Exercise 3

Configuring a Load Balancer
Stress Testing

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

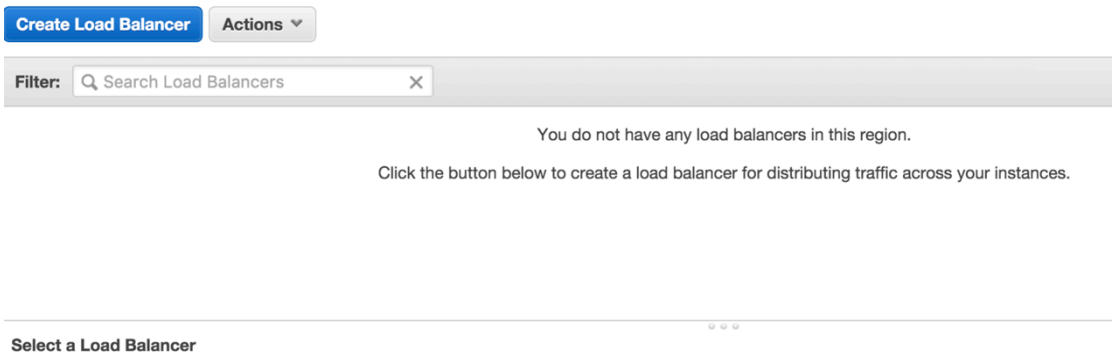
Learning Objectives
Creating an elastically scaled system in the cloud
How to stress test using Linux siege command

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

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

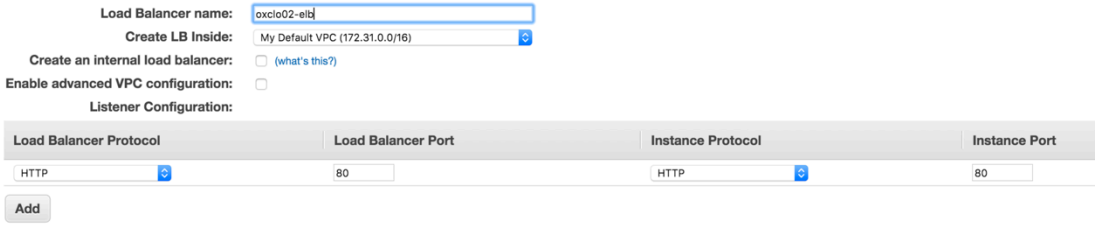
1. Go to the AWS Console and then the EC2 Console.

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. Choose **Classic Load Balancer**

5. 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.

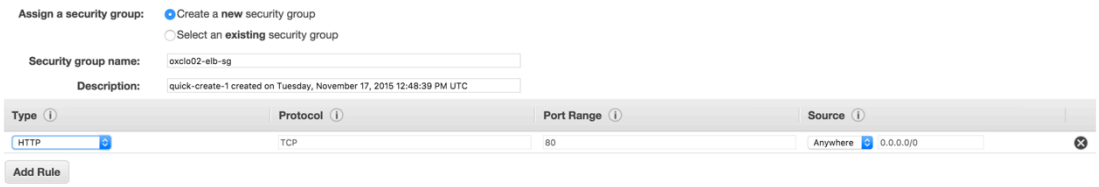


6. Click **Next: Assign Security Groups**

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

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

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

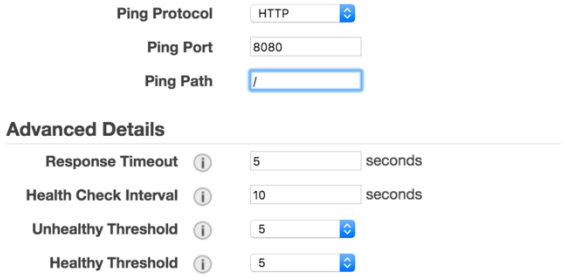


10. Click **Next: Configure Security Settings**

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

12. Change the settings as follows:

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



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

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

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

16. Click **Review and Create** then **Create**

17. Click **Close**

18. Now let's create our AutoScaling Group

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

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

21. 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**

22. On the following screen

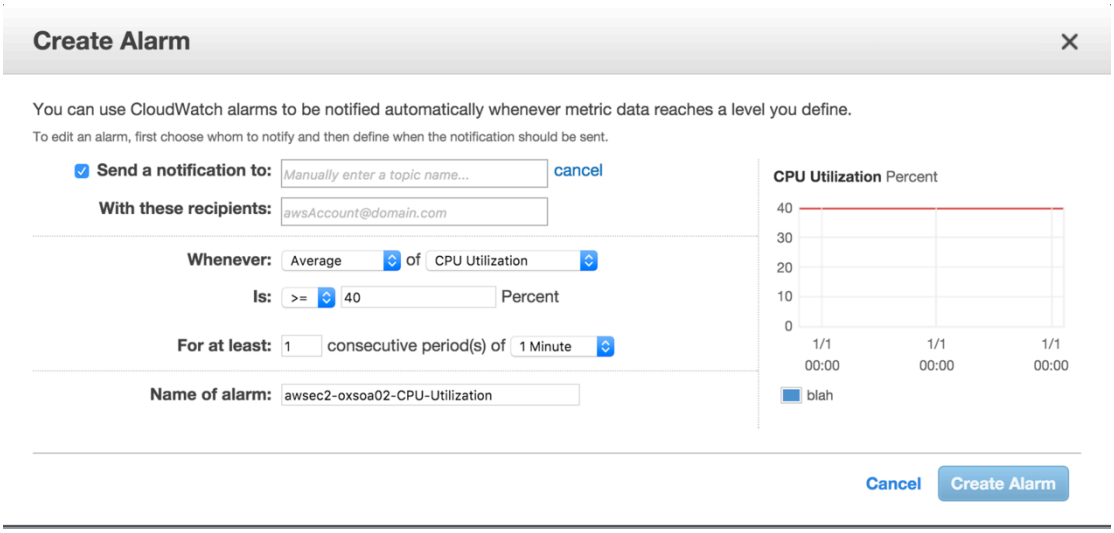
a. Select **Use scaling policies….**

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 >= 35% for more than 1 minute (we want to see scaling, so this is deliberately low). Note that the picture below shows 40%, but I recommend using 35% to ensure that we cause enough work to see scaling.



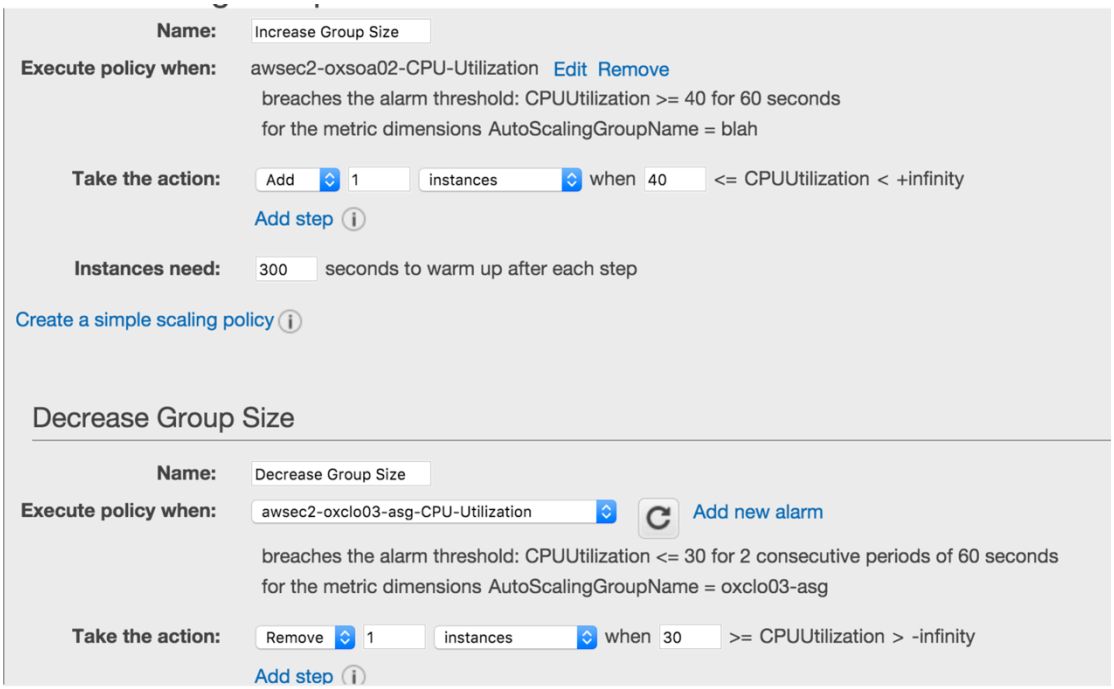
f. Click **Create Alarm**

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

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

25. Create a similar Alarm for when CPU utilization is <= 25% for 2 minutes, and change the rule to Remove 1 instance.

26. It should look like (again, note that the picture shows different %ages)



27. Click **Next: Configure Notifications**

28. Click **Next: Configure Tags**

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

30. Click **Review**

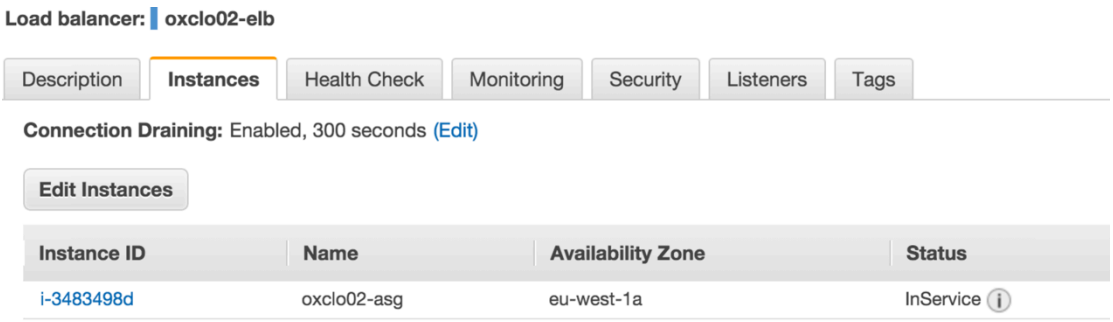
31. Click **Create Autoscaling Group**

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

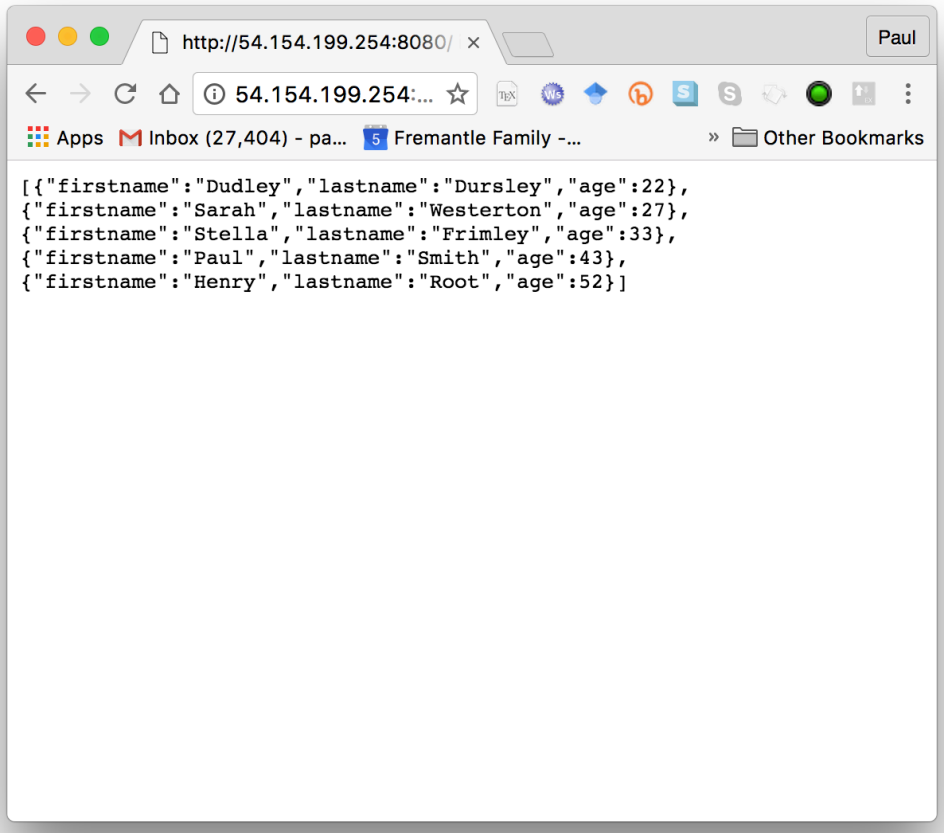
PART B – Stress testing

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

34. 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:



35. 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, because the load balancer listens on 80.

36. 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.

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

    a. **Ubuntu Server 16.04 LTS (HVM)**

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

    c. User Data: please cut and paste from http://freo.me/oxclo-siege-ud

    d. This simply installs the latest version of *siege* and sets correct parameters for the OS to handle this. (the version in the Ubuntu repo is out of date and buggy unfortunately). Later we will see how Docker would solve this problem better….

    e. Tag Name: *userid*-siege

    f. Security Group: node-security-group

    g. Your existing SSH Key

38. Check the instance is running in the EC2 dashboard and then SSH into the instance as in Exercise 1.

39. Accept the fingerprint as before.

40. In the SSH session type:

```
siege -c100 -t15m http://your-lb-dns-goes-here

e.g

siege -c100 -t15m http://oxclo02-elb-137471382.eu-west-1.elb.amazonaws.com
```

41. You should see something like:

```
New configuration template added to /home/ubuntu/.siege
Run siege -C to view the current settings in that file
[alert] Zip encoding disabled; siege requires zlib support to
enable it: No such file or directory
** SIEGE 4.0.3rc4
** Preparing 100 concurrent users for battle.
The server is now under siege...
```

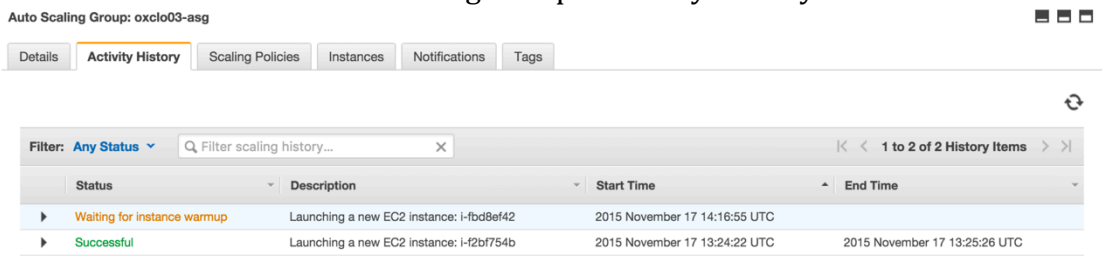
42. This is basically hitting your Load Balancer with 100 concurrent clients for 15 minutes. This should be long enough to see the behaviour we want.

43. You may also see messages like:
    `[error] A temporary resolution error for oxclo01-elb-2104065837.eu-west-1.elb.amazonaws.com`
    You can ignore these.

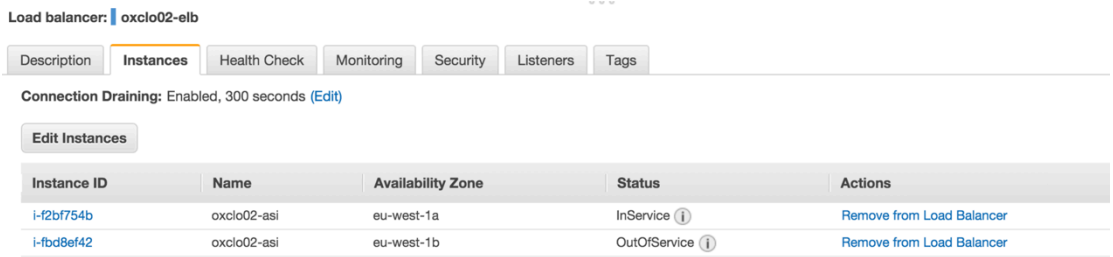
44. Unless we run out of network bandwidth, this should push the instances's average CPU above 40% and cause the Scaling Group to start another server.

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



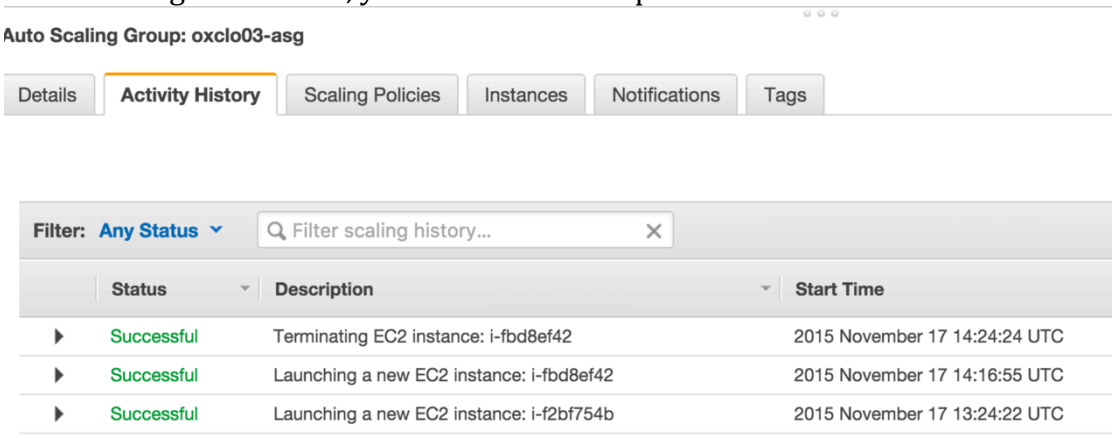
47. And the Elastic Load Balancer's instances



48. Once you have seen the scaling, you can end the siege if you like, by hitting Ctrl-C in the command line window:

```
Lifting the server siege...
Transactions:                 613905 hits
Availability:                  99.96 %
Elapsed time:                 599.40 secs
Data transferred:         157.49 MB
Response time:                  0.09 secs
Transaction rate:       1024.20 trans/sec
Throughput:                0.26 MB/sec
Concurrency:                   95.61
Successful transactions:      613905
Failed transactions:               3
Longest transaction:            6.01
Shortest transaction:           0.00
```

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



50. Once you have finished:
    a. Delete the autoscaling group
    b. Delete the load balancer
    c. Terminate the siege instance.
    d. Make sure that you have no further instances running in your name!

51. You have completed the exercise. Well done.

52. 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.