Auto Scaling

Introduction to Auto Scaling

Auto Scaling

In the previous lesson, we saw how the Load Balancer manages traffic between our existing resources, in other words, instances.

But, what if we want to automate the increasing or decreasing of my existing virtual machines when demand changes over time?

We call **Auto Scaling** that makes this process in AWS for you. Amazon EC2 Auto Scaling is a component that helps you ensure that you have the correct number of Amazon EC2 instances available to handle the load for your application.

Thanks to Auto Scaling, we can add or remove a new virtual machine when it is necessary. So, you can expand your system to respond to that demand according to the size of the request you receive, and automatically collapse when the demand decreases.

With Auto Scaling, you can also define certain rules and create new resources in the system according to these rules.

You can set rules and give direction which depends on some sort of condition. For example, you may say;

- If the virtual machines use more than 90% CPU resources for 5 minutes, add a new virtual machine,
- Keep running until the CPU usage of other machines decreases to 30%,
- If the CPU usage drops below 30%, delete this virtual machine.

Benefits of Auto Scaling

Setup Scaling Quickly
Pay Only for What You Need
Automatically Maintain Performance
Make Smart Scaling Decisions

Benefits of Auto Scaling

Setup Scaling Quickly:

AWS Auto Scaling lets you set target utilization levels for multiple resources in a single, intuitive interface. You can quickly see the average utilization of all of your scalable resources without having to navigate to other consoles.

Pay Only for What You Need:

AWS Auto Scaling can help you optimize your utilization and cost efficiencies when consuming AWS services so you only pay for the resources you actually need. When demand drops, AWS Auto Scaling will automatically remove any excess resource capacity so you avoid overspending. **AWS Auto Scaling is free to use and allows you to optimize the costs of your AWS environment.**

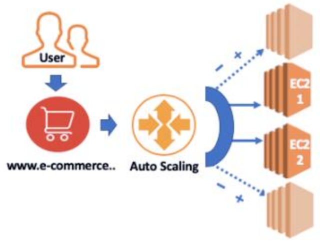
Automatically Maintain Performance:

Using AWS Auto Scaling, you maintain optimal application performance and availability, even when workloads are periodic, unpredictable, or continuously changing. AWS Auto Scaling continually monitors your applications to make sure that they are operating at your desired performance levels.

Make Smart Scaling Decisions:

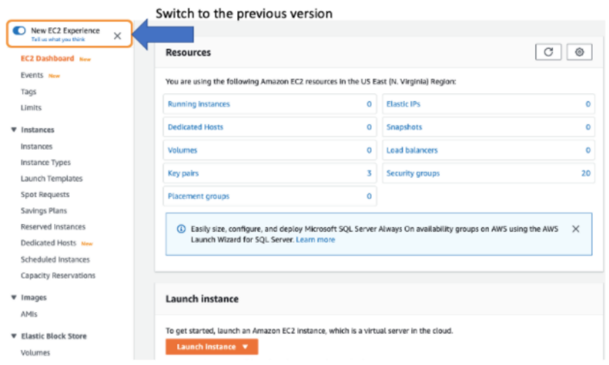
If you try to make scaling instead of AWS Auto Scaling you may not always find an optimum solution. AWS Auto Scaling automatically creates all of the scaling policies and sets targets for you based on your preference. AWS Auto Scaling monitors your application and automatically adds or removes capacity from your resource groups in real-time as demands change.

Auto Scaling Working Process



Auto Scaling Working Process

Auto Scaling Creating Process



You are using the following Amazon EC2 resources in the US East (N. Virginia) Region:

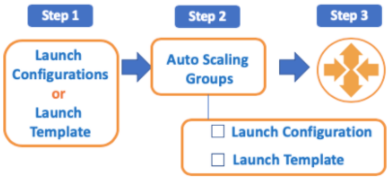
Running instances	0	Elastic IPs	0
Dedicated Hosts	0	Snapshots	0
Volumes	0	Load balancers	0
Key pairs	3	Security groups	20
Placement groups	0		

Easily size, configure, and deploy Microsoft SQL Server Always On availability groups on AWS using the AWS Launch Wizard for SQL Server. Learn more

To get started, launch an Amazon EC2 instance, which is a virtual server in the cloud.

⚠ **Avoid ! :**

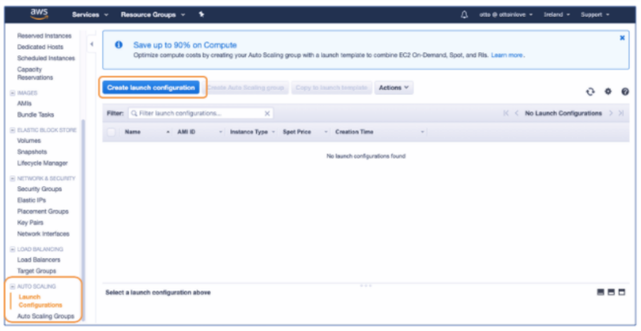
- Auto Scaling menu is newly changed by AWS . LMS content is more compatible with the previous version. So, it will be better to switch the page version as you see in the picture above.



While creating an Auto Scaling;

- First, we need to **Launch Configuration** or **Launch Templates**. Auto Scaling will create a new instance according to the features determined by Launch Configuration or Launch Templates. In fact, these are templates of instance. **Launch Template** is designed in place of **Launch Configuration**. But, we'll use **Launch Configuration** option for now.
- Then, we create an Auto Scaling Group. While creating an Auto Scaling Group we can choose either **Launch Configuration** or **Launch Templates** that we created before.
- Finally, we create an Auto Scaling.

Launch Configurations Menu



Save up to 90% on Compute
Optimize compute costs by creating your Auto Scaling group with a launch template to combine EC2 On-Demand, Spot, and RIs. Learn more.

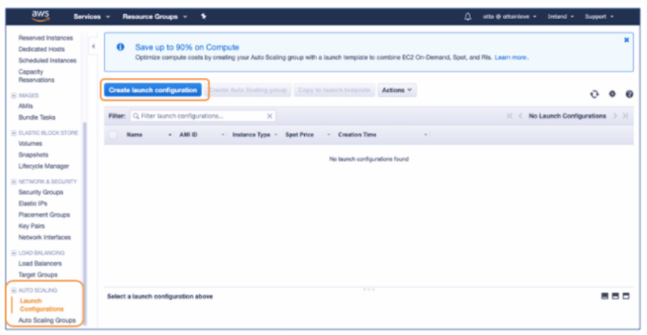
No launch configurations found

Select a launch configuration above

Launch Configuration Menu

In the EC2 AWS Management console, select Launch Configurations from the left-hand menu and then click **Create Launch Configuration** to start.

Launch Configurations Menu



Save up to 90% on Compute
Optimize compute costs by creating your Auto Scaling group with a launch template to combine EC2 On-Demand, Spot, and RIs. Learn more.

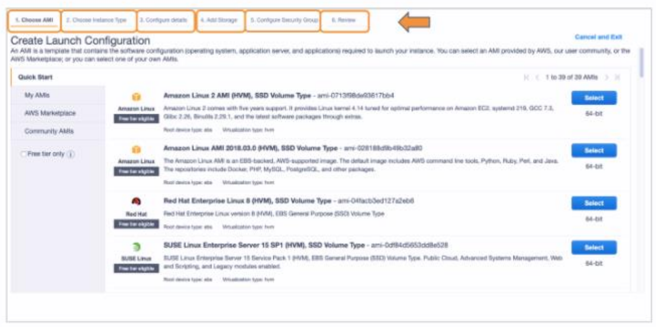
No launch configurations found

Select a launch configuration above

Launch Configuration Menu

In the EC2 AWS Management console, select Launch Configurations from the left-hand menu and then click **Create Launch Configuration** to start.

Create Launch Configurations



Create Launch Configuration
An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. You can select an AMI provided by AWS, our user community, or the AWS Marketplace; or you can select one of your own AMIs.

Amazon Linux 2 AMI (HVM), SSD Volume Type - ami-0713f98de93617bb4
Amazon Linux 2 comes with five years support. It provides Linux kernel 4.14 tuned for optimal performance on Amazon EC2, systemd 219, GCC 7.3, Glibc 2.26, Binutils 2.29.1, and the latest software packages through extras.

Amazon Linux AMI 2018.03.0 (HVM), SSD Volume Type - ami-028188d9b49b32a80
The Amazon Linux AMI is an EBS-backed, AWS-supported image. The default image includes AWS command line tools, Python, Ruby, Perl, and Java. The repositories include Docker, PHP, MySQL, PostgreSQL, and other packages.

Red Hat Enterprise Linux 8 (HVM), SSD Volume Type - ami-04facb3ed127a2eb6
Red Hat Enterprise Linux version 8 (HVM), EBS General Purpose (SSD) Volume Type

SUSE Linux Enterprise Server 15 SP1 (HVM), SSD Volume Type - ami-0df84d5653dd8e528
SUSE Linux Enterprise Server 15 Service Pack 1 (HVM), EBS General Purpose (SSD) Volume Type. Public Cloud, Advanced Systems Management, Web and Scripting, and Legacy modules enabled.

Create Launch Configuration Steps

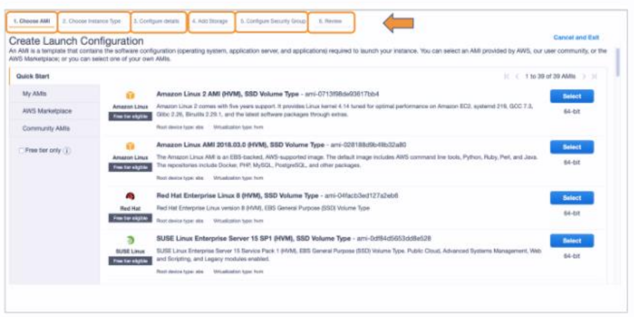
As you see in the picture above, it is the same menu and the same steps as if you launch an EC2 instance.

We'll determine what we need to choose from disk size, IAM role, key pair to AMI type, we will do everything just like creating a new virtual machine.

But at the end, we will not create a virtual machine. Instead, we'll actually save these things as a template and Auto Scaling will create a new instance according to this configuration when needed.

Let's see.

Launch Configuration Steps



Create Launch Configuration Steps

Step 1 Choose AMI:,

We select the Linux AMI top of the list.

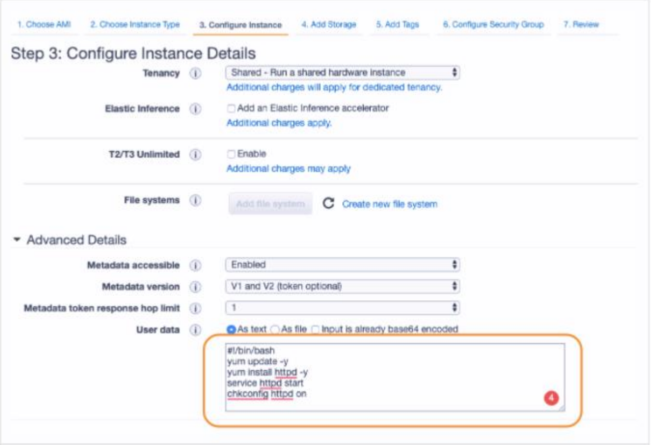
Step 2 Choose Instance Type:

t2.micro is enough for now. We select this one.

Step 3 Configure Details:

- **Name:** Let's name as First-Launch-Config
- **Purchasing option Request:** No need to Spot Instances.
- **IAM Role:** We choose an existing role.
- **Monitoring:** We don't want to open Detailed Monitoring in, for now, so keep it as is.
- **Advanced Details:** To ensure instance to be installed web server software automatically when created, we enter the following command in the **User Data** section, as you see in the picture below:

```
#!/bin/bash
yum update -y
yum install httpd -y
service httpd start
chkconfig httpd on
```



Step 3: Configure Instance Details

Tenancy: Shared - Run a shared hardware instance
Additional charges will apply for dedicated tenancy.
Elastic Inference: Add an Elastic Inference accelerator
Additional charges apply.
T2/T3 Unlimited: Enable
Additional charges may apply
File systems: Add file system | Create new file system

Advanced Details
Metadata accessible: Enabled
Metadata version: V1 and V2 (token optional)
Metadata token response hop limit: 1
User data: [✓ As text] As file | Input is already base64 encoded

```
#!/bin/bash
yum update -y
yum install httpd -y
service httpd start
chkconfig httpd on
```

User Data

Step 4 Choose Instance Type:

In the storage part, the 8 GB gp2 root device is sufficient in our example.

Step 5 Configure Security Group:

In this part, we select the group that we created before.

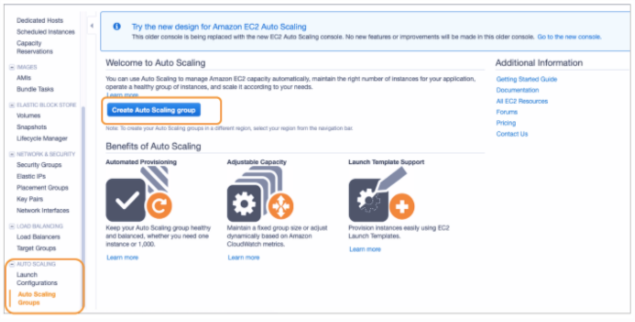
Step 6 Review:

Check the details of the configuration here.

- Click the **Create Launch Configuration** tab and pick a key pair. We confirm that we still have the key as we will continue with the existing one.
- We click **Create Launch Configuration** again.

Now, it's ready.

Creating an Auto Scaling Group



Try the new design for Amazon EC2 Auto Scaling
This older console is being replaced with the new EC2 Auto Scaling console. No new features or improvements will be made in this older console. Go to the new console.

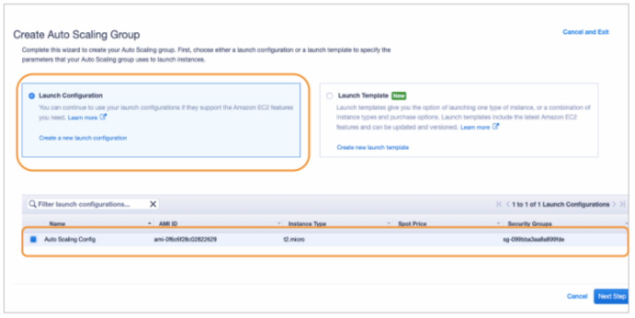
Welcome to Auto Scaling
You can use Auto Scaling to manage Amazon EC2 capacity automatically, maintain the right number of instances for your application, operate a healthy group of instances, and scale it according to your needs.

Note: To create your Auto Scaling groups in a different region, select your region from the navigation bar.

Benefits of Auto Scaling
Automated Provisioning: Keep your Auto Scaling group healthy and balanced, whether you need one instance or 1,000.
Adjustable Capacity: Maintain a fixed group size or adjust dynamically based on Amazon CloudWatch metrics.
Launch Template Support: Provision instances easily using EC2 Launch Templates.

Auto Scaling Menu

In the EC2 AWS Management console, select Auto Scaling Group from the left-hand menu at the bottom and then click **Create Auto Scaling Group** to start.


Cancel and Exit
Create a new launch configuration
Create new launch template
Filter launch configurations...
1 to 1 of 1 Launch Configurations
Cancel
Next Step

Create Auto Scaling Group
Complete this wizard to create your Auto Scaling group. First, choose either a launch configuration or a launch template to specify the parameters that your Auto Scaling group uses to launch instances.

[✓ Launch Configuration]
You can continue to use your launch configurations if they support the Amazon EC2 features you need. Learn more

Launch Template New
Launch templates give you the option of launching one type of instance, or a combination of instance types and purchase options. Launch templates include the latest Amazon EC2 features and can be updated and versioned. Learn more

Name	AMI ID	Instance Type	Spot Price	Security Groups
Auto Scaling Config	ami-0f6c6f28c02822629	t2.micro		sg-099bba3aa8a899fde

Auto Scaling Menu

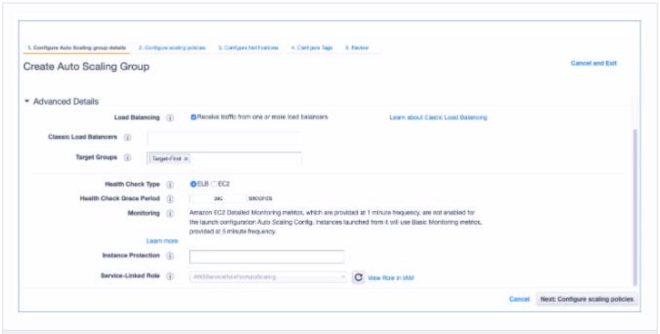
When the page is opened;

First, we are asked what type of template to use. We have 2 options: Launch Configuration and Launch Template.

Since we created Launch Configuration before, we will continue with **Launch Configuration**.

Then we select the Launch Configuration that we just created, and click **Next** to continue.

Step 1 - Configure Auto Scaling Group Details


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

Create Auto Scaling Group — Cancel and Exit

Advanced Details

Load Balancing	☑ Receive traffic from one or more load balancers — Learn about Classic Load Balancing
Classic Load Balancers	
Target Groups	Target-First
Health Check Type	● ELB ○ EC2
Health Check Grace Period	300 seconds
Monitoring	Amazon EC2 Detailed Monitoring metrics, which are provided at 1 minute frequency, are not enabled for the launch configuration Auto Scaling Config. Instances launched from it will use Basic Monitoring metrics, provided at 5 minute frequency. Learn more
Instance Protection	
Service-Linked Role	AWSServiceRoleForAutoScaling — View Role in IAM

Cancel | Next: Configure scaling policies

Configure Auto Scaling Group Details

Name:

The Auto Scaling Group is named as First-Group.

Group Size:

This part is critical for the Auto Scaling Group process. Here, we need to decide how many machines will be created by Auto Scaling.

According to our scenario, there should be at least 2 machines at a time so that one can back up the other in case of malfunction. That's why we set the number of Instances that this AutoScaling Group will create as 2.

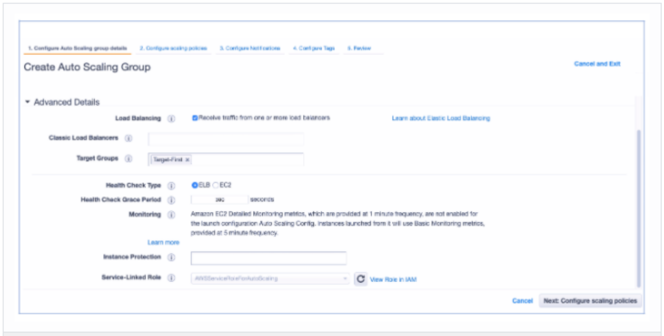
Network:

We select default VPC

Subnet:

Let's select the 3 subnets under the default VPC for now.

Step 1 - Advanced Details


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

Create Auto Scaling Group — Cancel and Exit

Advanced Details

Load Balancing	☑ Receive traffic from one or more load balancers — Learn about Classic Load Balancing
Classic Load Balancers	
Target Groups	Target-First
Health Check Type	● ELB ○ EC2
Health Check Grace Period	300 seconds
Monitoring	Amazon EC2 Detailed Monitoring metrics, which are provided at 1 minute frequency, are not enabled for the launch configuration Auto Scaling Config. Instances launched from it will use Basic Monitoring metrics, provided at 5 minute frequency. Learn more
Instance Protection	
Service-Linked Role	AWSServiceRoleForAutoScaling — View Role in IAM

Cancel | Next: Configure scaling policies

Advanced Details

Advanced Details Section

- **Load Balancing:**

First, AWS asks you whether you want to automatically add the machines you will create with this Auto Scaling to the back of a load balancer you have already created or not.

Check the box that says **Receive traffic from one or more load balancers**. Because we want our Auto Scaling to be coordinated with Load Balancer.

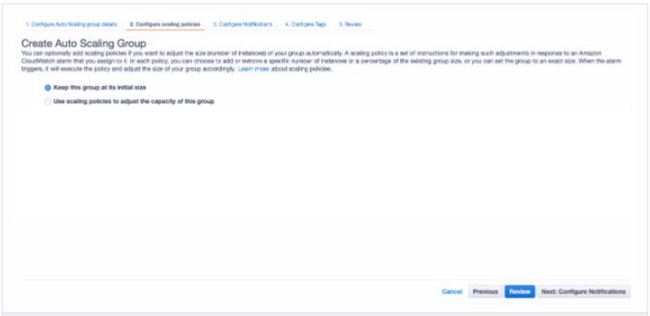
Classic Load Balancers:

If you want to check this, you can not handle it. Because we created the application load balancer, not a classic load balancer.

Target Groups:

We select our target group named Target-First that is created for the load balancer.

Step 2 - Configure Scaling Policies


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

Create Auto Scaling Group

You can optionally add scaling policies if you want to adjust the size (number of instances) of your group automatically. A scaling policy is a set of instructions for making such adjustments in response to an Amazon CloudWatch alarm that you assign to it. In each policy, you can choose to add or remove a specific number of instances or a percentage of the existing group size, or you can set the group to an exact size. When the alarm triggers, it will execute the policy and adjust the size of your group accordingly. Learn more about scaling policies.

[✓ Keep this group at its initial size]
Use scaling policies to adjust the capacity of this group

Cancel | Previous | Review | Next: Configure Notifications

Configure Scaling Policies

This is the most important part of creating an Auto Scaling process where we determine the rules.

We have two options here. These are;

- **Keep This Group at Initial Size**
- **Use Scaling Policies to Adjust the Capacity of This Group**

If we choose the first option, Keep This Group at Initial Size, we ensure that Auto Scaling Group will always have 2 virtual machines running under all circumstances. Because we determined Group Size as 2 in "Configure Auto Scaling Group Details" section

So, when this is selected, the Auto Scaling Group will create 2 machines for us.

If we somehow terminate these 2 machines, Auto Scaling will re-create 2 machines.

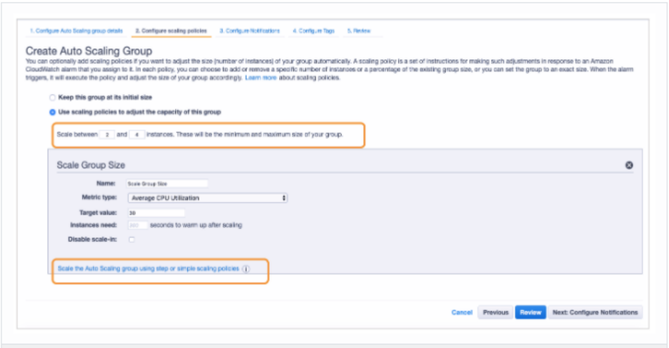
And, if there is a problem with one of the virtual machines, Auto Scaling will create a new one to delete this machine.

But that's not what we want. In fact,

- We want new virtual machines to be added when we receive too many requests for them or,
- If this intense demand is reduced, we want these added machines to be deleted afterward.

We can do this with the second option. Let's see how.

Step 2 - Configure Scaling Policies (Using a Scaling Policies)


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

Create Auto Scaling Group

You can optionally add scaling policies if you want to adjust the size (number of instances) of your group automatically. A scaling policy is a set of instructions for making such adjustments in response to an Amazon CloudWatch alarm that you assign to it. In each policy, you can choose to add or remove a specific number of instances or a percentage of the existing group size, or you can set the group to an exact size. When the alarm triggers, it will execute the policy and adjust the size of your group accordingly. Learn more about scaling policies.

Keep this group at its initial size
[✓ Use scaling policies to adjust the capacity of this group]

Scale between 2 and 4 instances. These will be the minimum and maximum size of your group.

Scale Group Size

Name:	Scale Group Size
Metric type:	Average CPU Utilization
Target value:	30
Instances need:	300 seconds to warm up after scaling
Disable scale-in:	☐

Scale the Auto Scaling group using step or simple scaling policies

Cancel | Previous | Review | Next: Configure Notifications

Scaling Policies

This is the most important part where Auto Scaling is planned. Here we asked to prefer scaling values first. We prefer a range of **2** and **4**

This means that;

- Auto Scaling will ensure that there are at least 2 machines at a time.
- But, if the capacity usage increase Auto Scaling will add up to 4 new machines or,
- If the capacity usage decrease, the number of virtual machines will be reduced to 2 again.

Name:

We write the name as **Scale Group Size**

Metric Type:

Now, we will determine the rules according to which rule we will set the number of machines between 2 and 4.

Here we have 4 metric types as you see in options. But the most common ones are **Application Load Balancer Request Count Per Target** and **Average CPU Utilization.**

Application Load Balancer Request Count Per Target:

This option allows auto-scaling based on the number of user requests selected.

Average CPU Utilization:

If you select this option, Auto Scaling takes into account CPU usage while adding or removing the instance. Let's continue with this option.

Target Value:

We select 30 as the target value. It means that,

- If the average CPU of the machines in the environment is below 30%, we want the number of the instance to be stable.
- But, if CPU usage exceeds 30%, this time we order to AWS to add machines here.

Now we have described this in a simple way, but at the bottom of the page, you'll see the **Scale the Auto Scaling Group using step or simple scaling policies** option. If you click this tab you can reach advance settings. Since we prefer the simple scaling, for now, leave it as is.

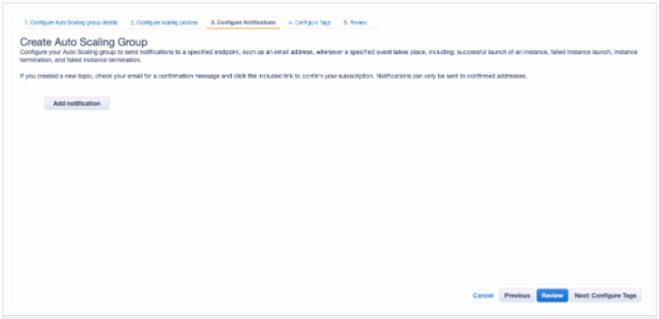
30% of the Average CPU Usage option is enough for our scenario.

Instances Need:

This is the amount of time that your instances need to warm up. Keep it as 300

Let's click next and go on.

Step 3 - Configure Notifications


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.

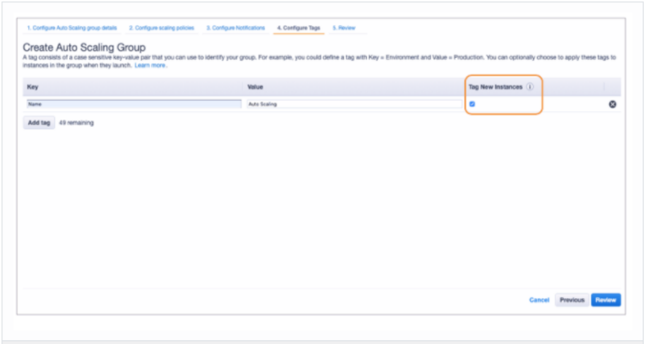
If you created a new topic, check your email for a confirmation message and click the included link to confirm your subscription. Notifications can only be sent to confirmed addresses.

Configure Notifications

If we want AWS to send us a notification when Auto Scaling creates or removes a new machine, we can click **Add Notification**.

Let's skip this step for now and click next.

Step 4 - Configure Tags


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

Create Auto Scaling Group

A tag consists of a case sensitive key-value pair that you can use to identify your group. For example, you could define a tag with Key = Environment and Value = Production. You can optionally choose to apply these tags to instances in the group when they launch. Learn more.

Key	Value	Tag New Instances
Name	Auto Scaling	✓

Add tag
49 remaining
Cancel
Previous
Review

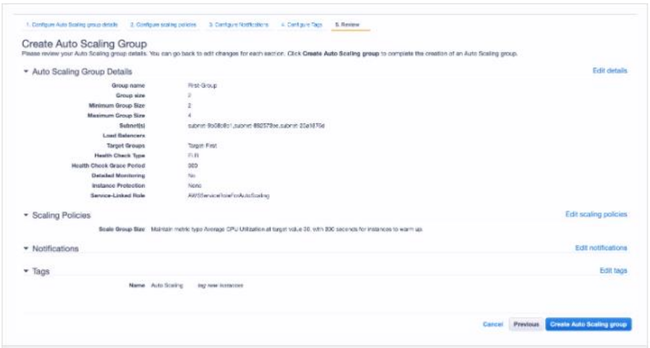
Configure Tags

Here, we can determine the tags we want. For example, let's write **Name** for Key and **Auto Scaling** for Value

Then, if we click **Tag New Instances** flag here, it will assign this tag to all instances created with the Auto Scaling.

We can give the same tag to all newly created instances in this way.

Step 5 - Review


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

Create Auto Scaling Group

Please review your Auto Scaling group details. You can go back to edit changes for each section. Click **Create Auto Scaling group** to complete the creation of an Auto Scaling group.

Auto Scaling Group Details

Group name	First-Group
Group size	2
Minimum Group Size	2
Maximum Group Size	4
Subnet(s)	subnet-9b58c8c1,subnet-882573ee,subnet-25a1876d
Load Balancers	
Target Groups	Target-First
Health Check Type	ELB
Health Check Grace Period	300
Detailed Monitoring	No
Instance Protection	None
Service-Linked Role	AWSServiceRoleForAutoScaling

Scaling Policies

Scale Group Size: Maintain metric type Average CPU Utilization at target value 30, with 300 seconds for instances to warm up.

Notifications

Tags

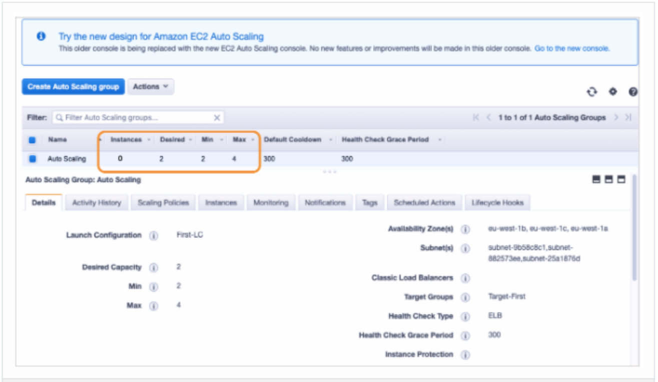
Name: Auto Scaling, tag new instances

Review

Here we can see our selections. After completing the check, click Create Auto Scaling Group to complete the creation process.

So, we created Auto scaling group.

Checking Auto Scaling Activities - General


Try the new design for Amazon EC2 Auto Scaling

This older console is being replaced with the new EC2 Auto Scaling console. No new features or improvements will be made in this older console. Go to the new console.

Create Auto Scaling group
Actions
Filter: Filter Auto Scaling groups...
1 to 1 of 1 Auto Scaling Groups

Name	Instances	Desired	Min	Max	Default Cooldown	Health Check Grace Period
Auto Scaling	0	2	2	4	300	300

Auto Scaling Group: Auto Scaling

Launch Configuration	First-LC
Desired Capacity	2
Min	2
Max	4
Availability Zone(s)	eu-west-1b, eu-west-1c, eu-west-1a
Subnet(s)	subnet-9b58c8c1,subnet-882573ee,subnet-25a1876d
Classic Load Balancers	
Target Groups	Target-First
Health Check Type	ELB
Health Check Grace Period	300
Instance Protection	

Auto Scaling Menu

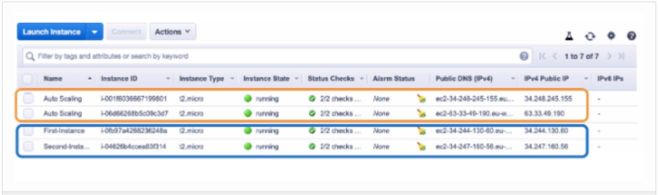
Let's go to the Auto Scaling menu and check the values.

Here we see; **Instance:** 0, **Desired:** 2, **Min:** 2 and **Max:** 4

It means there is no machine appearing in the system now. We desire 2 machines in ideal conditions and at least 2 machines always must be in use. The maximum number of instances that can be in use is 4.

After refreshing, 2 instances will be seen on the dashboard. Let's see the instances on the next page.

Auto Scaling Group Settings-1



Name	Instance ID	Instance Type	Instance State	Status Checks	Alarm Status	Public DNS (IPv4)	IPv4 Public IP	IPv6 IPs
Auto Scaling	i-001f6036667199801	t2.micro	running	2/2 checks ...	None	ec2-34-248-245-155.eu...	34.248.245.155	-
Auto Scaling	i-06d66268b5c09c3d7	t2.micro	running	2/2 checks ...	None	ec2-63-33-49-190.eu-w...	63.33.49.190	-
First-Instance	i-0fb97a4268236248a	t2.micro	running	2/2 checks ...	None	ec2-34-244-130-60.eu-...	34.244.130.60	-
Second-Insta...	i-04626b4ccea83f314	t2.micro	running	2/2 checks ...	None	ec2-34-247-160-56.eu-...	34.247.160.56	-

Instance Dashboard

Let's go to the Instance Dashboard and examine the number of instances.

If we look at the Instance dashboard shown in the picture above, we see that there are totally four instances in use. The top two are created by Auto Scaling, and their names are **Auto Scaling**. The remaining two instances belong to the Load Balancer.

In the beginning, as you remember, we had no machine appearing in the system, we desired 2 machines in ideal conditions and at least 2 machines would be in use all the time. The maximum number of instances that would be in use was considered as 4. (**Instance:** 0, **Desired:** 2, **Min:** 2 and **Max:** 4)

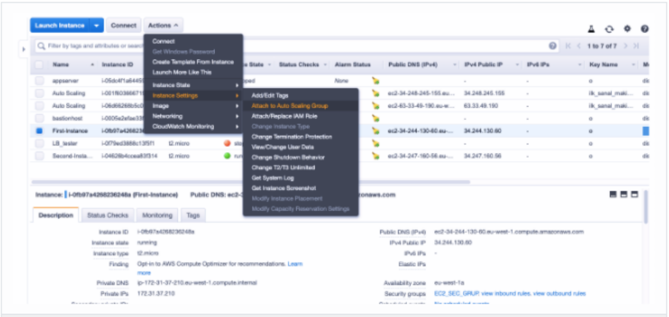
But now, there are 4 servers here. We had 2 servers already in the beginning, but Auto Scaling Group created 2 more servers.

Why Auto Scaling didn't consider the Load balancer's instances?

Auto Scaling Group Settings - 2

One of the rules is; **The Auto Scaling Group does not consider your current instance capacity**. So, even if you have 200 virtual machines in the environment, it still creates extra machines that you determine when configuring Auto Scaling.

Because the Auto Scaling Group doesn't know that these machines are built to do this job. So, we need to attach the machines to the Auto Scaling Group.



Instance: i-0fb97a4268236248a (First-Instance)

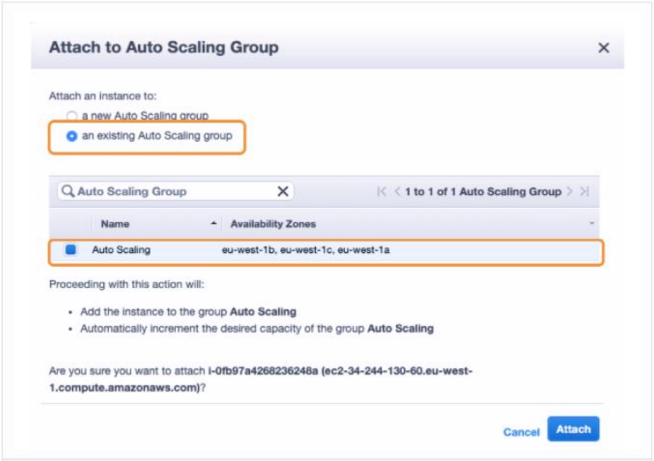
Instance ID	i-0fb97a4268236248a
Instance state	running
Instance type	t2.micro
Finding	Opt-in to AWS Compute Optimizer for recommendations. Learn more
Private DNS	ip-172-31-37-210.eu-west-1.compute.internal
Private IPs	172.31.37.210
Public DNS (IPv4)	ec2-34-244-130-60.eu-west-1.compute.amazonaws.com
IPv4 Public IP	34.244.130.60
IPv6 IPs	-
Availability zone	eu-west-1a
Security groups	EC2_SEC_GRUP. view inbound rules. view outbound rules

Actions Menu

After selecting the Instances menu, we check the instance we want to attach to ASG.

Then we select **Actions> Instance Settings> Attach to Auto Scaling Group** option as you see in the picture above.

After that, we check **an existing Auto Scaling Group** option and select Auto Scaling Group that we have just created. Then, complete the process by clicking **Attach**.



Attach to Auto Scaling Group

Attach an instance to:

- a new Auto Scaling group
- [✓ an existing Auto Scaling group]

1 to 1 of 1 Auto Scaling Group

Name	Availability Zones
Auto Scaling	eu-west-1b, eu-west-1c, eu-west-1a

Proceeding with this action will:

- Add the instance to the group **Auto Scaling**
- Automatically increment the desired capacity of the group **Auto Scaling**

Are you sure you want to attach **i-0fb97a4268236248a (ec2-34-244-130-60.eu-west-1.compute.amazonaws.com)**?

Attaching Instance

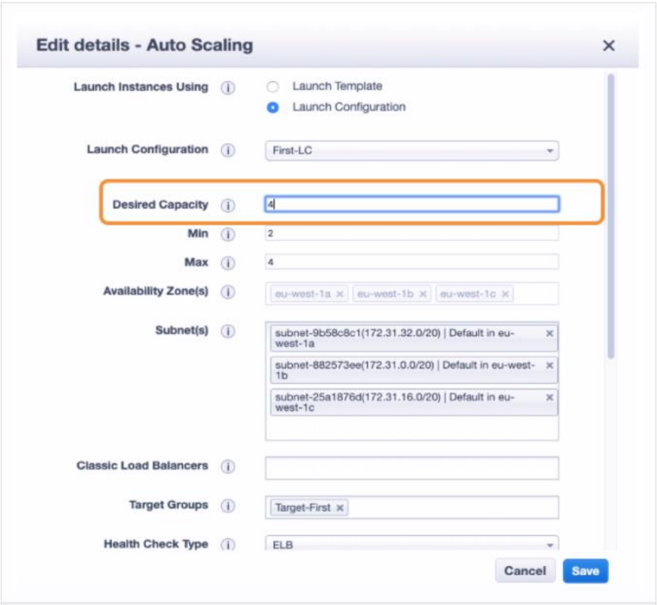
We repeat the same process for the other virtual machines.

Auto Scaling Group Settings - 3

Now come back to the Auto Scaling Group menu and control the instances. But this time we see Desired: 4, Min: 2, Max: 4.

Why the desired instance number appears 4, we originally determined it as 2.

Because the other rule for Auto Scaling is; **Manually added servers increase the number of desired servers that you determine before**.



Edit details - Auto Scaling

Launch Instances Using	Launch Template / [✓ Launch Configuration]
Launch Configuration	First-LC
Desired Capacity	4
Min	2
Max	4
Availability Zone(s)	eu-west-1a, eu-west-1b, eu-west-1c
Subnet(s)	subnet-9b58c8c1(172.31.32.0/20) | Default in eu-west-1a; subnet-882573ee(172.31.0.0/20) | Default in eu-west-1b; subnet-25a1876d(172.31.16.0/20) | Default in eu-west-1c
Classic Load Balancers	
Target Groups	Target-First
Health Check Type	ELB

Reducing the Number of Desired Instances

To fix this problem, choose your Auto Scaling Group by clicking the checkbox on the left, then select **Edit** from the **Actions** tab for correction as you see in the picture above. Then we reduce the desired capacity value to 2.

Now if you check the instance' status, you'll see that Auto Scaling will terminate two of its virtual machines.

Testing Auto Scaling

Now let's increase the CPU load of the machines to 30% and check whether the Auto Scaling Group is doing its job depending on this condition.

For this, there is an application called Stress, which is used in Ubuntu machines to increase CPU usage of the machines.

- First connect to ec2 with Terminal
- Let's get root privilege with this command seen below

```
sudo su
```

- Then install stress application with the following commands and increase CPU usage.

```
yum install https://dl.fedoraproject.org/pub/epel/epel-release-latest-7.noarch.
yum install stress
stress --cpu 80 --timeout 2000
```

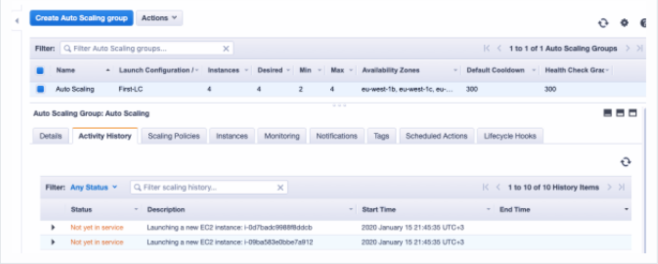
- Thus stress application started to increase CPU for this machine.
- We can do the same process in the other running machine. The CPU increase will gradually increase.

Let's wait for a while then see the result.

Auto Scaling Test Result

Let's see how Auto Scaling responded to the requests. So click **Auto Scaling Group** on the left-hand menu.

- If you select **Activity History** tab you'll see Auto Scaling Group starts to create new instances as you see in the picture below.



1 to 1 of 1 Auto Scaling Groups

Name	Launch Configuration	Instances	Desired	Min	Max	Availability Zones	Default Cooldown	Health Check Grac
Auto Scaling	First-LC	4	4	2	4	eu-west-1b, eu-west-1c, eu-...	300	300

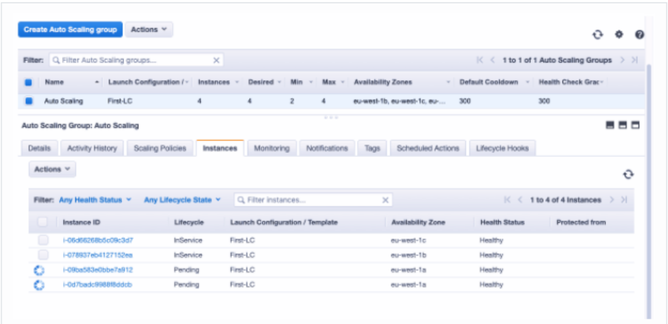
Auto Scaling Group: Auto Scaling

Filter: Any Status — 1 to 10 of 10 History Items

Status	Description	Start Time	End Time
Not yet in service	Launching a new EC2 instance: i-0d7badc9988f8ddcb	2020 January 15 21:45:35 UTC+3	
Not yet in service	Launching a new EC2 instance: i-09ba583e0bbe7a912	2020 January 15 21:45:35 UTC+3	

Activity History

- You can also see if you click **Instance** tab that 2 machines are in **pending** status as seen in the picture below.


Create Auto Scaling group
Actions
Filter: Filter Auto Scaling groups...
1 to 1 of 1 Auto Scaling Groups

Name	Launch Configuration /	Instances	Desired	Min	Max	Availability Zones	Default Cooldown	Health Check Grac
Auto Scaling	First-LC	4	4	2	4	eu-west-1b, eu-west-1c, eu-...	300	300

Auto Scaling Group: Auto Scaling

Instance ID	Lifecycle	Launch Configuration / Template	Availability Zone	Health Status	Protected from
i-06d66268b5c09c3d7	InService	First-LC	eu-west-1c	Healthy	
i-078937eb4127152ea	InService	First-LC	eu-west-1b	Healthy	
i-09ba583e0bbe7a912	Pending	First-LC	eu-west-1a	Healthy	
i-0d7badc9988f8ddcb	Pending	First-LC	eu-west-1a	Healthy	

Instance

So, as a result, when the CPU usage increased, our Auto Scaling started to create new instances to handle the request.

Congratulations! It works.