**Create a New Role**

The access to the different services can be controlled by defining a new Policy. To do that need to do the following:

1. Select IAM under services
2. Click on Policies
3. Click on Create policy
4. Click on Choose a service
5. Enter EC2 and click on the EC2 service below
6. Click on Select actions
7. In the filter box enter DescribeTag and place a check mark to add the action
8. Click on Add additional permissions
9. Click on Choose a service
10. Enter CloudWatch and click on the CloudWatch service below
11. Click on Select actions
12. In the filter box enter PutMetricData and place a check mark to add the action
13. In the filter box enter GetMetricStatistics and place a check mark to add the action
14. In the filter box enter ListMetrics and place a check mark to add the action
15. Make sure that under CloudWatch section you have (3 actions) and under EC2 you have (2 actions)
16. Give it a meaningful name and description and click on Create policy

**Create Launch Configuration**

1. Select EC2
2. Click on Launch Configurations
3. Create launch configuration
4. Select My AMIs and select desired image
5. Select type and click on Next: Configure Details
6. Enter the name of the launch configuration
7. Select the IAM role created before
8. Select “Enable CloudWatch detailed monitoring”
9. Click on Advanced Details
10. Under User data enter the script that will send metrics to CloudWatch
    1. Example:

#!/bin/sh

cd /home/ubuntu

rm -f aws-mon-asg.sh

rm -f custommetrics.txt

wget http://ri2017-gpswks404.s3.amazonaws.com/aws-mon-asg.sh

chown ubuntu:ubuntu aws-mon-asg.sh

chmod 755 aws-mon-asg.sh

apt-get install awscli -y

echo "\*/1 \* \* \* \* ~ubuntu/aws-mon-asg.sh --all-items --disk-path=/ --from-cron" >> custommetrics.txt

sudo -u ubuntu crontab custommetrics.txt

apt-get install stress -y

stress --cpu 20 --io 4 --vm 2 --vm-bytes 128M --timeout 1000s &

1. Next: Add Storage
2. Set the desired storage capacity
3. Next: Configure Security Group
4. Select the desired security group and click on Review
5. Click on Create launch configuration
6. Select the key pair to use

**Create an Auto Scaling Group**

1. Click on Services and Select EC2 under Compute
2. Click on Auto Scaling Groups
3. Click on Create Auto Scaling group
4. Make sure “Create an Auto Scaling group from an existing launch configuration” is selected
5. Select the Launch Configuration created above
6. Click on Next Step
7. Give it a meaningful group name
8. Set the start with to 1
9. Set the VPC and subnet as desired
10. Click on Advanced Details
11. Set the Health Check Grace Period to 240
12. Make sure the Enable CloudWatch detailed monitoring” is enabled
13. Click on Next: Configure scaling policies
14. Select “Keep this group as its initial size”
15. Click on Next: Configure Notifications
16. Click on Next: Configure Tags
17. Add desired tags such as Key = Name and Value = MyConfgiGroup
18. Click on Review
19. Verify everything is OK and click on Create Auto Scaling group

**Create Alarms**

**Create a Scaling Up Alarm**

1. Open the CloudWatch dashboard
2. Click on Alarms
3. Click on Create Alarm
4. Under Custom Metrics select “System/Detail/Linux” as the Namespace
5. Select the Custom Metric. For this example I used LoadAverage1Min
6. Click on Next
7. Give the Alarm a meaningful name
8. Set the “Whenever: LoadAverage1Min” to >= 10
9. Set the datapoints to 2
10. Change the Period to 1
11. Delete the default Notification Action
12. Click on Create Alarm

**Create a Scaling Down Alarm**

1. Open the CloudWatch dashboard
2. Click on Alarms
3. Click on Create Alarm
4. Under Custom Metrics select “System/Detail/Linux” as the Namespace
5. Select the Custom Metric. For this example I used LoadAverage1Min
6. Click on Next
7. Give the Alarm a meaningful name
8. Set the “Whenever: LoadAverage1Min” to <= 15
9. Set the datapoints to 2
10. Change the Period to 5
11. Delete the default Notification Action
12. Click on Create Alarm

**Add Policies**

**Adding Scaling UP Policy**

1. Click on AutoScaling Groups
2. Select the created Auto Scaling Group
3. Select the Scaling Policies tab
4. Click on Add policy
5. Click on “Create a simple scaling policy”
6. Give it a meaningful name
7. under “execute policy when:” select the Alarm created previously for scaling up
8. Set the “Take the action:” to Add 1 instance
9. Set the “And then wait:” 300 seconds before allowing another scaling activity
10. Click on Create

**Adding Scaling Down Policy**

1. Click on AutoScaling Groups
2. Select the created Auto Scaling Group
3. Select the Scaling Policies tab
4. Click on Add policy
5. Click on “Create a simple scaling policy”
6. Give it a meaningful name
7. under “execute policy when:” select the Alarm created previously for scaling down
8. Set the “Take the action:” to Remove 1 instance
9. Set the “And then wait:” 300 seconds before allowing another scaling activity
10. Click on Create

**Another way is Using the AWS Command Line Interface (awscli)**

**Create Launch Configuration**

aws autoscaling create-launch-configuration\

--launch-configuration-name my-custom-asg-config\

--associate-public-ip-address\

--instance-monitoring Enabled=true\

--image-id ami-1853ac65\

--instance-type t2.micro\

--security-groups sg-54410d2f\

--key aws\_serv\_server\_key\

--iam-instance-profile CcdpCustomMetricsAS\

--user-data file:///nishome/oegante/metrics.sh

**Create Auto Scaling Group**

aws autoscaling create-auto-scaling-group\

--auto-scaling-group-name my-custom-asg\

--launch-configuration my-custom-asg-config\

--min-size 1\

--max-size 3\

--default-cooldown 180\

--desired-capacity 1\

--health-check-grace-period 240\

--tag "Key=Name,Value=my-custom-asg,PropagateAtLaunch=true"\

--vpc-zone-identifier subnet-d7008b8f\

--availability-zones us-east-1d

**Create Up Auto Scaling Policy**

aws autoscaling put-scaling-policy\

--policy-name LoadHighPolicy\

--auto-scaling-group-name my-custom-asg\

--scaling-adjustment=1\

--adjustment-type ChangeInCapacity

STORE THE ARN FROM ABOVE IS NEEDED BELOW IS SHOULD LOOK LIKE:

{

"Alarms": [],

"PolicyARN": "arn:aws:autoscaling:us-east-1:451796069025:scalingPolicy:98a4d453-87f7-4a97-a7df-e425a9369873:autoScalingGroupName/my-custom-asg:policyName/LoadHighPolicy"

}

YOU NEED EVERYTHING AFTER PolicyARN

**Create Up Auto Scaling Alarm**

aws cloudwatch put-metric-alarm\

--alarm-name HighLoadAlarm\

--comparison-operator GreaterThanThreshold\

--evaluation-periods 1\

--metric-name "LoadAverage1Min"\

--namespace "System/Detail/Linux"\

--period 60\

--statistic Average\

--threshold 10\

--dimensions Name=AutoScalingGroupName,Value=my-custom-asg\

--alarm-actions < THE ARN FROM ABOVE!!!! >

**Create Scaling Down Policy**

aws autoscaling put-scaling-policy\

--policy-name LoadLowPolicy\

--auto-scaling-group-name my-custom-asg\

--scaling-adjustment=-1\

--adjustment-type ChangeInCapacity

STORE THE ARN FROM ABOVE IS NEEDED BELOW IS SHOULD LOOK LIKE:

{

"Alarms": [],

"PolicyARN": "arn:aws:autoscaling:us-east-1:451796069025:scalingPolicy:284d2036-b5e7-4871-8143-9cf12826a6af:autoScalingGroupName/my-custom-asg:policyName/LoadLowPolicy"

}

**Create Down Auto Scaling Alarm**

aws cloudwatch put-metric-alarm\

--alarm-name LowTCPAlarm\

--comparison-operator LessThanThreshold\

--evaluation-periods 1\

--metric-name "LoadAverage1Min"\

--namespace "System/Detail/Linux"\

--period 60\

--statistic Average\

--threshold 15\

--dimensions Name=AutoScalingGroupName,Value=my-custom-asg\

--alarm-actions < THE ARN FROM ABOVE!!!! >