

Auto scaling group-Launch Configuration

Prepared by Sriganesh Pera

Launch Configurations:

A launch configuration is a template that an Auto Scaling group uses to launch EC2 instances.

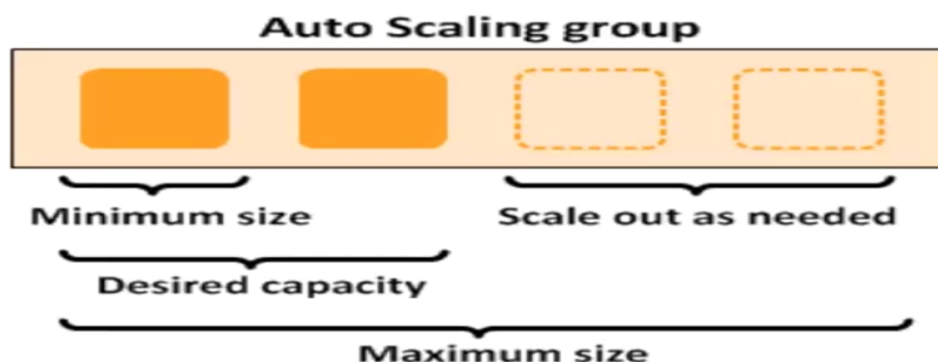
When you create an Auto Scaling group, you must specify a launch configuration. You can specify your launch configuration with multiple Auto Scaling groups. However, you can only specify one launch configuration for an Auto Scaling group at a time, and you can't modify a launch configuration after you've created it.

Therefore, if you want to change the launch configuration for your Auto Scaling group, you must create a launch configuration and then update your Auto Scaling group with the new launch configuration.

Auto Scaling:

Auto Scaling helps you maintain application availability and allows you to scale your Amazon EC2 capacity up or down automatically per conditions you define.

Auto Scaling can also automatically increase the number of Amazon EC2 instances during demand spikes to maintain performance and decrease capacity during lulls to reduce costs



Example:

My application to be live	– Minimum 1 server
My application to perform normal	– Desired 3 servers
My application to perform during peak season	– Maximum 20 servers
During peak load – Scale out by	- 2 Servers every 10 minutes

1. Launch Configuration (Template)

- AMI (Ex: Amazon Linux, Redhat Linux)
- Instance type (Ex: t2.micro, m4.large)
- Storage
- Security Group
- SSH-Key pair

2. Autoscaling Group

- Launch Configuration
- Network/Subnet's
- Scaling policies for increase
- Scaling policies for decrease
- Monitoring & Alarm

Scenario:

Auto-scaling test scenario

- Build webserver
- Create AMI (image)
- Create Launch configuration using webserver AMI
- Create Autoscale Group with minimum 1 instance, increase by 1 & maximum 4.
- Run the following script to Increase CPU load above 50%

[illegible]

dd if=/dev/zero of=/dev/null bs=50000 count=100000

Lab :Launching an EC2 instance with auto scaling group and notice how AWS automatically increase EC2 instance when the load is up and decrease EC2 instances when load is down

Step 1: Login to AWS console and create an EC2 instance and install apache webserver

As you know how to create an EC2 instance we straight away connect the instance and install Apache webserver.

Type in command prompt:

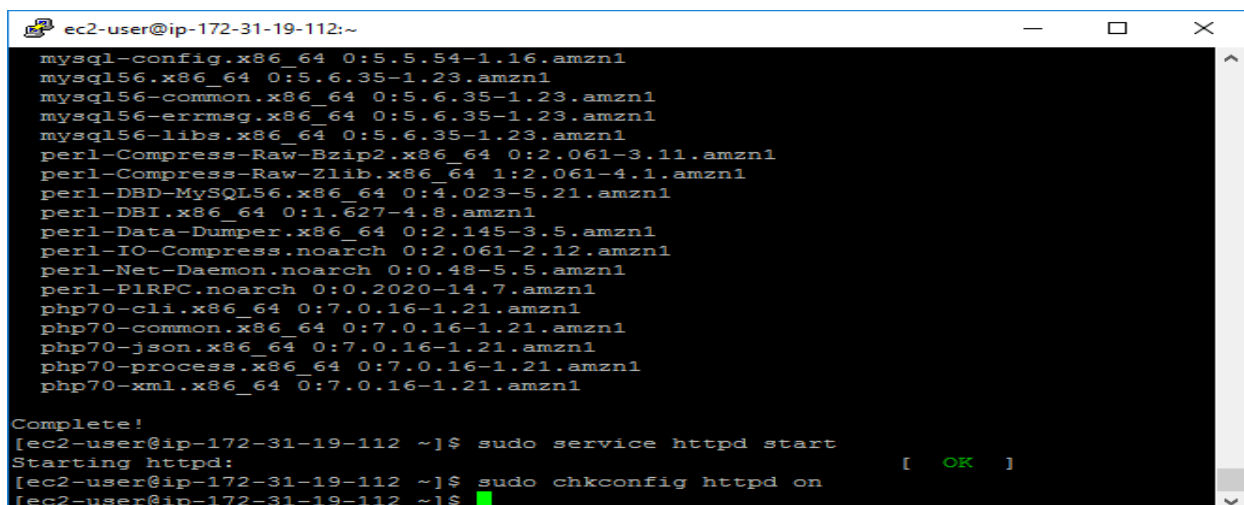
sudo yum update -y

sudo yum install -y httpd24 php70 mysql56-server php70-mysqld

sudo service httpd start

sudo chkconfig httpd on

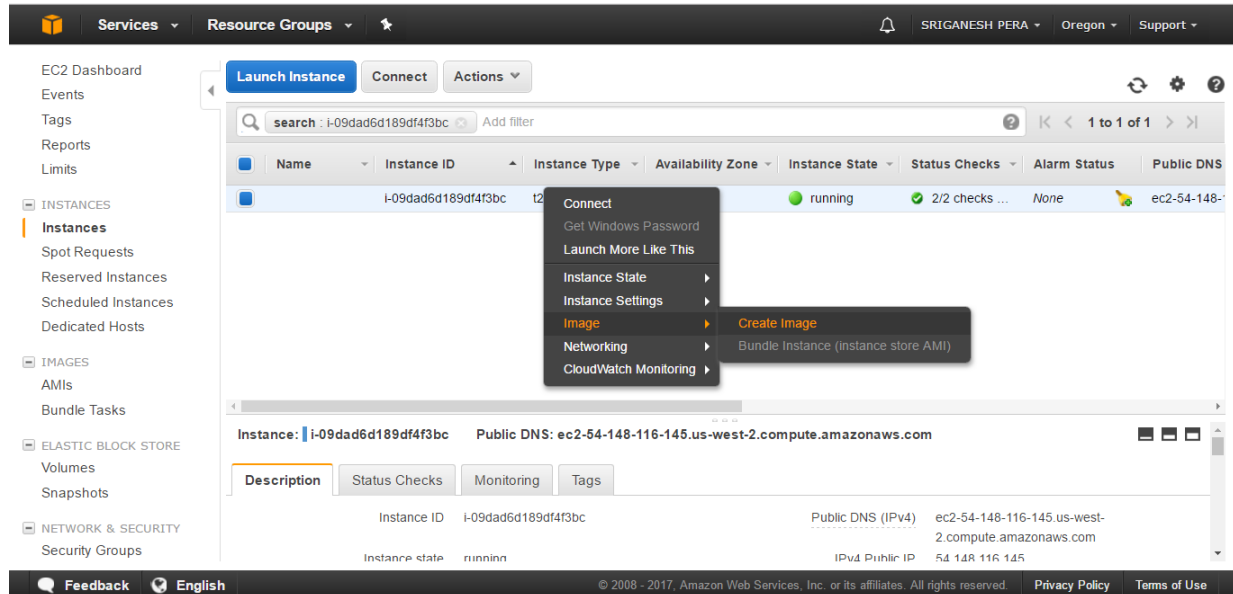
Now we have installed Apache webserver on our instance.



```
ec2-user@ip-172-31-19-112:~  
mysql-config.x86_64 0:5.5.54-1.16.amzn1  
mysql56.x86_64 0:5.6.35-1.23.amzn1  
mysql56-common.x86_64 0:5.6.35-1.23.amzn1  
mysql56-errmsg.x86_64 0:5.6.35-1.23.amzn1  
mysql56-libs.x86_64 0:5.6.35-1.23.amzn1  
perl-Compress-Raw-Bzip2.x86_64 0:2.061-3.11.amzn1  
perl-Compress-Raw-Zlib.x86_64 1:2.061-4.1.amzn1  
perl-DBD-MySQL56.x86_64 0:4.023-5.21.amzn1  
perl-DBI.x86_64 0:1.627-4.8.amzn1  
perl-Data-Dumper.x86_64 0:2.145-3.5.amzn1  
perl-IO-Compress.noarch 0:2.061-2.12.amzn1  
perl-Net-Daemon.noarch 0:0.48-5.5.amzn1  
perl-PlRPC.noarch 0:0.2020-14.7.amzn1  
php70-cli.x86_64 0:7.0.16-1.21.amzn1  
php70-common.x86_64 0:7.0.16-1.21.amzn1  
php70-json.x86_64 0:7.0.16-1.21.amzn1  
php70-process.x86_64 0:7.0.16-1.21.amzn1  
php70-xml.x86_64 0:7.0.16-1.21.amzn1  
Complete!  
[ec2-user@ip-172-31-19-112 ~]$ sudo service httpd start  
Starting httpd: [ OK ]  
[ec2-user@ip-172-31-19-112 ~]$ sudo chkconfig httpd on  
[ec2-user@ip-172-31-19-112 ~]$
```

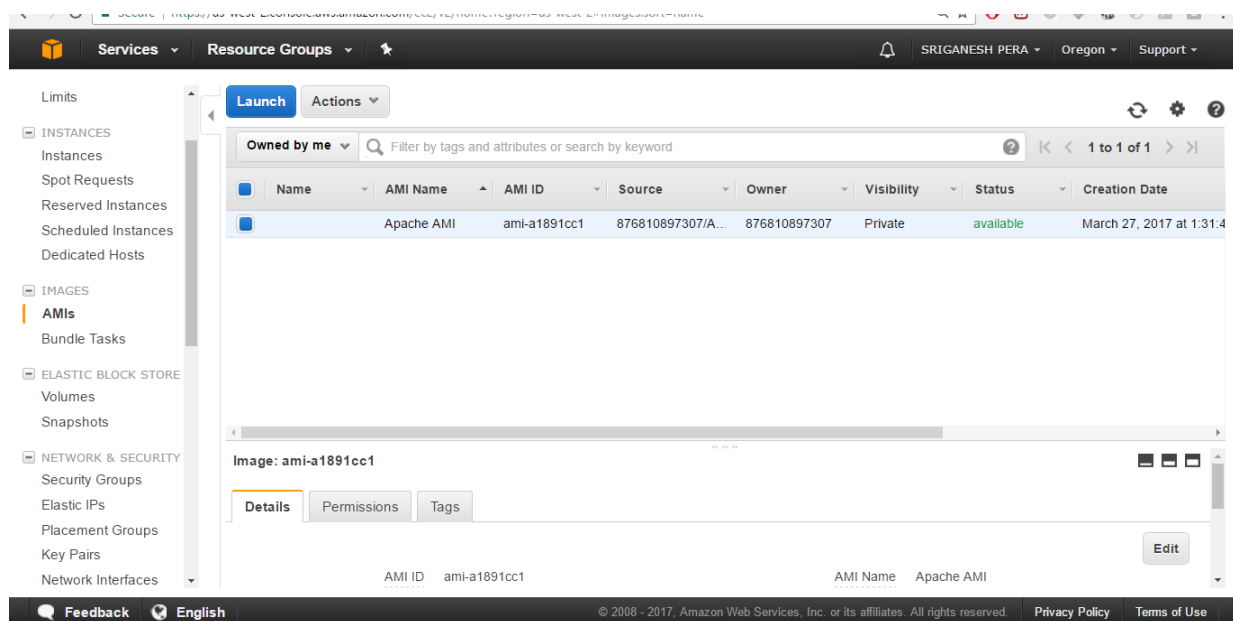
Now we have to **create an AMI** for this EC2 instance

Go to EC2 dash board and right click on the instance to create an image



Now we have created an AMI with pre-loaded Apache webserver on it

Which can be found in EC2 dashboard under AMI section



So as we know this AMI contains Apache webserver pre installed in it. After AMI is created terminate the instance.

Step 2: Create Launch configuration which can be found in EC2 dashboard and click on Create Auto Scaling group

The screenshot shows the AWS Management Console interface for the 'Welcome to Auto Scaling' page. The top navigation bar includes 'Services', 'Resource Groups', and a user profile 'SRIGANESH PERA' in the 'Oregon' region. The left sidebar lists various AWS services, with 'INSTANCES' expanded to show 'Instances', 'Spot Requests', 'Reserved Instances', 'Scheduled Instances', and 'Dedicated Hosts'. The main content area is titled 'Welcome to Auto Scaling' and includes a 'Create Auto Scaling group' button. Below this, a 'Benefits of Auto Scaling' section highlights three key features: 'Reusable Instance Templates', 'Automated Provisioning', and 'Adjustable Capacity'. Each feature is accompanied by an icon and a brief description. The bottom of the page features a 'Feedback' button, a language selector set to 'English', and copyright information for Amazon Web Services, Inc. (© 2008 - 2017).

Step 3: Click on launch configuration

The screenshot displays the 'Create Auto Scaling Group' wizard, specifically 'Step 1: Create launch configuration'. The top navigation bar is consistent with the previous screenshot. The main content area explains the purpose of a launch configuration: to create an Auto Scaling group, you first need to choose a template (launch configuration) that your group will use when launching instances. It also notes that you can create another launch configuration later to update the software of existing instances. A diagram illustrates this process: a person icon is shown next to a gear icon, followed by an equals sign and a stack of three gear icons, representing the creation of a launch configuration template. Below the diagram, the text states: 'Step 1: Create launch configuration. First, define a template that your Auto Scaling group will use to launch instances. You can change your group's launch configuration at any time.' At the bottom right, there are two buttons: 'Cancel' and 'Create launch configuration'. The bottom of the page includes the same footer as the previous screenshot.

Now this launch configuration is a template which launches an EC2 instance and also allows Auto scaling group associate with it.

Step 4: Click on my AMI and choose your AMI you just created. Here I choose Apache AMI

Services Resource Groups

1. Choose AMI 2. Choose Instance Type 3. Configure details 4. Add Storage 5. Configure Security Group 6. Review

Create Launch Configuration

Cancel and Exit

Quick Start

Search my AMIs

My AMIs

AWS Marketplace

Community AMIs

Ownership

☒ Owned by me

☐ Shared with me

Architecture

☐ 32-bit

☐ 64-bit

ami2 - ami-3e4cd85e

Root device type: ebs Virtualization type: hvm Owner: 876810897307

64-bit

Select

Apache AMI - ami-a1891cc1

Root device type: ebs Virtualization type: hvm Owner: 876810897307

64-bit

Select

Feedback English

© 2008 - 2017, Amazon Web Services, Inc. or its affiliates. All rights reserved. Privacy Policy Terms of Use

Step 5: Select t2.micro instance which is a free tier

Services Resource Groups

1. Choose AMI 2. Choose Instance Type 3. Configure details 4. Add Storage 5. Configure Security Group 6. Review

Create Launch Configuration

Amazon EC2 provides a wide selection of instance types optimized to fit different use cases. Instances are virtual servers that can run applications. They have varying combinations of CPU, memory, storage, and networking capacity, and give you the flexibility to choose the appropriate mix of resources for your applications. [Learn more](#) about instance types and how they can meet your computing needs.

Filter by: All instance types Current generation Show/Hide Columns

Currently selected: t2.micro (Variable ECUs, 1 vCPUs, 2.5 GHz, Intel Xeon Family, 1 GiB memory, EBS only)

	Family	Type	vCPUs	Memory (GiB)	Instance Storage (GB)	EBS-Optimized Available	Network Performance
<input type="checkbox"/>	General purpose	t2.nano	1	0.5	EBS only	-	Low to Moderate
<input checked="" type="checkbox"/>	General purpose	t2.micro Free tier eligible	1	1	EBS only	-	Low to Moderate
<input type="checkbox"/>	General purpose	t2.small	1	2	EBS only	-	Low to Moderate
<input type="checkbox"/>	General purpose	t2.medium	2	4	EBS only	-	Low to Moderate

Cancel Previous Next: Configure details

Feedback English

© 2008 - 2017, Amazon Web Services, Inc. or its affiliates. All rights reserved. Privacy Policy Terms of Use

Step 6: Name your launch configuration and click next: Add storage and also click on Next: Security configuration

Services Resource Groups

1. Choose AMI 2. Choose Instance Type 3. Configure details 4. Add Storage 5. Configure Security Group 6. Review

Create Launch Configuration

Amazon EC2 provides a wide selection of instance types optimized to fit different use cases. Instances are virtual servers that can run applications. They have varying combinations of CPU, memory, storage, and networking capacity, and give you the flexibility to choose the appropriate mix of resources for your applications. [Learn more](#) about instance types and how they can meet your computing needs.

Filter by: All instance types Current generation Show/Hide Columns

Currently selected: t2.micro (Variable ECUs, 1 vCPUs, 2.5 GHz, Intel Xeon Family, 1 GiB memory, EBS only)

	Family	Type	vCPUs	Memory (GiB)	Instance Storage (GB)	EBS-Optimized Available	Network Performance
<input type="checkbox"/>	General purpose	t2.nano	1	0.5	EBS only	-	Low to Moderate
<input checked="" type="checkbox"/>	General purpose	t2.micro Free tier eligible	1	1	EBS only	-	Low to Moderate
<input type="checkbox"/>	General purpose	t2.small	1	2	EBS only	-	Low to Moderate
<input type="checkbox"/>	General purpose	t2.medium	2	4	EBS only	-	Low to Moderate

Cancel Previous Next: Configure details

Step 7: Configure security settings and select create new security group and keep settings as shown in this screenshot and click on review

Services Resource Groups

1. Choose AMI 2. Choose Instance Type 3. Configure details 4. Add Storage 5. Configure Security Group 6. Review

Create Launch Configuration

A security group is a set of firewall rules that control the traffic for your instance. On this page, you can add rules to allow specific traffic to reach your instance. For example, if you want to set up a web server and allow Internet traffic to reach your instance, add rules that allow unrestricted access to the HTTP and HTTPS ports. You can create a new security group or select from an existing one below. [Learn more](#) about Amazon EC2 security groups.

Assign a security group: ☒ Create a new security group ☐ Select an existing security group

Security group name: AutoScaling-Security-Group-1

Description: AutoScaling-Security-Group-1 (2017-03-30 18:49:25.690+05:30)

Type	Protocol	Port Range	Source
SSH	TCP	22	My IP 106.51.26.95/32
HTTP	TCP	80	Anywhere 0.0.0.0/0

Add Rule

Cancel Previous Review

Feedback English © 2008 - 2017, Amazon Web Services, Inc. or its affiliates. All rights reserved. Privacy Policy Terms of Use

HTTP port 80 makes sure the client/server interaction is available

SSH my IP suggests that only I can securely login to my command prompt

Step 8: Review whatever the settings you made and click on Create Launch configuration and it asks for a key. Choose a key and click on Create Launch configuration

The screenshot shows the 'Create Launch Configuration' page in the AWS Management Console. The breadcrumb trail at the top indicates the steps: 1. Choose AMI, 2. Choose Instance Type, 3. Configure details, 4. Add Storage, 5. Configure Security Group, and 6. Review (which is the active step). The page title is 'Create Launch Configuration'. Under the 'Launch configuration details' section, the following settings are displayed: Name (mylaunchconfig1), Purchasing option (On demand), EBS Optimized (No), Monitoring (No), IAM role (None), Tenancy (Shared tenancy (multi-tenant hardware)), Kernel ID (Use default), RAM Disk ID (Use default), User data, and IP Address Type (Only assign a public IP address to instances launched in the default VPC and subnet. (default)). There are links to 'Edit details', 'Edit storage', and 'Edit security groups'. At the bottom of the configuration section are buttons for 'Cancel', 'Previous', and 'Create launch configuration'. The footer includes a 'Feedback' link, 'English' language selection, copyright information (© 2008 - 2017, Amazon Web Services, Inc. or its affiliates. All rights reserved.), and links to 'Privacy Policy' and 'Terms of Use'.

Now as soon as you click on **Create launch configuration** it takes you to **Auto scaling group** for setting up policies for you instance as you can see in the screenshot.

The screenshot shows the 'Create Auto Scaling Group' page in the AWS Management Console. The breadcrumb trail at the top indicates the steps: 1. Configure Auto Scaling group details (active), 2. Configure scaling policies, 3. Configure Notifications, 4. Configure Tags, and 5. Review. The page title is 'Create Auto Scaling Group'. The 'Launch Configuration' is set to 'mylaunchconfig1'. The 'Group name' field is empty. The 'Group size' is set to 'Start with 1 instances'. The 'Network' is set to 'vpc-d695eab3 (172.31.0.0/16) (default)', with a 'Create new VPC' button next to it. The 'Subnet' field is empty, with a 'Create new subnet' button next to it. A note states: 'Each instance in this Auto Scaling group will be assigned a public IP address.' There is a link to 'Advanced Details'. At the bottom of the configuration section are buttons for 'Cancel' and 'Next: Configure scaling policies'. The footer includes a 'Feedback' link, 'English' language selection, copyright information (© 2008 - 2017, Amazon Web Services, Inc. or its affiliates. All rights reserved.), and links to 'Privacy Policy' and 'Terms of Use'.

Step 9: Create a name for you Auto scaling group and choose default VPC and choose all Availability zones for your subnet and then click Next

The screenshot shows the 'Create Auto Scaling Group' wizard in the AWS Management Console. The top navigation bar includes 'Services', 'Resource Groups', and a user profile 'SRIGANESH PERA' in the 'Oregon' region. The wizard progress bar shows five steps: 1. Configure Auto Scaling group details (active), 2. Configure scaling policies, 3. Configure Notifications, 4. Configure Tags, and 5. Review. The main form is titled 'Create Auto Scaling Group' and includes a 'Cancel and Exit' link. The 'Launch Configuration' is set to 'mylaunchconfig1'. The 'Group name' is 'muautoscalinggroup1'. The 'Group size' is set to 'Start with 1 instances'. The 'Network' is 'vpc-d695eab3 (172.31.0.0/16) (default)', with a 'Create new VPC' link. The 'Subnet' dropdown shows three options: 'subnet-21fc9156(172.31.32.0/20) | Default in us-west-2a', 'subnet-7bd49b1e(172.31.16.0/20) | Default in us-west-2b', and 'subnet-294ec870(172.31.0.0/20) | Default in us-west-2c', with a 'Create new subnet' link. A note states: 'Each instance in this Auto Scaling group will be assigned a public IP address.' At the bottom, there is a 'Cancel' button and a 'Next: Configure scaling policies' button. The footer includes 'Feedback', 'English', copyright information '© 2008 - 2017, Amazon Web Services, Inc. or its affiliates. All rights reserved.', and links for 'Privacy Policy' and 'Terms of Use'.

Step 10: Now choose your scaling policy and click on [Use scaling policies to adjust the capacity of this group.](#)

This is a very important where in you choose number of instances for scaling up and scaling down which keeps your instance highly available when there is a increase/decrease of workload.

The screenshot shows the 'Create Auto Scaling Group' wizard in the AWS Management Console, specifically the 'Configure scaling policies' step. The progress bar shows five steps: 1. Configure Auto Scaling group details, 2. Configure scaling policies (active), 3. Configure Notifications, 4. Configure Tags, and 5. Review. The main form is titled 'Create Auto Scaling Group'. It features two sections: 'Increase Group Size' and 'Decrease Group Size'. The 'Increase Group Size' section has a 'Name' field with 'Increase Group Size', an 'Execute policy when' dropdown set to 'No alarm selected' with an 'Add new alarm' link, a 'Take the action' dropdown set to 'Add' with a value of '0' and a unit of 'instances', and an 'Instances need' field set to '300' seconds to warm up after each step. There is a 'Create a simple scaling policy' link. The 'Decrease Group Size' section has a 'Name' field with 'Decrease Group Size'. At the bottom, there are four buttons: 'Cancel', 'Previous', 'Review' (highlighted in blue), and 'Next: Configure Notifications'.

As you can see I have two policies that I need to choose 'Increase group size' when there is a load spike and 'decrease group size' when the load is less.

Step 11: Choose Scale between 1 and 4 instances for this lab. This means when the load is normal 1 instance works and when the load increases by 30% or 50%(CPU utilization rate) it launches upto 4 instances

Services Resource Groups

1. Configure Auto Scaling group details 2. Configure scaling policies 3. Configure Notifications 4. Configure Tags 5. Review

Create Auto Scaling Group

Use scaling policies to adjust the capacity of this group

Scale between 1 and 4 instances. These will be the minimum and maximum size of your group.

Increase Group Size

Name: Increase Group Size

Execute policy when: No alarm selected Add new alarm

Take the action: Add 0 instances Add step

Instances need: 300 seconds to warm up after each step

Create a simple scaling policy

Cancel Previous Review Next: Configure Notifications

Step 12: Increase group size by 1 in the take action field and click on create an alarm and set CPU utilization to 30% for every 1 minute and click on create alarm

Services Resource Groups

1. Configure Auto Scaling group details 2. Configure scaling policies 3. Configure Notifications 4. Configure Tags 5. Review

Create Auto Scaling Group

Increase Group Size

Name: Increase Group Size

Execute policy when: awsec2-muautoscalinggroup1-CPU-Utilization Edit Remove
breaches the alarm threshold: CPUUtilization >= 30 for 60 seconds
for the metric dimensions AutoScalingGroupName = muautoscalinggroup1

Take the action: Add 1 instances when 30 <= CPUUtilization < +infinity Add step

Instances need: 300 seconds to warm up after each step

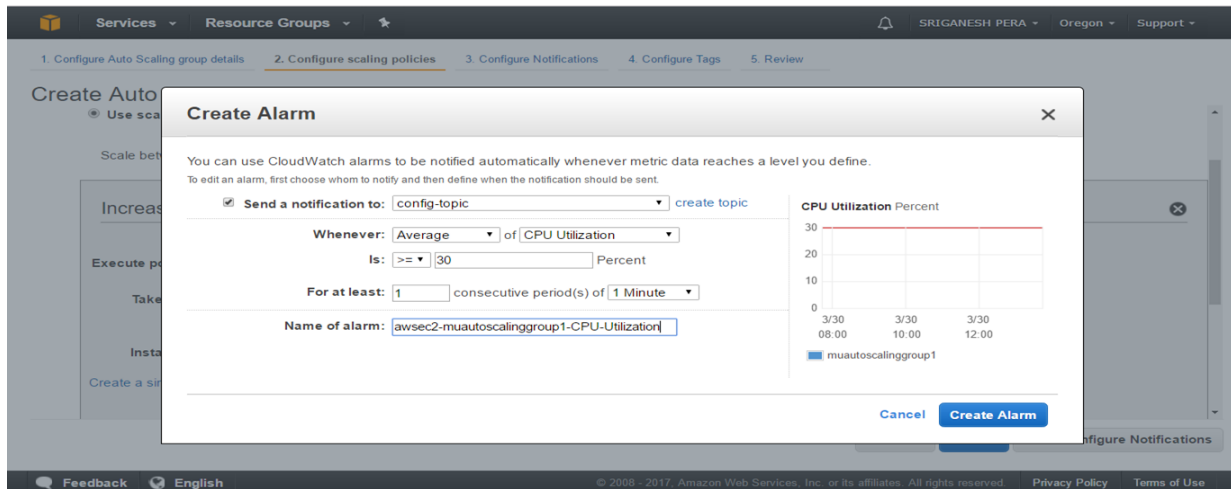
Create a simple scaling policy

Decrease Group Size

Name: Decrease Group Size

Cancel Previous Review Next: Configure Notifications

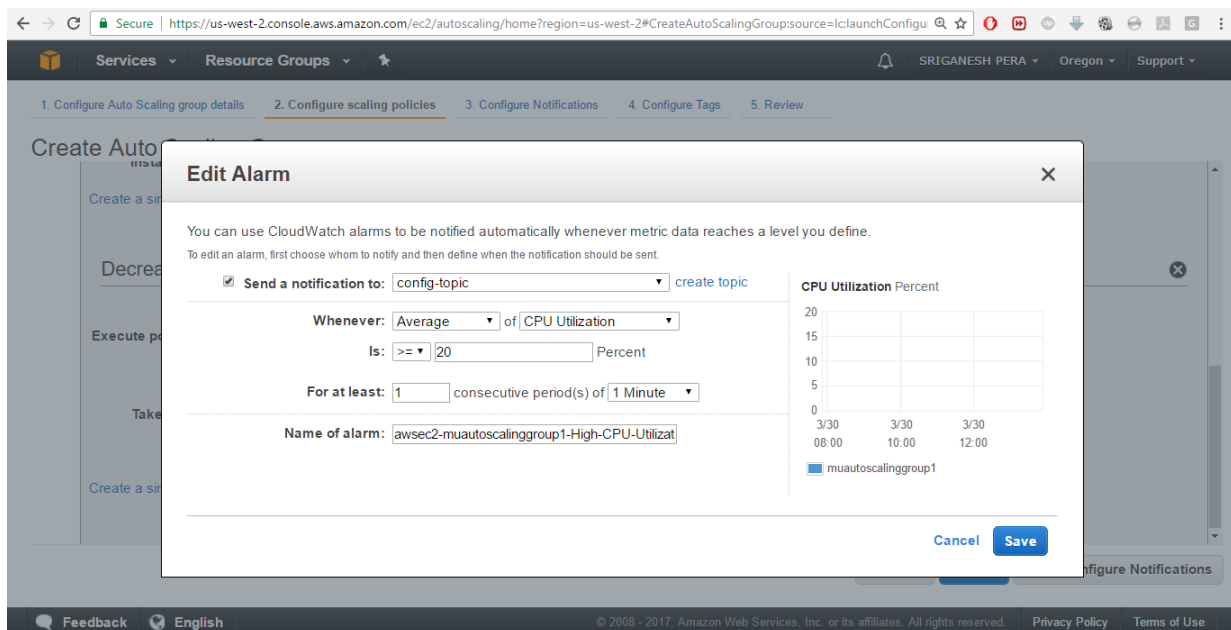
Feedback English © 2008 - 2017, Amazon Web Services, Inc. or its affiliates. All rights reserved. Privacy Policy Terms of Use



As you can see I set the CPU utilization to 30% and 1 instance for every 1 minute which means if CPU utilization reaches 30% for every minute 1 instance gets launched.

This setting is for demo purpose but for more practical scenario we need to choose time as 5 minutes or 15 minutes which is the best practice for choosing Auto scaling group

Step 13: Decrease group size and now click on create an alarm and set CPU utilization to 20% and select instance 1 and set timing as 1 minute



Now this says that whenever CPU utilization decrease to 20% terminate 1 instance

Now select 1 in the take action field choose '**Remove**' and then click on Configure notifications

Services Resource Groups

1. Configure Auto Scaling group details 2. Configure scaling policies 3. Configure Notifications 4. Configure Tags 5. Review

Create Auto Scaling Group

instances need: 600 seconds to warm up after each step

Create a simple scaling policy ⓘ

Decrease Group Size

Name: Decrease Group Size

Execute policy when: awsec2-muautoscalinggroup1-High-CPU-Utilization Edit Remove
breaches the alarm threshold: CPUUtilization >= 20 for 60 seconds
for the metric dimensions AutoScalingGroupName = muautoscalinggroup1

Take the action: Remove 1 instances when 20 <= CPUUtilization < +infinity

Add step ⓘ

Create a simple scaling policy ⓘ

Cancel Previous Review Next: Configure Notifications

Feedback English © 2008 - 2017, Amazon Web Services, Inc. or its affiliates. All rights reserved. Privacy Policy Terms of Use

Step 14: Click on add notification and create new topic and set notifications as per your requirement and make AWS send an E-mail(your aws account email) to you in case of events of triggered as per our requirement

Services Resource Groups

1. Configure Auto Scaling group details 2. Configure scaling policies 3. Configure Notifications 4. Configure Tags 5. Review

Create Auto Scaling Group

Configure your Auto Scaling group to send notifications to a specified endpoint, such as an email address, whenever a specified event takes place, including: successful launch of an instance, failed instance launch, instance termination, and failed instance termination.

If you created a new topic, check your email for a confirmation message and click the included link to confirm your subscription. Notifications can only be sent to confirmed addresses.

Send a notification to: Auto scaling notifications use existing topic

With these recipients: nincndude@gmail.com

Whenever instances:



- ☒ launch
- ☒ terminate
- ☒ fail to launch
- ☒ fail to terminate


Add notification

Cancel Previous Review Next: Configure Tags


Feedback English © 2008 - 2017, Amazon Web Services, Inc. or its affiliates. All rights reserved. Privacy Policy Terms of Use

Now click on Tags and create one(optional) if you want.If you don't want then click review and check everything before you click on create Autoscaling group

 Services ▾ Resource Groups ▾ 

 SRIGANESH PERA ▾ Oregon ▾ Support ▾

Auto Scaling group creation status



 **Successfully created Auto Scaling group**
[View creation log](#)

▼ View

- [View your Auto Scaling groups](#)
- [View your launch configurations](#)

▶ Here are some helpful resources to get you started

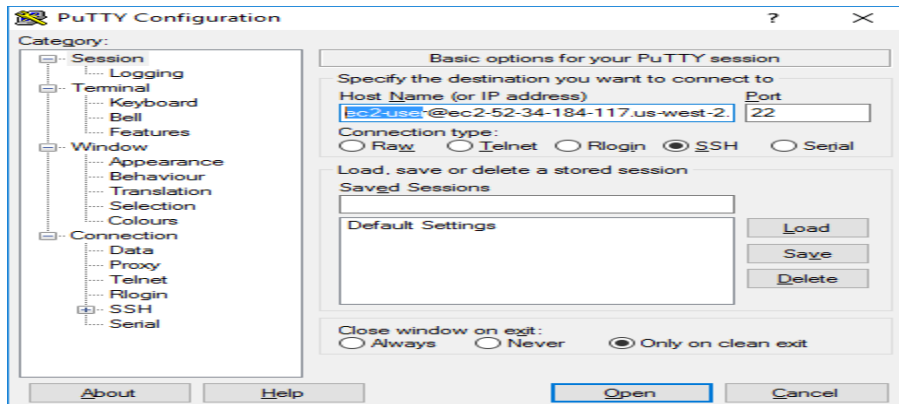
Close

 Feedback  English

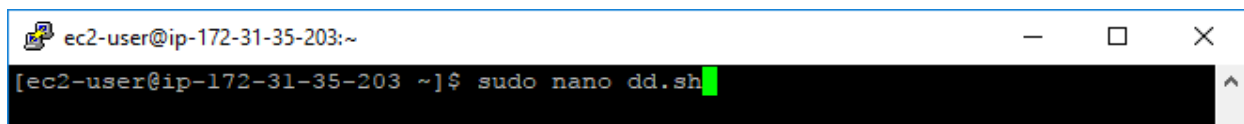
© 2008 - 2017, Amazon Web Services, Inc. or its affiliates. All rights reserved. [Privacy Policy](#) [Terms of Use](#)

Now go back to EC2 dashboard to see an instance running which is automatically created for us by Auto scaling group.Can you tell why an instance is already created for us by AWS?

Step 15: Now SSH into your instance(Launch Configuration one) and install a script using dd command(which sends garbage data to dev/null) and execute it with nohup command!

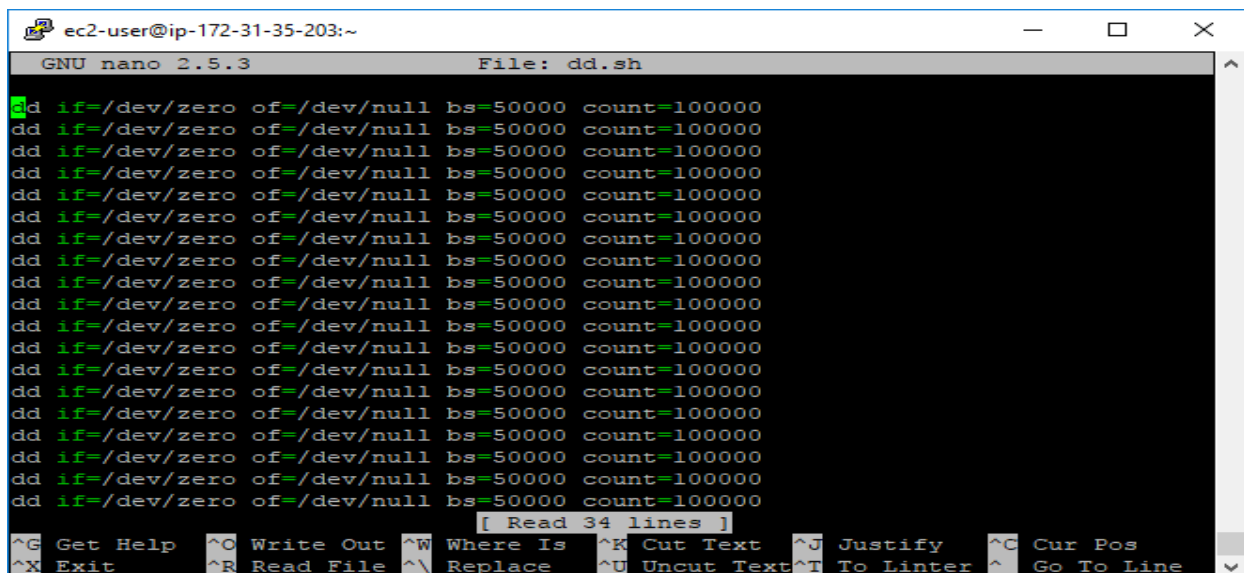


Now create a file dd.sh



Now paste this script multiple times as shown:

dd if=/dev/zero of=/dev/null bs=50000 count=100000



Now lets see our CPU utilization

As you can see my cpu utilization is 0%

```
ec2-user@ip-172-31-20-109:~  
top - 14:52:32 up 14 min, 1 user, load average: 0.00, 0.00, 0.00  
Tasks: 79 total, 1 running, 78 sleeping, 0 stopped, 0 zombie  
Cpu(s): 0.0%us, 0.0%sy, 0.0%ni,100.0%id, 0.0%wa, 0.0%hi, 0.0%si, 0.0%st  
Mem: 1019116k total, 170968k used, 848148k free, 8724k buffers  
Swap: 0k total, 0k used, 0k free, 92036k cached  


| PID  | USER   | PR | NI  | VIRT  | RES  | SHR  | S | %CPU | %MEM | TIME+   | COMMAND       |
|------|--------|----|-----|-------|------|------|---|------|------|---------|---------------|
| 2631 | apache | 20 | 0   | 401m  | 14m  | 6916 | S | 0.3  | 1.4  | 0:00.07 | httpd         |
| 1    | root   | 20 | 0   | 19632 | 2424 | 2108 | S | 0.0  | 0.2  | 0:00.94 | init          |
| 2    | root   | 20 | 0   | 0     | 0    | 0    | S | 0.0  | 0.0  | 0:00.00 | kthreadd      |
| 3    | root   | 20 | 0   | 0     | 0    | 0    | S | 0.0  | 0.0  | 0:00.00 | ksoftirqd/0   |
| 4    | root   | 20 | 0   | 0     | 0    | 0    | S | 0.0  | 0.0  | 0:00.00 | kworker/0:0   |
| 5    | root   | 0  | -20 | 0     | 0    | 0    | S | 0.0  | 0.0  | 0:00.00 | kworker/0:0H  |
| 6    | root   | 20 | 0   | 0     | 0    | 0    | S | 0.0  | 0.0  | 0:00.00 | kworker/u30:0 |
| 7    | root   | 20 | 0   | 0     | 0    | 0    | S | 0.0  | 0.0  | 0:00.02 | rcu_sched     |
| 8    | root   | 20 | 0   | 0     | 0    | 0    | S | 0.0  | 0.0  | 0:00.00 | rcu_bh        |
| 9    | root   | RT | 0   | 0     | 0    | 0    | S | 0.0  | 0.0  | 0:00.00 | migration/0   |
| 10   | root   | 20 | 0   | 0     | 0    | 0    | S | 0.0  | 0.0  | 0:00.00 | kdevtmpfs     |
| 11   | root   | 0  | -20 | 0     | 0    | 0    | S | 0.0  | 0.0  | 0:00.00 | netns         |
| 12   | root   | 0  | -20 | 0     | 0    | 0    | S | 0.0  | 0.0  | 0:00.00 | perf          |
| 13   | root   | 20 | 0   | 0     | 0    | 0    | S | 0.0  | 0.0  | 0:00.00 | kworker/u30:1 |
| 14   | root   | 20 | 0   | 0     | 0    | 0    | S | 0.0  | 0.0  | 0:00.00 | xenwatch      |
| 19   | root   | 20 | 0   | 0     | 0    | 0    | S | 0.0  | 0.0  | 0:00.00 | xenbus        |
| 20   | root   | 20 | 0   | 0     | 0    | 0    | S | 0.0  | 0.0  | 0:00.05 | kworker/0:1   |


```

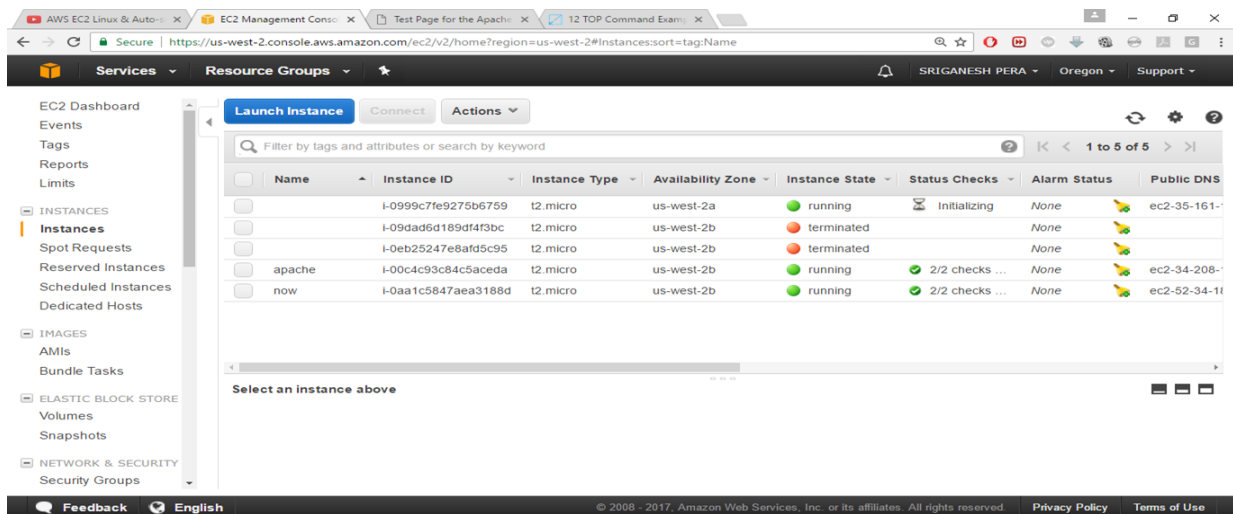
Make sure you give read write permission to your file

```
ec2-user@ip-172-31-20-109:~  
[ec2-user@ip-172-31-20-109 ~]$ chmod 755 dd.sh  
[ec2-user@ip-172-31-20-109 ~]$
```

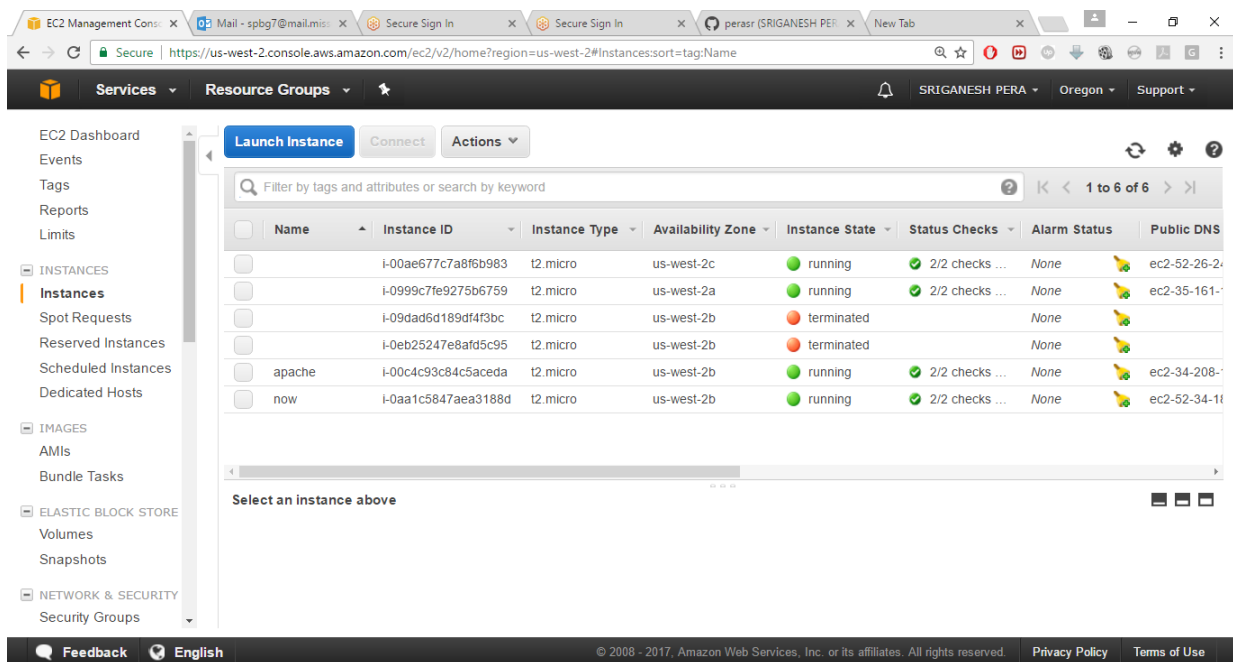
Now run the script multiple times which spikes the CPU utilization using the following command

\$ nohup ./dd.sh &

After 5 minutes(cooling period) check your EC2 dashboard which will show the EC2 instance added automatically

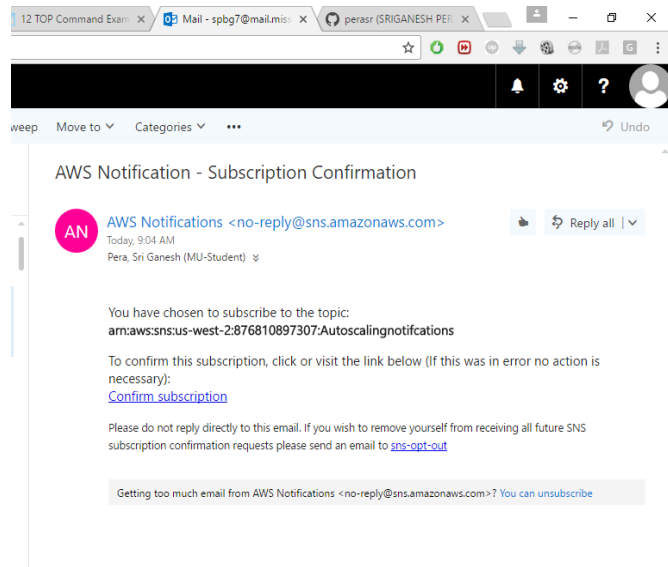


The first instance is the automatically created instance after the CPU utilization spike



Now I have got one more instance running.

I got a Notification as well from AWS autoscaling group with regards to the CPU util spike.



Step 15:Cleanup

Terminate all instances,auto scaling,launch scaling groups!