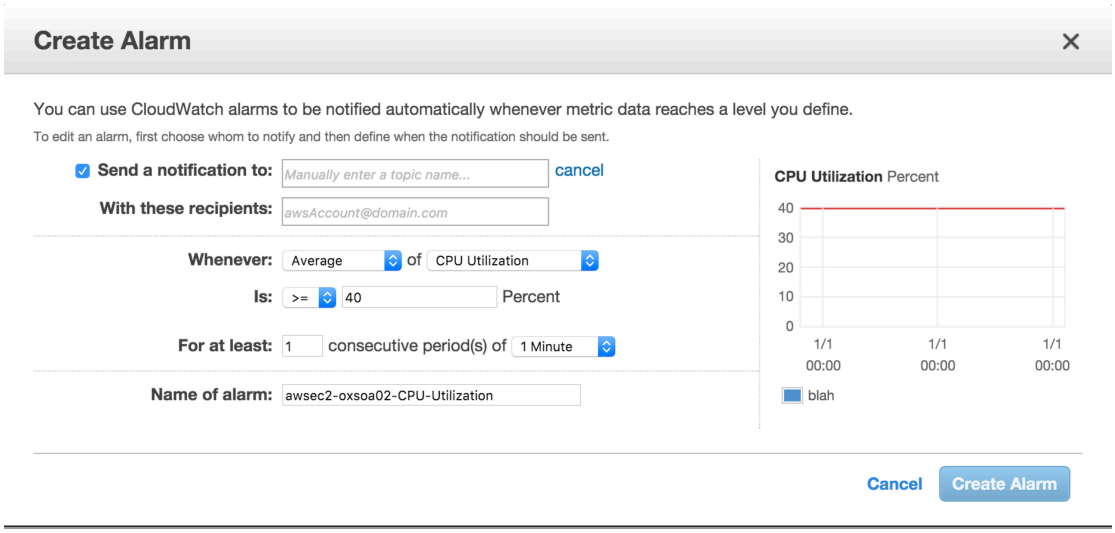
a. Select **Use scaling policies….**

b. Change it to support scaling between 1 and 4 instances

c. Click Add New Alarm

d. If you want notifications, choose your own topic that you defined before.

e. Change the Alarm to fire when the CPU utilization is >= 40% for more than 1 minute (we want to see scaling, so this is deliberately low)



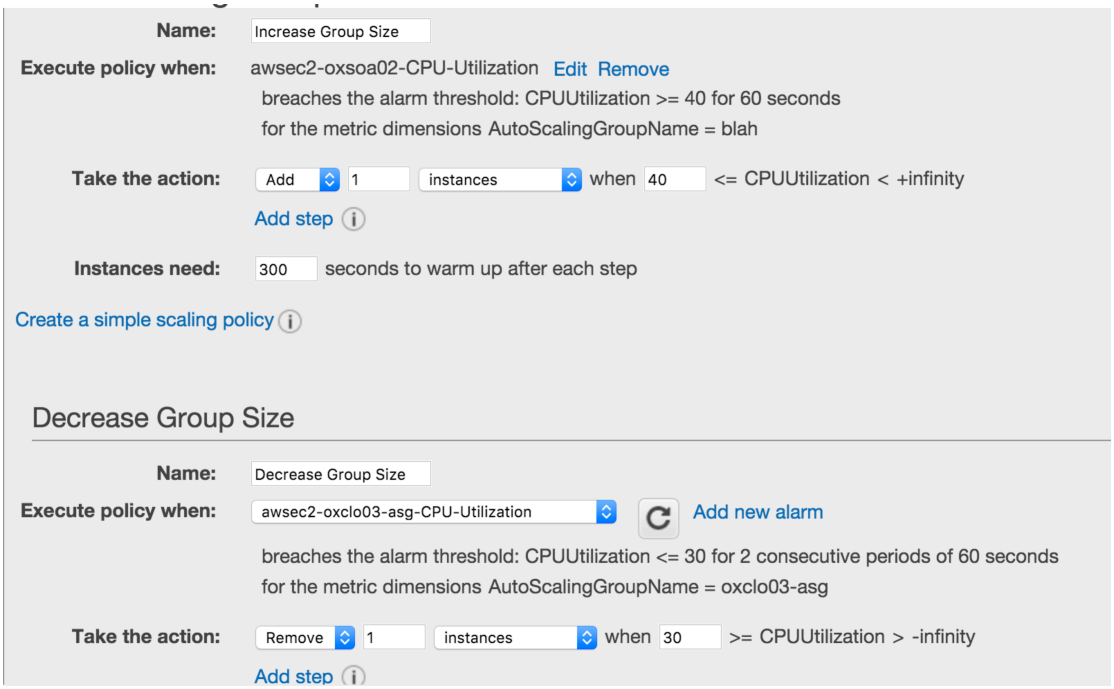
f. Click **Create Alarm**

22. Now update the rule to **Add 1 instance**

23. Set **Instances need 300 seconds to warm up after each step**

24. Create a similar Alarm for when CPU utilization is <= 30% for 2 minutes, and change the rule to Remove 1 instance.

25. It should look like:



26. Click **Next: Configure Notifications**

27. Click **Next: Configure Tags**

28. Add the tag: Name / *userid*-asi

29. Click **Review**

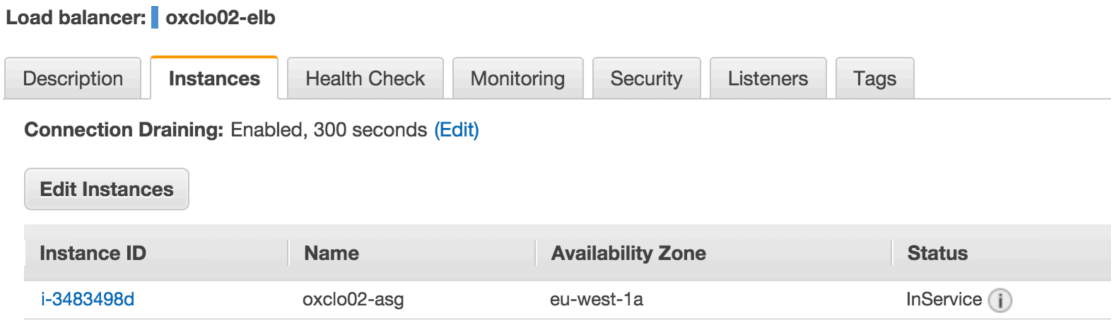
30. Click **Create Autoscaling Group**

31. Go and see if your instances are being started.

PART B – Stress testing

32. Navigate to view your ELB's dashboard page. You can find the DNS address of your ELB this way:

33. After the system has warmed up and your instance is running, it will eventually be tested by the ELB and become **In-Service.** You should see something like this:



34. Once you have an InService instance, copy and paste the DNS name into the address bar of your browser. You should see JSON returned from the node.js app.

Notice this is now available on port 80 and no longer using 8080.

35. We are going to create a new instance in the same subnet to stress test the servers from. We could do it from here, but we will take out network delays if we can do it within the Amazon EC2 network.

36. Using the EC2 Launch wizard like before, start a new instance with the following settings:

    a. **Ubuntu Server 14.04 LTS (HVM)**

    b. **t2.medium** (we want a beefier machine to be able to drive our nodes hard)

    c. User Data: (this is available in http://freo.me/oxclo-siege-ud)

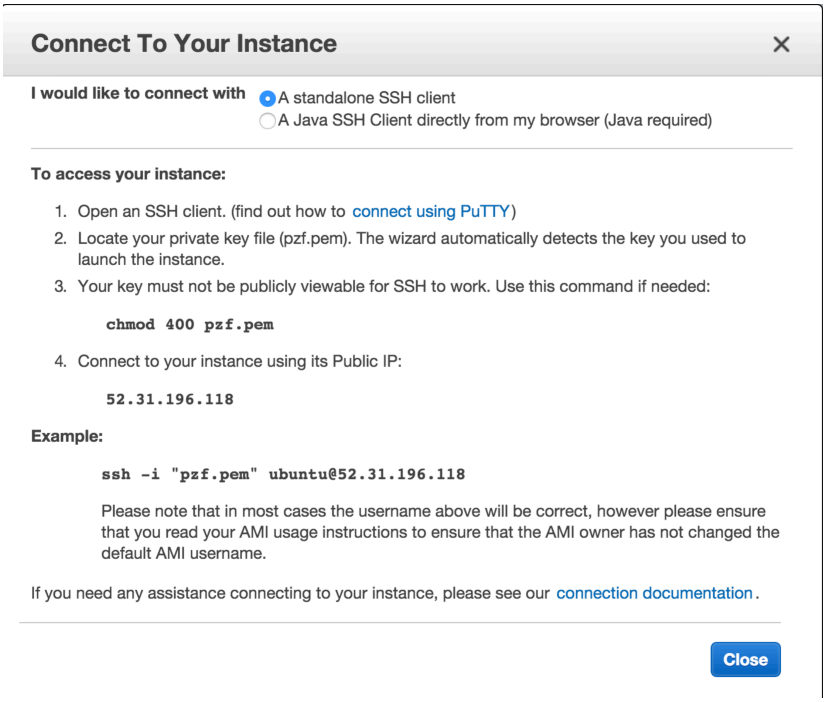
```
#!/bin/bash
# verbosity
set -e -x
# update the package list
apt-get update
# install node, node package manager and git.
apt-get -y install siege
# set more file descriptors
echo "* hard nofile 64000" >> /etc/security/limits.conf
echo "* soft nofile 64000" >> /etc/security/limits.conf
```

    d. Tag Name: *userid*-siege

    e. Security Group: node-security-group

    f. Your existing SSH Key

37. Once the instance is started, right-click on it and select "Connect" You will see a screen like:



38. You should be able to cut and paste the SSH line to your Terminal window and SSH into the server. Alternatively try out the built-in Java SSH client.

39. Accept the fingerprint as before.

40. In the SSH session type:

```
siege -c200 -t10m http://your-lb-dns-goes-here &

e.g

siege -c200 -t10m http://oxclo02-elb-137471382.eu-west-1.elb.amazonaws.com &
```

41. You should see something like:

```
** SIEGE 3.0.5
** Preparing 200 concurrent users for battle.
The server is now under siege...
```

42. This is basically hitting your Load Balancer with 200 concurrent clients for 10 minutes.

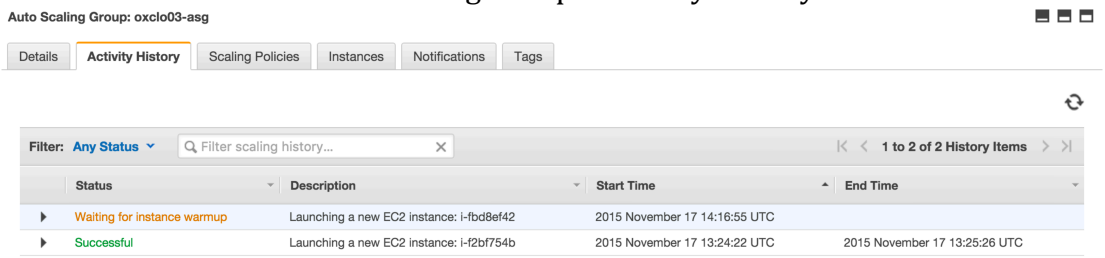
43. Unfortunately we are using a slightly old version of siege that has a few bugs, and it doesn't support high concurrency without crashing. We need to ramp up the stress on the cluster to cause it to scale up. This is why we added the & to the end of the line. That causes siege to run in the background. Simply pushing the up arrow will retrieve the same command line and we can start another siege.

    Start 3 or 4 this way.

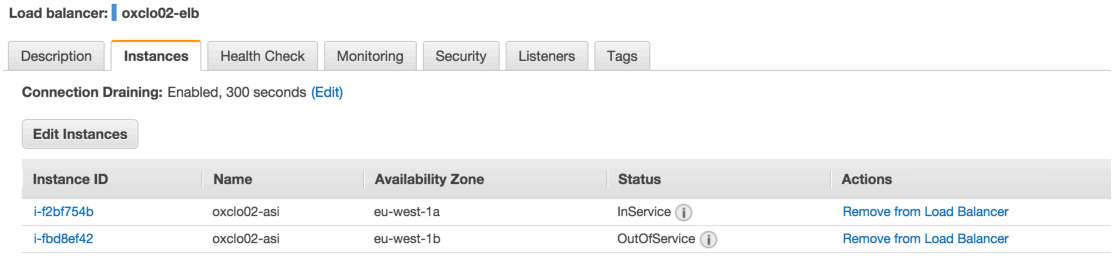
44. Unless we run out of network bandwidth, this should push the instances's average CPU above 40% and cause the Scaling Group to start another server.

45. Assuming all is well you should see a new instance spawned shortly.

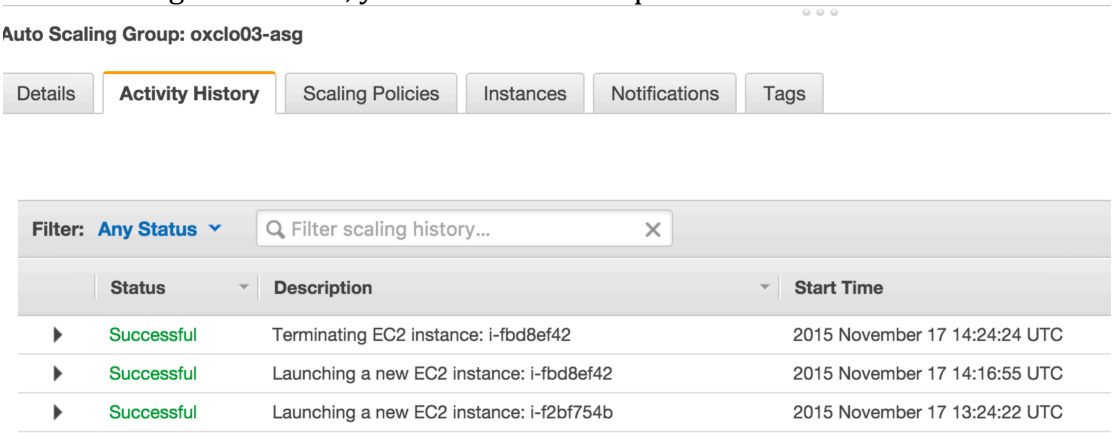
46. You can also check the Auto Scaling Group's Activity History



47. And the Elastic Load Balancer's instances



48. Once the siege has ended, you should see the spare instance removed:



49. Once you have finished, **delete** the autoscaling group and **terminate** the siege instance. Make sure that you have no further instances running in your name!

50. You have completed the exercise. Well done.

51. As an **extension**, come up with a plan to secure the cloud instances better through improved configuration of the security groups. Identify which systems need to talk to which, and then suggest a set of security groups that would allow this.