

Lab 3W - Using Auto Scaling (Windows)

Monday, April 20, 2020 3:00 PM

Systems Operations on AWS - Lab 3W - Using Auto Scaling (Windows)

3 hours

Free

★★★★★ [Rate Lab](#)



© 2020 Amazon Web Services, Inc. and its affiliates. All rights reserved.

This work may not be reproduced or redistributed, in whole or in part, without prior written permission from Amazon Web Services, Inc.

Commercial copying, lending, or selling is prohibited.

Errors or corrections? Email us at aws-course-feedback@amazon.com.

Other questions? Contact us at <https://aws.amazon.com/contact-us/aws-training/>

In this lab, you will be tasked with creating a new Amazon Machine Image (AMI) from an existing EC2 instance, and use that machine as the basis for defining a system that will scale automatically under increasing loads.

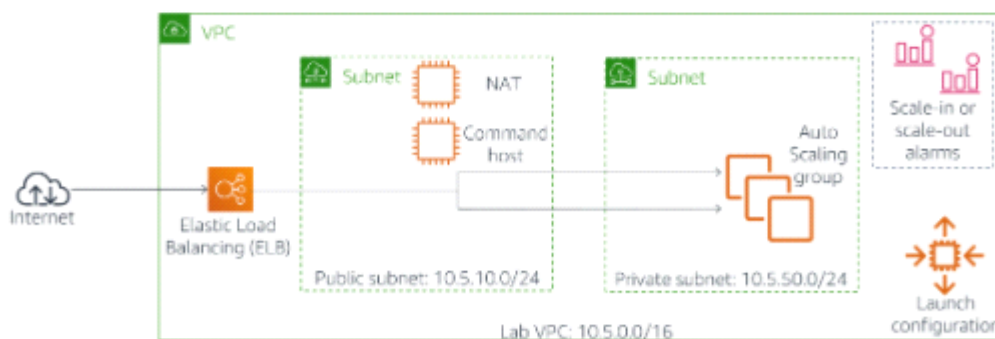
Duration

This lab will require approximately **45 minutes** to complete.

⚠ This is the Windows version of this lab. It means that you will be connecting to an Amazon EC2 Windows instance via RDP. If you instead wish to use SSH to connect to an Amazon EC2 Linux instance, **please use the Linux version of the lab.**

Scenario

In this lab, you will create the scalable web server system shown in the following diagram:



Objectives

After completing this lab, you will be able to:

- Create a new Amazon Machine Image (AMI) by using the Amazon Command Line Interface (CLI).

- Create a new Amazon Machine Image (AMI) by using the Amazon Command Line Interface (CLI).
- Use Auto Scaling to scale up the number of servers available for a specific task when other servers are experiencing heavy load.

The following components are created for you as a part of the lab environment:

- Amazon VPC
- Public Subnets
- Private Subnets
- Amazon EC2 - Command Host (*in the public subnet*), you will log in to this instance to create a few of your AWS assets.

You will create the following components for this lab:

- Amazon EC2 - Web Server
- Amazon Machine Image (AMI)
- Auto Scaling Launch Configuration
- Auto Scaling Group
- Auto Scaling Policies
- ELB



Accessing the AWS Management Console

Start Lab

1. At the top of your screen, launch your lab by clicking [Start Lab](#)

This will start the process of provisioning your lab resources. An estimated amount of time to provision your lab resources will be displayed. You must wait for your resources to be provisioned before continuing.

i If you are prompted for a token, use the one distributed to you (or credits you have purchased).

2. Open your lab by clicking [Open Console](#)

This will open an AWS Management Console sign-in page.

3. On the Sign-in page, configure:

- **IAM user name:** `awsstudent`
- **Password:** Paste the value of **Password** located to the left of these instructions.
- Click [Sign In](#)

⚠ Please do not change the Region unless instructed.

⚠ Please do not change the Region unless instructed.

Common login errors

Error: You must first log out

Amazon Web Services Sign In

You must first log out before logging into a different AWS account.

To logout, [click here](#)

If you see the message, **You must first log out before logging into a different AWS account:**

- Click **click here**
- Close your browser tab to return to your initial Qwiklabs window
- Click [Open Console](#) again

Task 1: Create a New Amazon Machine Image for Amazon EC2 Auto Scaling

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

the Command Host to perform all of these operations.

Connect to the Windows Server Command Host from a Windows Machine

In this procedure, you will connect to the instance, from a Windows machine using Microsoft Remote Desktop Connection.

If you are using Mac or Linux, [skip to the next section](#).

4. To the left of the instructions you are currently reading, click **Download PEM**.
5. In the **AWS Management Console**, on the **Services** menu, click **EC2**.
6. In the left navigation pane, click **Instances**.
7. Right-click on **Command Host** instance to which you will connect, and select **Get Windows Password**.
8. On the **Retrieve Default Windows Administrator Password** dialog, next to **Key Pair Path**, click **Choose File**, and select the **PEM file** that you downloaded earlier.
9. Click **Decrypt Password**

This will give you the password that you need to log into this Windows instance.

10. Open the Remote Desktop Connection application on your computer.
 - On Windows 7, click the **Start** icon, and in the **Search programs and files** textbox, type **Remote Desktop Connection**. Click on the application when it appears in the Programs list.
 - On Windows 8 and 10, activate the Charms menu by moving the cursor into the lower right corner of the screen, and click the **Search**

- On Windows 8 and 10, activate the Charms menu by moving the cursor into the lower right corner of the screen, and click the **Search** icon. Type in `Remote Desktop Connection`. Click the application when it appears in the **Programs** list.

11. In Remote Desktop Connection, for **Computer** field, paste the Public IP \ Public DNS address of your Windows instance.

12. Click **Connect**.

13. Remote Desktop Connection will prompt you with a Login dialog asking for your username and password. By default, the application will use your current Windows username and domain. To change this, click **Use another account**.

14. For your login credentials, use the following values:

- **User name:** `\Administrator`
- **Password:** Enter the password obtained using the **Get Windows Password** feature.

⚠ Copying and pasting the Windows password from the **Retrieve Default Windows Administrator Password** window sometimes results in a failed login attempt, as it picks up extraneous spaces before or after the password. If your password is rejected, try typing the password manually rather than copying and pasting.

The \ in the user name is important, as it tells Remote Desktop Connection that you are logging in as the local Administrator, and not as a domain user.

15. To connect to your instance, click **OK**.

If you receive a prompt that the certificate used to verify the connection was not a known, trusted root certificate, click **Yes**.

Result: Your connection to your remote instance should start momentarily.

Result: Your connection to your remote instance should start momentarily.

16. [Windows Users: Click here to skip ahead to the next task.](#)

Connect to the Windows Server Command Host from a macOS Machine

These instructions are for Mac/Linux users only. If you are a Windows user, [skip to the next task.](#)

17. To the left of the instructions you are currently reading, click **Download PEM**.
18. In the **AWS Management Console**, on the **Services** menu, click **EC2**.
19. In the left navigation pane, click **Instances**.
20. Right-click on **Command Host** instance to which you will connect, and select **Get Windows Password**.
21. On the **Retrieve Default Windows Administrator Password** dialog, next to **Key Pair Path**, click **Choose File**, and select the **PEM file** that you downloaded earlier.
22. Click **Decrypt Password**.

This will give you the password that you need to log into this Windows instance.

23. From the Dock, launch the **App store**.
24. Search for the following string: **Microsoft Remote Desktop**
25. Click **Install**.

25. Click **Install**.
26. On the Dock, click **Launchpad**.
27. From **Launchpad**, click **Microsoft Remote Desktop**.
28. To create a new connection, click **New**.
29. To create a connection as Administrator on the remote system, use the following values:
 - **Connection name:** Choose a name
 - **PC name:** Paste in the Public IP \ Public DNS address of your Windows Server instance
 - **User name:** \Administrator
 - **Password:** Paste in the password obtained using the **Get Windows Password** feature.

The \ in the user name is important, as it tells the Remote Desktop application that you are logging in as the local Administrator, and not as a domain user.

30. To close the **New Connection** dialog box, click the **red X** icon.
31. Select your connection in the Microsoft Remote Desktop window, and click **Start**.

If you receive a prompt that the certificate used to verify the connection was not a known, trusted root certificate, click **Continue**.

Result: Your connection to your remote instance should start momentarily.

Create a New EC2 Instance

Create a New EC2 Instance

Now that you are logged in to Command Host, you will use the AWS CLI to create a new instance that hosts a web server.

32. Click the **Windows Start** button and then click on the **Windows PowerShell** icon.

33. Use the type command in PowerShell to inspect the script `c:\temp\UserData.txt` that was installed for you as part of the Command Host

```
type c:\temp\UserData.txt
```

This PowerShell script performs a number of initialization tasks, including updating all installed software on the box and installing a small ASP.NET web application that will enable you to simulate a high CPU load on the instance.

34. Copy the **KEYNAME**, **AMIID**, **HTTPACCESS** and **SUBNETID** values shown to the left of the instructions you are currently reading and replace it into relevant sections of the below script.

```
aws ec2 run-instances --key-name KEYNAME --instance-type  
t3.medium --image-id AMIID --user-data  
file:///c:\temp\UserData.txt --security-group-ids HTTPACCESS -  
-subnet-id SUBNETID --associate-public-ip-address --tag-  
specifications 'ResourceType=instance,Tags=  
[ {Key=Name,Value=WebServerBaseImage}]' --output text --query  
'Instances[*].InstanceId'
```

The output of this command will provide you with an **InstanceId**. This value is referred to as **new-instance-id** in subsequent steps and should be replaced appropriately.

35. Use the **aws ec2 wait instance-running** command to monitor this instance's status. Replace *NEW_INSTANCE_ID* with the value you copied

35. Use the **aws ec2 wait instance-running** command to monitor this instance's status. Replace *NEW-INSTANCE-ID* with the value you copied previously.

```
aws ec2 wait instance-status-ok --instance-ids NEW-INSTANCE-ID
```

Wait for the command to return to a prompt, before proceeding to the next step.

36. Your instance should have started a new web server. To test that the web server was installed properly, obtain the public DNS name. Copy the output of this value (minus quotation marks). This value is referred to as **public-dns-address** in the next procedure. Replace *NEW-INSTANCE-ID* with the value you copied previously.

```
aws ec2 describe-instances --query  
'Reservations[0].Instances[0].NetworkInterfaces[0].Association.PublicDnsName'  
--output text --instance-id NEW-INSTANCE-ID
```

37. Use a Web browser to navigate to the following page. Replace *PUBLIC-DNS-ADDRESS* with the value you copied in the last step.



- Make sure to add `/Default` at the end of the below command.
- It could take eight to ten minutes for the user data to execute.
- Do not click on *Start Stress* at this stage.

```
http://PUBLIC-DNS-ADDRESS/Default
```

If your web server does not appear to be running, check with your instructor.

instructor.

Create a Custom AMI

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

38. Use the **aws ec2 create-image** command to create a new AMI based on this instance. Replace *NEW-INSTANCE-ID* with the value you copied previously.

```
aws ec2 create-image --name WebServer --instance-id NEW-  
INSTANCE-ID
```

💡 By default, **create-image** will restart the current instance before creating the AMI, in order to ensure the integrity of the image on the file system.

Task 2: Create an Auto Scaling Environment

In this section, you will create a load balancer that pools a group of EC2 instances under a single DNS address. You will use Auto Scaling to create a dynamically scalable pool of EC2 instances based on the image that you created in the previous section. Finally, you will create a set of alarms that will 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

your load balancer group whenever the CPU performance of any machine within the group exceeds or falls below a set of specified thresholds.

The following task can be performed using either the AWS CLI or the Management Console. For purposes of simplicity, you will implement this task using the Management Console.

Create an Application Load Balancer

39. In the left navigation pane, click **Load Balancers** (you might need to scroll down to find it).

40. Click **Create Load Balancer**

41. Under **Application Load Balancer**, click **Create**

42. For **Name**, enter: `webserverloadbalancer`

43. Scroll down to the **Availability Zones** section, then:

- For **VPC**, select: **Lab VPC**

You will now specify which *subnets* the Load Balancer should use. It will be an Internet-facing load balancer, so you will select both Public Subnets.

44. Click the **first** displayed Availability Zone, then click the **Public Subnet 1** displayed underneath.

45. Click the **second** displayed Availability Zone, then click the **Public Subnet 2** displayed underneath.

You should now have two subnets selected: **Public Subnet 1** and **Public Subnet 2**. (If not, go back and try the configuration again.)

Public Subnet 2. (If not, go back and try the configuration again.)

46. Click **Next: Configure Security Settings**

A warning is displayed, which recommends using HTTPS for improved security. This is good advice but is not necessary for this lab.

47. Click **Next: Configure Security Groups**

48. Select ☒ **HTTPAccess** and ensure it is the only item selected.

49. Click **Next: Configure Routing**

Target Groups define where to *send* traffic that comes into the Load Balancer. The Application Load Balancer can send traffic to multiple Target Groups based upon the URL of the incoming request, such as having requests from mobile apps going to a different set of servers. Your web application will use only one Target Group.

50. For **Name**, enter: `webserver-app`

51. Expand ► **Advanced health check settings**.

The Application Load Balancer automatically performs *Health Checks* on all instances to ensure that they are responding to requests. The default settings are recommended, but you will make them slightly faster for use in this lab.

52. Configure these values:

- **Path** `/Default`
- **Healthy threshold:** `2`
- **Unhealthy threshold:** `10`
- **Timeout:** `10`
- **Interval:** `30`

This means that the Health Check will be performed every 30 seconds

This means that the Health Check will be performed every 30 seconds and if the instance responds correctly twice in a row, it will be considered healthy.

53. Click **Next: Register Targets**

Targets are the individual instances that will respond to requests from the Load Balancer. You do not have any web application instances yet, so you can skip this step.

54. Click **Next: Review**

55. Review the settings and click **Create** then **Close**

Create a Launch Configuration

The launch configuration will be used by your Auto Scaling group to specify which AMI to use to create new EC2 instances. For this example, you will launch the AMI that you created previously, which automatically configures itself as a web server when it is launched.

56. In the left navigation pane, click **Launch Configurations**.

57. Click **Create launch configuration**

58. On the **Choose AMI** step, click the **My AMIs** tab.

59. In the row for **WebServer**, click **Select**

60. On the **Choose Instance Type** step, make sure the **t3.micro** instance is selected.

61. Click **Next: Configure details**

62. On the **Configure details** step, configure:

62. On the **Configure details** step, configure:

- **Name:** `WebServerLaunchConfiguration`
- **Monitoring:** Select ☒ **Enable CloudWatch detailed monitoring**

63. Click **Next: Add Storage**

You will use the default storage settings.

64. Click **Next: Configure Security Group**

65. On the **Configure Security Group** step, click ☒ **Select an existing security group.**

66. Select ☒ **HTTPAccess.**

67. Click **Review**

You may receive a warning that your security group will not enable you to SSH into the instance. You may safely ignore this warning because you will not need to SSH into the instances in your Auto Scaling group.

Click **Continue**

68. Click **Create launch configuration**

69. Select ☒ the acknowledgement check box, then click

Create launch configuration

70. Click **Create an Auto Scaling group using this launch configuration**

Create an Auto Scaling Group

Your Auto Scaling group will create a minimum number of Amazon EC2 instances that will reside behind your load balancer. In subsequent procedures, you will also add scale-out and scale-in policies that

instances that will reside behind your load balancer. In subsequent procedures, you will also add scale-out and scale-in policies that increase or decrease the number of running instances in reaction to alarms triggered by Amazon CloudWatch.

You should already be on the **Create Auto Scaling Group** page.

71. Configure these settings:

- **Group name:** `WebServersASGroup`
- **Group size:** Start with **2** instances.
- **Network:** **Lab VPC**
- **Subnet:** **Private Subnet 1** and **Private Subnet 2**

🗨 You may receive a warning message that no public IP addresses will be assigned to your instances because you are hosting them outside of your default VPC. This is fine because the public load balancer will be responsible for directing traffic to your instances.

72. Expand ► **Advanced Details**, specify the following settings:

- **Load Balancing:** Select ☒ **Receive traffic from one or more load balancers**
- **Target Groups:** Click `webserver-app`
- **Monitoring:** Select ☒ **Enable CloudWatch detailed monitoring**

73. Click **Next: Configure scaling policies**.

74. Select ☒ **Use scaling policies to adjust the capacity of this group**.

75. Specify scaling between **2** and **4** instances.

You will configure Auto Scaling to target 40% CPU Utilization. If average CPU Utilization exceeds this target, additional instances will be launched. If average CPU Utilization falls below this target, instances *might* be terminated.

launched. If average CPU utilization falls below this target, instances *might* be terminated.

76. For **Target value**, enter: 40

77. Click on **Next: Configure Notifications**

78. Click on **Next: Configure Tags**

79. Enter **Name** under **Key** and **WebApp** under **Value**.

80. Click **Review**

81. Review your configuration, and then click **Create Auto Scaling group**

82. On the **Auto Scaling group creation status** page, click **Close**

Verifying the Auto Scaling configuration

In this task, you will 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, thus triggering the scale out alarm.

83. In the left navigation pane, click **Instances**.

84. Verify that two new instances labelled **WebApp** are being created as part of your Auto Scaling group.

85. Wait for the two new instances to complete initialization before you proceed to the next step. Observe the **Status Checks** field for the instances until it shows that both status checks have completed successfully.

86. In the left navigation pane, click **Target Groups**, and then select ☒ your target group (**webserver-app**).

86. In the left navigation pane, click **Target Groups**, and then select ☒ your target group (**webserver-app**).
87. On the **Targets** tab in the lower half of your screen, verify that two instances are being created. Keep refreshing this list until the **Status** of these instances changes to **healthy**.
88. In the left navigation pane, click **Load Balancers** and then select ☒ **webserverloadbalancer**.
89. On the **Description** tab below, copy the **DNS name** value (which should look like **webserverloadbalancer-xxxxxxxxxx.xx-xxxx-x.elb.amazonaws.com**). This value will be referred to as **load-balancer-url** in the next step.
90. Use a Web browser to navigate to the below page. Replace *LOAD-BALANCER-URL* with the DNS name value you copied in the last step.

```
http://LOAD-BALANCER-URL/Default
```

91. On the web page, click **Start Stress**.

This will call the application consume.exe in the background, causing the CPU utilization on the instance that serviced this request to spike to 100%.

92. In the left navigation pane of the management console, click **Auto Scaling Groups**.
93. Select ☒ **WebServerASGroup**.
94. Select the **Activity History** tab for your Auto Scaling group. After **eight to ten minutes**, you should see your Auto Scaling group add a new instance.

💡 If you do not see any scaling activity, connect to the ELB application

● If you do not see any scaling activity, connect to the ELB application in a new browser and click on Start Stress to increase CPU utilization on the backend instances.



A new instance was created because CloudWatch detected that the average CPU utilization of your Auto Scaling group exceeded 40%, and your scale-up policy has been triggered in response.

Lab Complete

Congratulations! You have completed the lab.

End Lab

Follow these steps to close the console, end your lab, and evaluate the experience.

95. Return to the AWS Management Console.
96. On the navigation bar, click **awsstudent@<AccountNumber>**, and then click **Sign Out**.
97. Click  **End Lab**
98. Click  **OK**



98. Click 

99. (Optional):

- Select the applicable number of stars ☆
- Type a comment
- Click **Submit**
 - 1 star = Very dissatisfied
 - 2 stars = Dissatisfied
 - 3 stars = Neutral
 - 4 stars = Satisfied
 - 5 stars = Very satisfied

You may close the dialog if you don't want to provide feedback.

