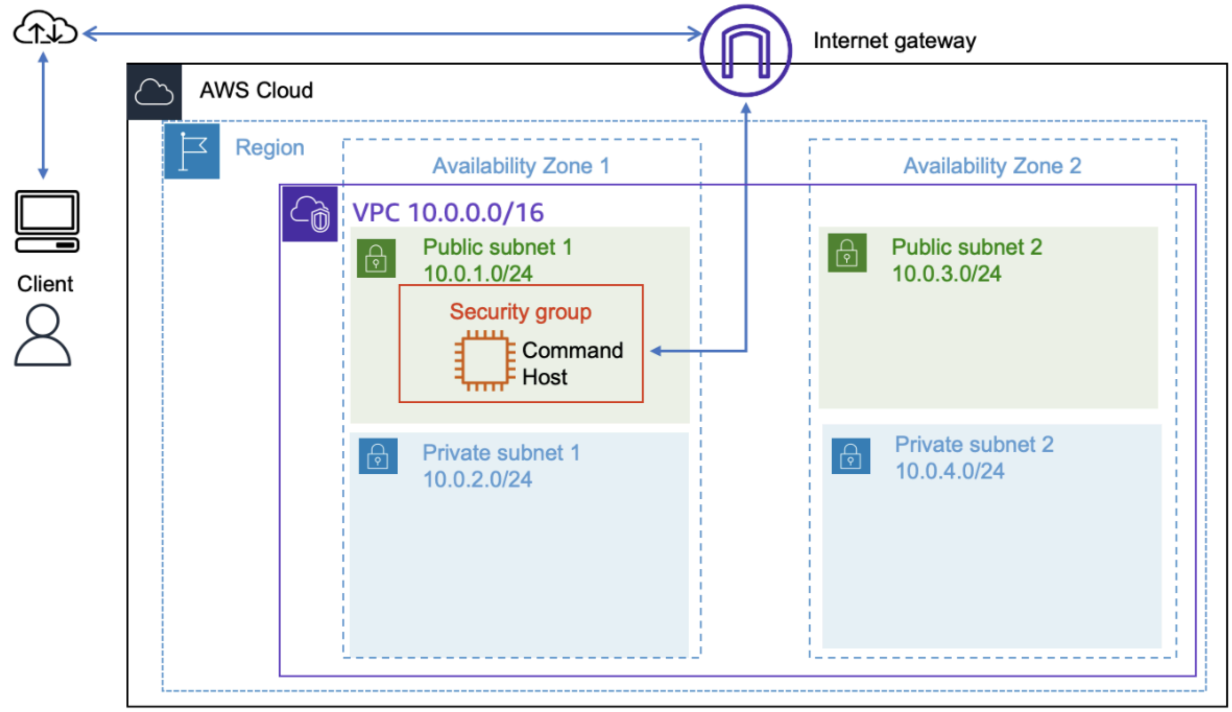
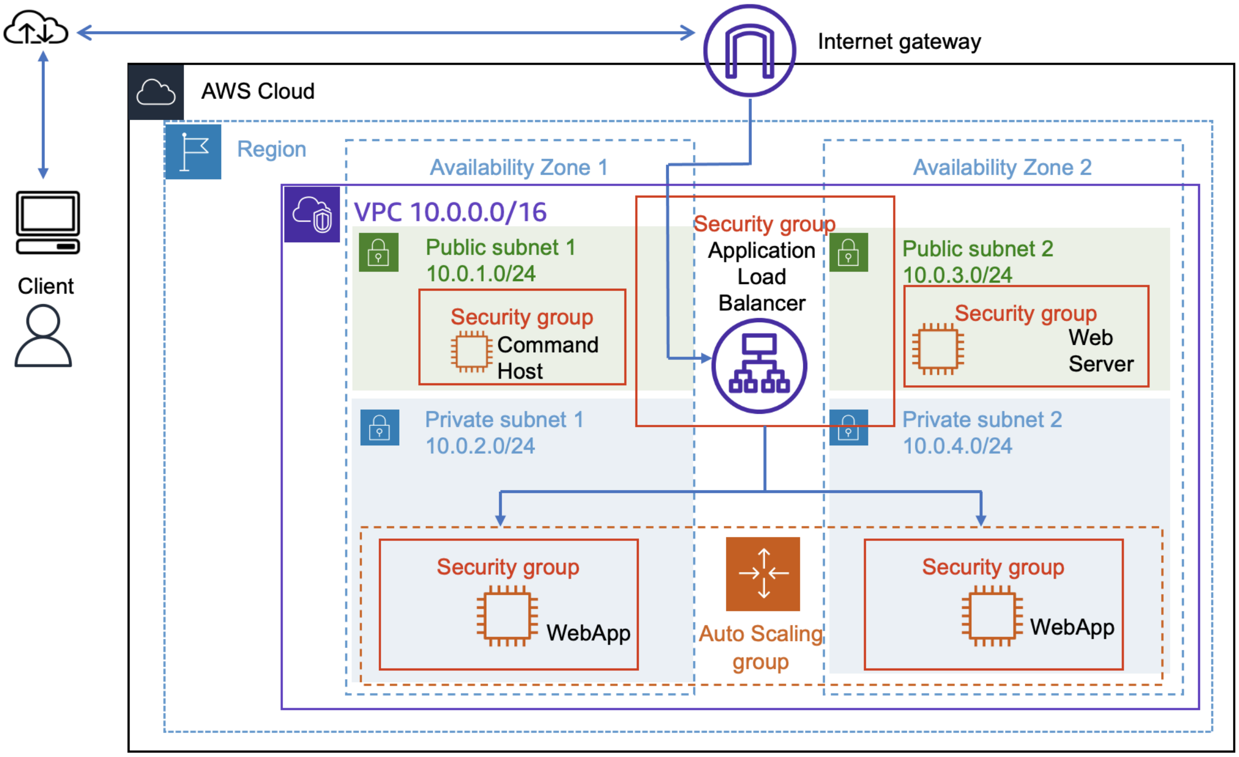
Using Auto Scaling in AWS (Linux)

Lab overview

In this lab, you use the AWS Command Line Interface (AWS CLI) to create an Amazon Elastic Compute Cloud (EC2) instance to host a web server and create an Amazon Machine Image (AMI) from that instance. You then use that AMI as the basis for launching a system that scales automatically under a variable load by using Amazon EC2 Auto Scaling. You also create an Elastic Load Balancer to distribute the load across EC2 instances created in multiple Availability Zones by the auto scaling configuration.


Internet gateway
AWS Cloud
Region
Availability Zone 1
Availability Zone 2
VPC 10.0.0.0/16
Public subnet 1
10.0.1.0/24
Security group
Command Host
Public subnet 2
10.0.3.0/24
Private subnet 1
10.0.2.0/24
Private subnet 2
10.0.4.0/24
Client


Internet gateway
AWS Cloud
Region
Availability Zone 1
Availability Zone 2
VPC 10.0.0.0/16
Public subnet 1
10.0.1.0/24
Security group
Command Host
Security group
Application Load Balancer
Public subnet 2
10.0.3.0/24
Security group
Web Server
Private subnet 1
10.0.2.0/24
Private subnet 2
10.0.4.0/24
Security group
WebApp
Auto Scaling group
Security group
WebApp
Client

Objectives

After completing this lab, you will be able to do the following:

- Create an EC2 instance by using an AWS CLI command.
- Create a new AMI by using the AWS CLI.
- Create an Amazon EC2 launch template.
- Create an Amazon EC2 Auto Scaling launch configuration.
- Configure scaling policies and create an Auto Scaling group to scale in and scale out the number of servers based on a variable load.

Duration

This lab requires approximately **45 minutes** to complete.

Accessing the AWS Management Console

1. At the top of these instructions, choose `Start Lab` to launch the lab.
   A **Start Lab** panel opens displaying the lab status.
2. Wait until the message "Lab status: ready" appears, and then choose **X** to close the **Start Lab** panel.
3. At the top of these instructions, choose `AWS` to open the AWS Management Console on a new browser tab. The system automatically signs you in.
   Tip If a new browser tab does not open, a banner or icon at the top of your browser will indicate that your browser is preventing the site from opening pop-up windows. Choose the banner or icon, and choose **Allow pop-ups**.
4. Arrange the AWS Management Console so that it appears alongside these instructions. Ideally, you should be able to see both browser tabs at the same time to follow the lab steps.
   Important: Do not change the lab Region unless specifically instructed to do so.

Task 1: Creating a new AMI for Amazon EC2 Auto Scaling

In this task, you launch a new EC2 instance and then create a new AMI based on that running instance. You use the AWS CLI on the Command Host EC2 instance to perform all of these operations.

Task 1.1: Connecting to the Command Host instance

In this task, you use EC2 Instance Connect to connect to the Command Host EC2 instance that was created when the lab was provisioned. You use this instance to run AWS CLI commands.

5. On the **AWS Management Console**, in the **Search** bar, enter and choose `EC2` to open the **EC2 Management Console**.
6. In the navigation pane, choose **Instances**.
7. From the list of instances, select the **Command Host** instance.
8. Choose **Connect**.
9. On the **EC2 Instance Connect** tab, choose **Connect**.
   Note: If you prefer to use an SSH client to connect to the EC2 instance, see the guidance to Connect to Your Linux Instance.

Now that you are connected to the Command Host instance, you can configure and use the AWS CLI to call AWS services.

Task 1.2: Configuring the AWS CLI

The AWS CLI is preconfigured on the Command Host instance.

10. To confirm that the Region in which the Command Host instance is running is the same as the lab (the us-west-2 Region), run the following command:
11.
    ```
    curl http://169.254.169.254/latest/dynamic/instance-identity/document | grep region
    ```
12. You use this Region information in the next steps.

13. To update the AWS CLI software with the correct credentials, run the following command:
14. `aws configure`
15. At the prompts, enter the following information:
    - **AWS Access Key ID**: Press Enter.
    - **AWS Secret Access Key**: Press Enter.
    - **Default region name**: Enter the name of the Region from the previous steps in this task (for example, `us-west-2`). If the Region is already displayed, press Enter.
    - **Default output format**: Enter `json`

Now you are ready to access and run the scripts detailed in the following steps.

13. To access these scripts, enter the following command to navigate to their directory:
14. `cd /home/ec2-user/`

Task 1.3: Creating a new EC2 Instance

In this task, you use the AWS CLI to create a new instance that hosts a web server.

14. To inspect the UserData.txt script that was installed for you as part of the Command Host creation, run the following command:
15. `more UserData.txt`
16. This script performs a number of initialization tasks, including updating all installed software on the box and installing a small PHP web application that you can use to simulate a high CPU load on the instance. The following lines appear near the end of the script:
17.
```
find -wholename /root/.*history -wholename /home/*/.*history -exec rm -f {} \;
find / -name 'authorized_keys' -exec rm -f {} \;
rm -rf /var/lib/cloud/data/scripts/*
```
18. These lines erase any history or security information that might have accidentally been left on the instance when the image was taken.
19. At the top of this page, choose **Details**, and choose **Show**.
20. Copy the **KEYNAME**, **AMIID**, **HTTPACCESS**, and **SUBNETID** values into a text editor document, and then choose **X** to close the **Credentials** panel.
21. In the following script, replace the corresponding text with the values from the previous step.

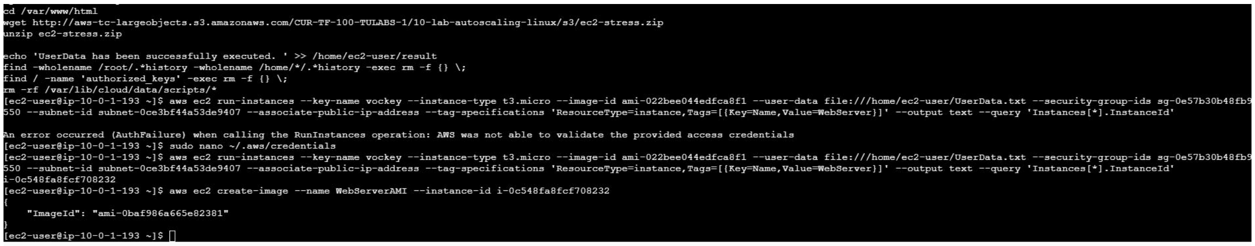
22.
    ```
    aws ec2 run-instances --key-name KEYNAME --instance-type
    t3.micro --image-id AMIID --user-data
    file:///home/ec2-user/UserData.txt --security-group-ids
    HTTPACCESS --subnet-id SUBNETID --associate-public-ip-address
    --tag-specifications
    'ResourceType=instance,Tags=[{Key=Name,Value=WebServer}]'
    --output text --query 'Instances[*].InstanceId'
    ```
23. Enter your modified script into the terminal window, and run the script.
    The output of this command provides you with an **InstanceId**. Subsequent steps in this lab refer to this value as **NEW-INSTANCE-ID**. Replace this value as needed throughout this lab.
24. Copy and paste the **InstanceId** value into a text editor to use later.
25. To use the aws ec2 wait instance-running command to monitor this instance's status, replace *NEW-INSTANCE-ID* in the following command with the **InstanceID** value that you copied in the previous step. Run your modified command.
26.
    ```
    aws ec2 wait instance-running --instance-ids NEW-INSTANCE-ID
    ```
27. Wait for the command to return to a prompt before proceeding.
    Your instance starts a new web server. To test that the web server was installed properly, you must obtain the public DNS name.
28. To obtain the public DNS name, in the following command, replace *NEW-INSTANCE-ID* with the value that you copied previously, and run your modified command:
29.
    ```
    aws ec2 describe-instances --instance-id NEW-INSTANCE-ID
    --query
    'Reservations[0].Instances[0].NetworkInterfaces[0].Association.
    PublicDnsName'
    ```
30. Copy the output of this command without the quotation marks.
    The value of this output is referred to as **PUBLIC-DNS-ADDRESS** in the next steps.
31. In a new browser tab, enter the output that you copied from the previous step.
    It could take a few minutes for the web server to be installed. Wait 5 minutes before continuing to the next steps.
    Do not choose **Start Stress** at this stage.
32. In the following command, replace *PUBLIC-DNS-ADDRESS* with the value that you copied in the previous steps, and then run your modified command.
33.
    ```
    http://PUBLIC-DNS-ADDRESS/index.php
    ```
34. If your web server does not appear to be running, check with your instructor.

Task 1.4: Creating a Custom AMI

In this task, you create a new AMI based on that instance that you just created.

25. To create a new AMI based on this instance, in the following aws ec2 create-image command, replace *NEW-INSTANCE-ID* with the value that you copied previously, and run your adjusted command:
26. `aws ec2 create-image --name WebServerAMI --instance-id NEW-INSTANCE-ID`

By default, the aws ec2 create-image command restarts the current instance before creating the AMI to ensure the integrity of the image on the file system. While your AMI is being created, proceed to the next task.



```
cd /var/www/html
wget http://aws-tc-largeobjects.s3.amazonaws.com/CUR-TF-100-TULABS-1/10-lab-autoscaling-linux/s3/ec2-stress.zip
unzip ec2-stress.zip

echo 'UserData has been successfully executed. ' >> /home/ec2-user/result
find -wholename /root/.*history -wholename /home/*/.*history -exec rm -f {} \;
find / -name 'authorized_keys' -exec rm -f {} \;
rm -rf /var/lib/cloud/data/scripts/*
[ec2-user@ip-10-0-1-193 ~]$ aws ec2 run-instances --key-name vockey --instance-type t3.micro --image-id ami-022bee044edfca8f1 --user-data file:///home/ec2-user/UserData.txt --security-group-ids sg-0e57b30b48fb9
550 --subnet-id subnet-0ce3bf44a53de9407 --associate-public-ip-address --tag-specifications 'ResourceType=instance,Tags=[{Key=Name,Value=WebServer}]' --output text --query 'Instances[*].InstanceId'

An error occurred (AuthFailure) when calling the RunInstances operation: AWS was not able to validate the provided access credentials
[ec2-user@ip-10-0-1-193 ~]$ sudo nano ~/.aws/credentials
[ec2-user@ip-10-0-1-193 ~]$ aws ec2 run-instances --key-name vockey --instance-type t3.micro --image-id ami-022bee044edfca8f1 --user-data file:///home/ec2-user/UserData.txt --security-group-ids sg-0e57b30b48fb9
550 --subnet-id subnet-0ce3bf44a53de9407 --associate-public-ip-address --tag-specifications 'ResourceType=instance,Tags=[{Key=Name,Value=WebServer}]' --output text --query 'Instances[*].InstanceId'
i-0c548fa8fcf708232
[ec2-user@ip-10-0-1-193 ~]$ aws ec2 create-image --name WebServerAMI --instance-id i-0c548fa8fcf708232
{
    "ImageId": "ami-0baf986a665e82381"
}
[ec2-user@ip-10-0-1-193 ~]$ 
```

Task 2: Creating an auto scaling environment

In this section, you create a load balancer that pools a group of EC2 instances under a single Domain Name System (DNS) address. You use auto scaling to create a dynamically scalable pool of EC2 instances based on the image that you created in the previous task. Finally, you create a set of alarms that scale out or scale in the number of instances in your load balancer group whenever the CPU performance of any machine within the group exceeds or falls below a set of specified thresholds.

You can perform the following task by using either the AWS CLI or the AWS Management Console. For this lab, you use the AWS Management Console.

Task 2.1: Creating an Application Load Balancer

In this task, you create a load balancer that can balance traffic across multiple EC2 instances and Availability Zones.

26. On the EC2 Management Console, in the left navigation pane, locate the **Load Balancing** section, and choose **Load Balancers**.
27. Choose **Create load balancer**.
28. In the **Load balancer types** section, for **Application Load Balancer**, choose **Create**.
29. On the **Create Application Load Balancer** page, in the **Basic configuration** section, configure the following option:
    - For **Load balancer name**, enter `WebServerELB`
30. In the **Network mapping** section, configure the following options:
    - For **VPC**, choose **Lab VPC**.
    - For **Mappings**, choose both Availability Zones listed.
        - For the first Availability Zone, choose **Public Subnet 1**.
        - For the second Availability Zone, choose **Public Subnet 2**.
    - These options configure the load balancer to operate across multiple Availability Zones.
31. In the **Security groups** section, choose the **X** for the **default** security group to remove it.
32. From the **Security groups** dropdown list, choose **HTTPAccess**.
    The **HTTPAccess** security group has already been created for you, which permits HTTP access.
33. In the **Listeners and routing** section, choose the **Create target group** link.
    Note: This link opens a new browser tab with the **Create target group** configuration options.
34. On the **Specify group details** page, in the **Basic configuration** section, configure the following options:
    - For **Choose a target type**, choose **Instances**.
    - For **Target group name**, enter `webserver-app`
35. In the **Health checks** section, for **Health check path**, enter `/index.php`
36. At the bottom of the page, choose **Next**.
37. On the **Register targets** page, choose **Create target group**.
    Once the target group has been created successfully, close the **Target groups** browser tab.
38. Return to the **Load balancers** browser tab, and locate the **Listeners and routing** section. For **Default action**, choose **Refresh** to the right of the **Forward to** dropdown list.
39. From the **Forward to** dropdown list, choose **webserver-app**.
40. At the bottom of the page, choose **Create load balancer**.

You should receive a message similar to the following:

Successfully created load balancer: WebServerELB

41. To view the **WebServerELB** load balancer that you created, choose **View load balancer**.
42. To copy the **DNS name** of the load balancer, use the copy option , and paste the DNS name into a text editor.

You need this information later in the lab.

Task 2.2: Creating a launch template

In this task, you create a launch template for your Auto Scaling group. A launch template is a template that an Auto Scaling group uses to launch EC2 instances. When you create a launch template, you specify information for the instances, such as the AMI, instance type, key pair, security group, and disks.

43. On the EC2 Management Console, in the left navigation pane, locate the **Instances** section, and choose **Launch Templates**.
44. Choose **Create launch template**.
45. On the **Create launch template** page, in the **Launch template name and description** section, configure the following options:
    - For **Launch template name - *required***, enter `web-app-launch-template`
    - For **Template version description**, enter `A web server for the load test app`
    - For **Auto Scaling guidance**, select **Provide guidance to help me set up a template that I can use with EC2 Auto Scaling**.
46. In the **Application and OS Images (Amazon Machine Image) - required** section, choose the **My AMIs** tab.
    Notice that **WebServerAMI** is already chosen.
47. In the **Instance type** section, choose the **Instance type** dropdown list, and choose **t3.micro**.
48. In the **Key pair (login)** section, confirm that the **Key pair name** dropdown list is set to **Don't include in launch template**.
    Amazon EC2 uses public key cryptography to encrypt and decrypt login information. To log in to your instance, you must create a key pair, specify the name of the key pair when you launch the instance, and provide the private key when you connect to the instance.

    Note: In this lab, you do not need to connect to the instance.

49. In the **Network settings** section, choose the **Security groups** dropdown list, and choose **HTTPAccess**.
    When you launch an instance, you can pass user data to the instance. The data can be used to run configuration tasks and scripts.
50. Choose **Create launch template**.
    You should receive a message similar to the following:

    > Successfully created web-app-launch-template.

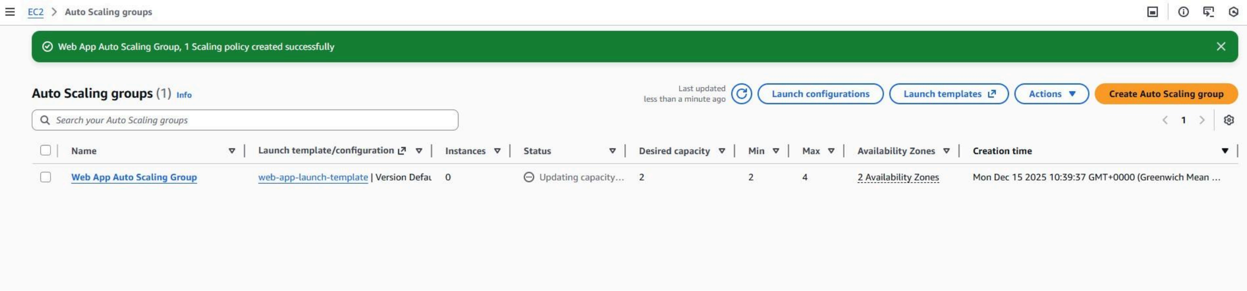
51. Choose **View launch templates**.

Task 2.3: Creating an Auto Scaling group

In this task, you use your launch template to create an Auto Scaling group.

52. Choose **web-app-launch-template**, and then from the **Actions** dropdown list, choose **Create Auto Scaling group**.
53. On the **Choose launch template or configuration** page, in the **Name** section, for **Auto Scaling group name**, enter `Web App Auto Scaling Group`
54. Choose **Next**.
55. On the **Choose instance launch options** page, in the **Network** section, configure the following options:
    - From the **VPC** dropdown list, choose **Lab VPC**.
    - From the **Availability Zones and subnets** dropdown list, choose **Private Subnet 1 (10.0.2.0/24)** and **Private Subnet 2 (10.0.4.0/24)**.
56. Choose **Next**.
57. On the **Configure advanced options – *optional*** page, configure the following options:
    - In the **Load balancing – *optional*** section, choose **Attach to an existing load balancer**.
    - In the **Attach to an existing load balancer** section, configure the following options:
        - Choose **Choose from your load balancer target groups**.
        - From the **Existing load balancer target groups** dropdown list, choose **webserver-app | HTTP**.
    - In the **Health checks** section, under **Additional health check types**, select **Turn on Elastic Load Balancing health checks**.

58. Choose **Next**.
59. On the **Configure group size and scaling policies – *optional*** page, configure the following options:
    - In the **Group size – *optional*** section, enter the following values:
        - **Desired capacity**:`2`
        - **Minimum capacity**: `2`
        - **Maximum capacity**: `4`
    - In the **Scaling policies – *optional*** section, configure the following options:
        - Choose **Target tracking scaling policy**.
        - For **Metric type**, choose **Average CPU utilization**.
        - For **Target value**, enter `50`
    - This change tells auto scaling to maintain an average CPU utilization across all instances of 50 percent. Auto scaling automatically adds or removes capacity as required to keep the metric at or close to the specified target value. It adjusts to fluctuations in the metric due to a fluctuating load pattern.
60. Choose **Next**.
61. On the **Add notifications – *optional*** page, choose **Next**.
62. On the **Add tags – *optional*** page, choose **Add tag** and configure the following options:
    - For **Key**, enter `Name`
    - For **Value - optional**, enter `WebApp`
63. Choose **Next**.
64. On the **Review** page, choose **Create Auto Scaling group**.
    These options launch EC2 instances in private subnets across both Availability Zones.
    Your Auto Scaling group initially shows an **Instances** count of zero, but new instances will be launched to reach the desired count of two instances.
    Note: If you experience an error related to the t3.micro instance type not being available, then rerun this task by choosing the t2.micro instance type instead.


EC2 > Auto Scaling groups
Auto Scaling groups (1) Info
Last updated less than a minute ago
Launch configurations
Launch templates
Actions ▼
Create Auto Scaling group
Search your Auto Scaling groups
1

Web App Auto Scaling Group, 1 Scaling policy created successfully

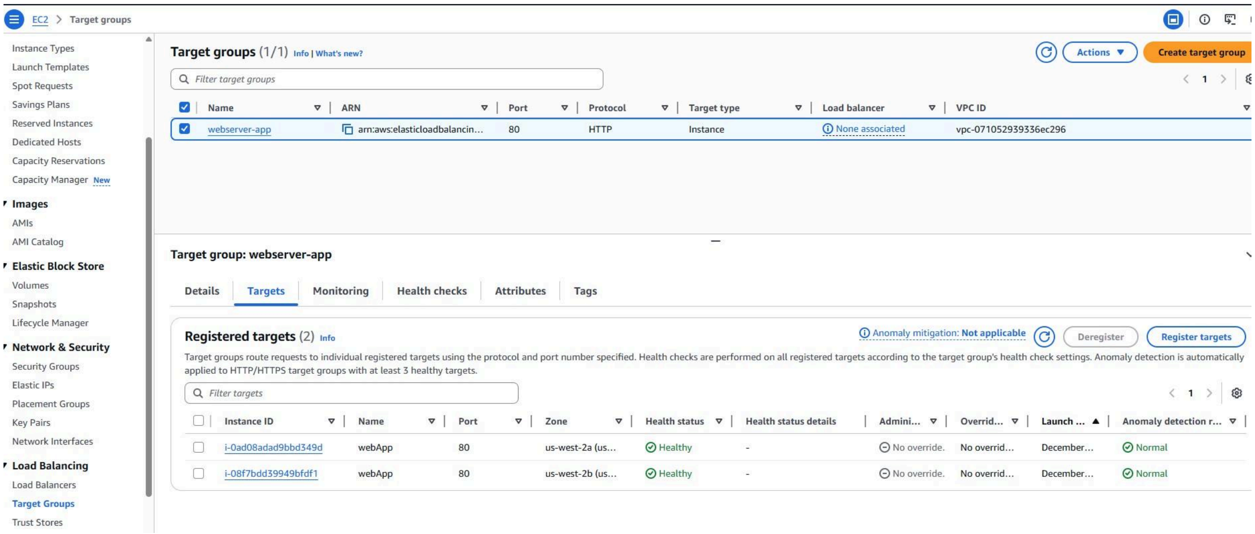
	Name	Launch template/configuration	Instances	Status	Desired capacity	Min	Max	Availability Zones	Creation time
□	Web App Auto Scaling Group	web-app-launch-template | Version Defau	0	Updating capacity...	2	2	4	2 Availability Zones	Mon Dec 15 2025 10:39:37 GMT+0000 (Greenwich Mean ...

Task 3: Verifying the auto scaling configuration

In this task, you verify that both the auto scaling configuration and the load balancer are working by accessing a pre-installed script on one of your servers that will consume CPU cycles, which invokes the scale out alarm.

65. In the left navigation pane, choose **Instances**.
    Two new instances named **WebApp** are being created as part of your Auto Scaling group. While these instances are being created, the **Status check** for these two instances is *Initializing*.
    Observe the **Status check** field for the instances until the status is *2/2 checks passed*. Wait for the two new instances to complete initialization before you proceed to the next step.
    You might need to choose **Refresh** to see the updated status.
66. Once the instances have completed initialization, in the left navigation pane in the **Load Balancing** section, choose **Target Groups**, and then select your target group, **webserver-app**.
67. On the **Targets** tab, verify that two instances are being created. Refresh this list until the **Health status** of these instances changes to *healthy*.

You can now test the web application by accessing it through the load balancer.



Target groups (1/1)

☑	Name	ARN	Port	Protocol	Target type	Load balancer	VPC ID
☑	webserver-app	arn:aws:elasticloadbalancin...	80	HTTP	Instance	None associated	vpc-071052939336ec296

Target group: webserver-app

Registered targets (2)

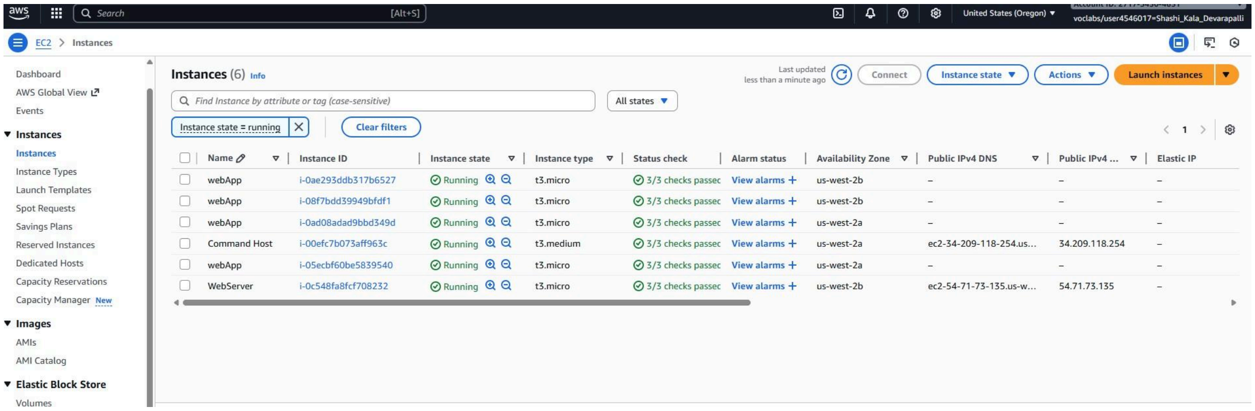
Target groups route requests to individual registered targets using the protocol and port number specified. Health checks are performed on all registered targets according to the target group's health check settings. Anomaly detection is automatically applied to HTTP/HTTPS target groups with at least 3 healthy targets.

□	Instance ID	Name	Port	Zone	Health status	Health status details	Admini...	Overrid...	Launch ...	Anomaly detection r...
□	i-0ad08adad9bbd349d	webApp	80	us-west-2a (us...	Healthy	-	No override.	No overrid...	December...	Normal
□	i-08f7bdd39949bfdf1	webApp	80	us-west-2b (us...	Healthy	-	No override.	No overrid...	December...	Normal

Task 4: Testing auto scaling configuration

68. Open a new web browser tab, and paste the **DNS name** of the load balancer that you copied earlier into the address bar, and press Enter.

69. On the web page, choose **Start Stress**.
    This step calls the application **stress** in the background, which causes the CPU utilization on the instance that serviced this request to spike to 100 percent.
70. On the EC2 Management console, in the left navigation pane in the **Auto Scaling** section, choose **Auto Scaling Groups**.
71. Select **Web App Auto Scaling Group**.
72. Choose the **Activity** tab.
    After a few minutes, you should see your Auto Scaling group add a new instance. This occurs because Amazon CloudWatch detected that the average CPU utilization of your Auto Scaling group exceeded 50 percent, and your scale-up policy has been invoked in response.
    You can also check the new instances being launched on the EC2 Dashboard.



Instances (6)

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4 ...	Elastic IP
webApp	i-0ae293ddb317b6527	Running	t3.micro	3/3 checks passec	View alarms +	us-west-2b	–	–	–
webApp	i-08f7bdd39949bfdf1	Running	t3.micro	3/3 checks passec	View alarms +	us-west-2b	–	–	–
webApp	i-0ad08adad9bbd349d	Running	t3.micro	3/3 checks passec	View alarms +	us-west-2a	–	–	–
Command Host	i-00efc7b073aff963c	Running	t3.medium	3/3 checks passec	View alarms +	us-west-2a	ec2-34-209-118-254.us...	34.209.118.254	–
webApp	i-05ecbf60be5839540	Running	t3.micro	3/3 checks passec	View alarms +	us-west-2a	–	–	–
WebServer	i-0c548fa8fcf708232	Running	t3.micro	3/3 checks passec	View alarms +	us-west-2b	ec2-54-71-73-135.us-w...	54.71.73.135	–