CMPE 281 - LAB #2 - Elastic Load Balancer

In this Lab, you will be creating a small three instance auto-scaled cluster using the AWS Linux AMI from your previous Lab. You will then configure an AWS Elastic Load Balancer and create load on instances to observe Cloud Elasticity at work. You will then perform equivalent steps on Google Cloud Platform.

Part 1 - Amazon Web Services - Elastic Load Balancer

Lab Documents:

- http://docs.aws.amazon.com/autoscaling/latest/userguide/as-register-lbs-with-asg.html
- http://docs.aws.amazon.com/autoscaling/latest/userguide/autoscaling-load-balancer.html
- http://docs.aws.amazon.com/elasticloadbalancing/latest/classic/elb-create-https-ssl-load-balancer.html
- http://docs.aws.amazon.com/autoscaling/latest/userguide/as-add-availability-zone.html#as-add-az-console
- http://docs.aws.amazon.com/elasticloadbalancing/latest/application/tutorial-load-balancer-routing.html
- http://docs.aws.amazon.com/elasticloadbalancing/latest/application/tutorial-target-ecs-containers.html

Lab Source Files:

- https://github.com/paulnguyen/cmpe281/tree/master/labs/lab2

Note: If you did not already in the previous Lab, make sure to add a "index.html" file in your Apache Web Root. The contents of "index.html" should be:

<h1>Health Check Test Page</h1>
Key Steps Are:

1. Create or Select a Launch Configuration
2. Create an Auto Scaling Group
3. Using a Load Balancer With an Auto Scaling Group

CREATE LAUNCH CONFIG AND AUTOSCALE GROUP

Tutorial: Set Up a Scaled and Load-Balanced Application

DOC: http://docs.aws.amazon.com/autoscaling/latest/userguide/as-register-lbs-with-asg.html

Create or Select a Launch Configuration

Select My AMI:	**cmpe281-ami**
Instance Type:	T2-Micro (Free Tier)
Launch Configuration Name:	aws-php-autoscale
Enable Monitoring:	Enable CloudWatch detailed monitoring
Select Public IP:	Assign a public IP address to every instance.
Security Group:	cmpe281-dmz (SG)
Select Key Pair:	cmpe281-us-west-2 (or cmpe281-us-east-1)
Select VPC:	cmpe281 (VPC) & Public Subnet

Create an Auto Scaling Group

Create Auto Scale Group:	aws-php-autoscale
Group Size (Starts with):	1
Network:	cmpe281 (VPC) | Public Subnet

Use scaling policies to adjust the capacity of this group

Scale between:	1 - 3 instances
Increase when:	AVG CPU >= 40% (for at lease 1 minute)
Decrease when:	AVG CPU <= 15% (for at lease 1 minute)

CREATE CLASSIC LOAD BALANCER

Using a Load Balancer With an Auto Scaling Group

DOC: http://docs.aws.amazon.com/autoscaling/latest/userguide/autoscaling-load-balancer.html

Create ELB (Classic Load Balancer)

Name:	aws-php-elb-classic
VPC:	cmpe281 (select public subnet)
SG:	cmpe281-dmz
Port:	80
Health Check:	Default path, Unhealthy Checks: 2, Healthy Checks: 4
Add Instances:	Select running instance (from aws-php-autoscale)
Edit Auto Scale Group:	aws-php-autoscale
Select ELB:	aws-php-elb-classic

Expanding Your Scaled and Load-Balanced Application to an Additional Availability Zone

DOC: http://docs.aws.amazon.com/autoscaling/latest/userguide/as-add-availability-zone.html#as-add-az-console

```
Select Auto Scale Group:        aws-php-autoscale
Select Edit / Details / AZs:    Select two Public Subnets (in us-west-1a and us-west-1b)
Set the Desired and Min to:     two instances
```

CREATE APPLICATION LOAD BALANCER

http://docs.aws.amazon.com/elasticloadbalancing/latest/application/tutorial-load-balancer-routing.html

Create Target Groups:

aws-linux-1 Port 80 Protocol HTTP (Register one or more Instances)
- Name: aws-linux-1
- Type: instance
- Protocol: http
- Port: 80
- VPC: cmpe281
- Health Check: / (HTTP)
- Instances: Add Instances to Target Group

aws-linux-2 Port 80 Protocol HTTP (Register one or more Instances)
- Name: aws-linux-2
- Type: instance
- Protocol: http
- Port: 80
- VPC: cmpe281
- Health Check: / (HTTP)
- Instances: Add Instances to Target Group

Create Application Load Balancer:

```
Name:               aws-php-elb-app
Internet Facing:    true
VPC:                cmpe281
AZ's:               AZ's in your region
                    (us-west-2 or us-east-1)
Listener: HTTP (Port 80)
Security Group: cmpe281-dmz
                Path: /loadtest.php  Target: aws-linux-1
                Path: /fibonacci.php Target: aws-linux-2
                (Default)            Target: aws-linux-1
```

Part 2 - Autoscaling VM Instances with Google Cloud Global Load Balancer

Lab Documents:

- https://cloud.google.com/compute/docs/instance-templates/create-instance-templates
- https://cloud.google.com/compute/docs/instance-groups/creating-groups-of-managed-instances
- https://cloud.google.com/compute/docs/instance-groups/adding-an-instance-group-to-a-load-balancer
- https://cloud.google.com/compute/docs/instance-groups/autohealing-instances-in-migs
- https://cloud.google.com/load-balancing/docs
- https://cloud.google.com/compute/docs/autoscaler
- https://cloud.google.com/compute/docs/autoscaler/scaling-cpu-load-balancing
- https://cloud.google.com/compute/docs/tutorials/high-availability-load-balancing
- https://cloud.google.com/compute/docs/autoscaler/scaling-stackdriver-monitoring-metrics

Lab Source Files:

- https://github.com/paulnguyen/cmpe281/tree/master/labs/lab2

CREATE INSTANCE TEMPLATE

1. In the Cloud Console / Compute Engine, go to the Instance templates page.
2. Create instance template:

Name:	php-template
Machine Family:	N1
Machine Type:	f1-micro
Boot Disk:	php-image (custom image)
Identity and API access:	default settings
Firewall:	Allow HTTP traffic

CREATE MANAGED INSTANCE GROUP

1. In the Cloud Console / Compute Engine, go to the Instance Groups page.
2. Create Instance Group:

Name:	php-group
Location:	multi-zone Region: us-west1 Zones: us-west1-a, us-west1-b, us-west1-c
Instance Template:	php-template
Auto Scaling:	CPU Utilization: 40% Cool Down Period: 60 seconds # of Instances: 1 (min) - 3 (max)
Health Check:	TCP (Port 80) Initial Delay: 300 seconds

CONFIGURE HEALTH CHECK FIREWALL RULE

1. In the Google Cloud Console / VPC Network, go to the Create a firewall rule page.
2. Create Firewall Rule:
   - Name: allow-health-check
   - Network: cmpe281
   - Targets: all instances in network
   - Source filter: 0.0.0.0/0
   - Protocols and Ports: tcp:80

USING A LOAD BALANCER WITH A MANAGED INSTANCE GROUP

1. In the Google Cloud Console / Network Services, go to the Create Load Balancer page.

2. Create Load Balancer:
   - Name: php-load-balancer
   - HTTP(S) Load Balancing
   - From Internet to my VMs

3. Configure Load Balancer:

   Backend:
   - Backend Service Name: php-backend
   - Backend Type: instance group
   - Select Instance Group: php-group
     - Port: 80
     - Balancing Mode: Utilization (Max 80)
     - Capacity: 10
     - Health Check: select healthcheck (tcp)

   Frontend:
   - Front End Name: php-frontend
   - Protocol: http
   - Network Service Tier: standard
   - Region: (same as backend)
   - IP Version: IPv4
   - IP Address: Ephemeral
   - Port: 80