

Systems Operations on AWS - Lab 2W - Creating Amazon EC2 Instances (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.

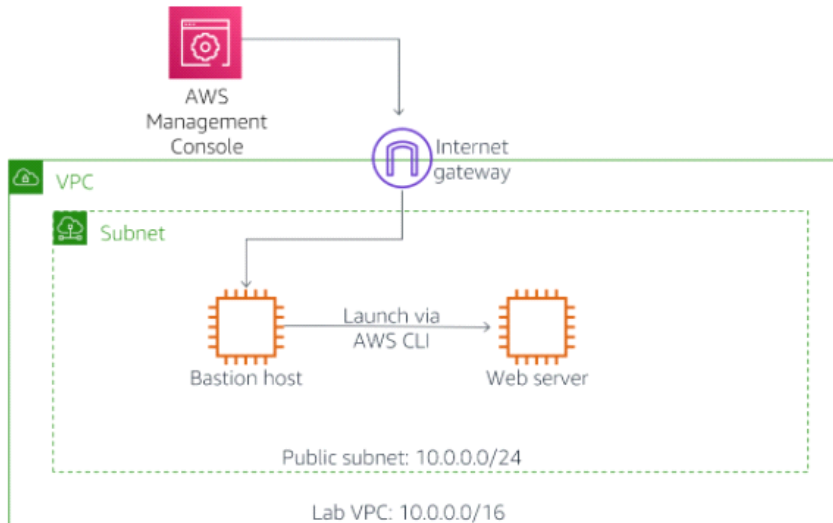
Other questions? Contact us at <https://aws.amazon.com/contact-us/aws-training/>

Traditional methods of deploying servers and configuring security are complex and often involve multiple teams and long delays. Fortunately, it is quick and easy to deploy secure infrastructure in the cloud. As a Systems Operator, you can automate many of these processes using the AWS Command-Line Interface.

In this lab you will:

- Launch an Amazon EC2 instance using the management console
- Launch an Amazon EC2 instance using the AWS Command-Line Interface (AWS CLI)

The final architecture will be:



If you have time, an **optional Challenge section** will then have you troubleshoot some issues with Amazon EC2 instances.

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 Remote Desktop. If you instead wish to use SSH to connect to an Amazon EC2 Linux instance, **please use the Linux version of this lab.**

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

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.

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: Launch an Amazon EC2 Instance using the Management Console

In this task, you will launch an Amazon EC2 instance using the management console. The instance will be a Bastion Server, from which you can use the AWS Command-

In this task, you will launch an Amazon EC2 instance using the management console. The instance will be a Bastion Server, from which you can use the AWS Command-Line Interface (AWS CLI).

4. In the **AWS Management Console**, on the **Services** menu, click **EC2**.
5. Click **Launch Instance** > **Launch instance**.

Step 1: Choose an Amazon Machine Image (AMI)

This step allows you to choose an AMI, which contains a copy of the disk volume that will be used to launch the instance.

● Examine the list of AMIs that are displayed, showing many versions of Microsoft Windows and Linux. These disk images are regularly updated to incorporate security patches and software that helps you use AWS services. You can also create your own AMI that includes your own data and applications, or you can select pre-built commercial applications from the **AWS Marketplace**.

Your Bastion Server will use *Microsoft Windows Server 2019 Base*.

6. Beside the **Microsoft Windows Server 2019 Base**, click **Select**

Step 2: Choose an Instance Type

This step allows you to choose an **Instance Type**, which determines the resources that will be allocated to your EC2 instance. Each Instance Type allocates a combination of virtual CPUs, memory, disk storage and network performance.

Instance Types are divided into **families** such as Compute-optimized, Memory-optimized and Storage-Optimized. The name of the Instance Type includes a family identifier, such as **t2** and **m4**. The number indicates the *generation* of the instance, so **m5** is newer than **m4**.

Your application will use a **t2.small** Instance Type, which is a small instance that can burst above baseline performance when it is busy. It is ideal for development, testing and for applications that have bursty workloads.

7. Select ☒ **t2.small**
8. Click **Next: Configure Instance Details**

Step 3: Configure Instance Details

This step allows you to configure instance details, such as the number of instances

This step allows you to configure instance details, such as the number of instances to launch and the network configuration. You can hover over the **i** icons to view a description of each field.

You will launch the instance in a public subnet within the *Lab VPC* network.

9. Configure these settings:

- **Network:** *Lab VPC*
- **Subnet:** *Public Subnet*
- **IAM role:** *Bastion-Role*

The *Bastion-Role* grants permission to applications running on the instance to make requests to the Amazon EC2 service. This is required for the second half of this lab, where you will use the AWS CLI to communicate with the EC2 service.

- Expand ► **Advanced Details**
- **User Data:** Paste this script:

```
<powershell>
Set-ExecutionPolicy Unrestricted -Force
New-Item -ItemType directory -Path 'C:\temp', 'C:\temp\aws'
$webclient = New-Object System.Net.WebClient
$webclient.DownloadFile('https://s3.amazonaws.com/aws-cli/AWSCLI64.msi',
'C:\temp\aws\AWSCLI64.msi')
Start-Process 'C:\temp\aws\AWSCLI64.msi' -ArgumentList /qn -Wait
</powershell>
```

This script will install the AWS Command-Line Interface.

10. Click **Next: Add Storage**

Step 4: Add Storage

This step can be used to add additional Amazon Elastic Block Store (EBS) disk volumes and configure their size and performance.

You can hover over the **i** icons to view a description of each field.

You will use the default disk size, so no changes are required.

11. Click **Next: Add Tags**

Step 5: Add Tags

Tags allow you to categorize your AWS resources in different ways, such as by purpose, owner, or environment. This is useful when you have many resources of the

Tags allow you to categorize your AWS resources in different ways, such as by purpose, owner, or environment. This is useful when you have many resources of the same type — you can quickly identify a specific resource by their tags. Each tag consists of a *Key* and a *Value*, both of which you define.

12. Click **Add Tag** then configure:

- **Key:** Name
- **Value:** Bastion Server

This name will appear on the instance in the EC2 management console.

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

Step 6: Configure Security Group

You will create a new Security Group that permits RDP connections. This security group will allow you to log in to the Bastion Server via RDP.

14. Configure these settings:

- **Security group name:** Bastion security group
- **Description:** Permit RDP connections

Permissions for inbound access via RDP (port 3389) have already been configured by default.

15. Click **Review and Launch**

Step 7: Review Instance Launch

This step displays a summary of the configuration for the instance you are about to launch.

16. Click **Launch**

A **Select an existing key pair or create a new key pair** window will appear.

17. Select ☒ **I acknowledge that....**

18. Click **Launch Instances**

Your instance will now be launched.

19. Click **View Instances**

The Bastion Server will appear in a *pending* state, which means it is being launched.

19. Click **View Instances**.

The Bastion Server will appear in a *pending* state, which means it is being launched. It will then change to *running*, which indicates that the instance has started booting.

20. Wait for the **Instance State** to change to  **running**.


21. Select ☒ **Bastion Server**.

Review the information displayed in the **Description** tab in the lower half of the page. It includes information about the instance type, security settings and network settings.

Task 2: Log into the Bastion Server

In this task, you will log into the Bastion Server that you just created.


Mac Users: You will need to download [Microsoft Remote Desktop](#) from the Mac App Store, or use another RDP utility such as [CoRD](#).

22. To the left of the instructions you are currently reading, click  **Download PEM**.

23. Click **Actions**  and select **Connect**.

When a new Windows instance is launched, a random Administrator password is generated. This ensures that nobody else will be able to login to the instance. The password is encrypted using the Key Pair selected when the instance was launched, so you must use the Key Pair to decrypt the password.

24. Use **Get Password**


 If you receive the message *Password not available yet*, please wait a minute and try again. It might take a few minutes for the instance to be ready.


25. For **Key Pair Path**, select the PEM file you downloaded earlier.

26. Click **Decrypt Password**

The login information for the Bastion Server will now be displayed.

27. Click **Download Remote Desktop File** and use it to login to the Bastion Server using the information displayed on the screen.

Tip: You can hover your mouse over the password field and click the Copy  icon that appears.

Tip: You can hover your mouse over the password field and click the Copy  icon that appears.

💬 Please ask your instructor for connection assistance if necessary.

Now that you are connected to the Bastion Server, you can use the AWS CLI to call AWS services.


Task 3: Launch an Instance using the AWS CLI

In this task, you will launch an Amazon EC2 instance using the AWS Command-Line Interface (CLI). The AWS CLI makes it easy to automate the provision and configuration of AWS resources.

The new instance will be configured as a Web Server.

Configure the AWS CLI

First, you will configure the AWS Command-Line Interface with the Region being used by your lab.

28. In your Remote Desktop session, click the **Windows**  menu and then click **Windows PowerShell**.

29. In the PowerShell window, run this command:

```
aws configure
```

30. Enter these details:

- **AWS Access Key ID:** Press Enter
- **AWS Secret Access Key:** Press Enter
- **Default region name:** Enter the **Region** value shown to the left of the instructions you are currently reading
- **Default output format [None]:** Press Enter

The AWS CLI will now send all future commands to that AWS region.

Obtain the AMI to Use

Obtain the AMI to Use

One of the parameters required when launching an instance is the Amazon Machine Image (AMI), which will populate the boot disk of the instance. AMIs are continually patched and updated by AWS, so it is recommended to always use the latest AMI when launching instances.

You will use the **AWS Systems Manager Parameter Store** to obtain the ID of the most recent *Microsoft Windows Server 2019 Base* AMI. AWS maintains a list of standard AMIs in the Parameter Store, making this task easy to automate.

31. Paste this script into your Remote Desktop session:

```
$AMI = (aws ssm get-parameters --names /aws/service/ami-windows-  
latest/Windows_Server-2019-English-Full-Base --query 'Parameters[0].  
[Value]' --output text)  
echo $AMI
```

This command did the following:

- Called the AWS Systems Manager (*ssm*) and used the **get-parameters** command to retrieve a value from the Parameter Store
- The AMI requested was for Microsoft Windows Server 2019 Base
- The AMI ID has been stored in an Environment Variable called *AMI*

Obtain the Subnet to Use

You will be launching the new instance in the Public Subnet. When launching an instance, the **SubnetId** can be specified.

The following command will retrieve the *SubnetId* for the Public Subnet:

32. Paste this command:

```
$SUBNET = (aws ec2 describe-subnets --filters  
"Name=tag:Name,Values=Public Subnet" --query Subnets[].SubnetId --output  
text)  
echo $SUBNET
```

This uses the AWS CLI to retrieve the Subnet ID of the subnet named *Public Subnet*. The Subnet ID should match the *PublicSubnet* shown to the left of these instructions.

Obtain the Security Group to Use

A *Web Security Group* has been provided as part of this lab, which allows inbound HTTP requests

A *Web Security Group* has been provided as part of this lab, which allows inbound HTTP requests.

33. Paste this command:

```
$SG = (aws ec2 describe-security-groups --filters Name=group-name,Values=WebSecurityGroup --query SecurityGroups[].GroupId --output text)
echo $SG
```

The command retrieves the *Security Group ID* of the Web Security Group.

The security group ID should match the *WebSecurityGroup* shown to the left of these instructions.

Obtain the Key Pair to Use

A Key Pair is used to encrypt the Administrator password. You will need to specify a Key Pair to be able to obtain the password to the new instance

34. Paste this command:

```
$KEYPAIR = (aws ec2 describe-key-pairs --query KeyPairs[].KeyName --output text)
echo $KEYPAIR
```

The command retrieves the name of the Key Pair that you downloaded earlier.

Download a User Data script

You will be launching an instance that will act as a Web Server. To install and configure the web server, you will provide a **User Data script** that will be automatically executed when the instance launches.

35. Paste this command to download the User Data script:

```
(new-object net.webclient).DownloadFile('https://us-west-2-tcprod.s3.amazonaws.com/courses/ILT-TF-100-SYSOPS/v3.3.15/lab-2-ec2-windows/scripts/UserData.txt','C:\temp\UserData.txt')
```

36. Paste this command to view the contents of the script:

```
cls; type C:\temp\UserData.txt
```

The script does the following:

The script does the following:

- Installs the IIS web server
- Downloads a zip file containing the web application
- Installs the web application

Launch the Instance

You now have all the necessary information require to launch the Web Server instance!

37. Paste this command:

```
$INSTANCE = (
aws ec2 run-instances --image-id $AMI --subnet-id $SUBNET --security-
group-ids $SG --key-name $KEYPAIR --user-data file://c:\temp\UserData.txt
--instance-type t2.small --tag-specifications
"ResourceType=instance,Tags=[{Key=Name,Value=Web Server}]" --query
"Instances[*].InstanceId" --output text
)
echo $INSTANCE
```

The command launches a new instance (*run_instances*) using these parameters:

- **Image:** Uses the AMI value obtained earlier from the Parameter Store
- **Subnet:** Specifies the *Public Subnet* obtained earlier and, by association, the VPC in which to launch the instance
- **Security Group:** Uses the *Web Security Group* obtained earlier, which permits HTTP access
- **Key Pair:** Specifies the Key Pair you downloaded earlier
- **User Data:** References the User Data script you downloaded, which installs the web application
- **Instance Type:** Specifies the type of instance to launch
- **Tags:** Assigns a *Name* tag with the value of *Web Server*

The **query** parameter specifies that the command should return the *Instance ID* once the instance is launched.

The **output** parameter specifies that the output of the command should be in *text*. Other output options are *json* and *table*.

The ID of the new instance has been stored in the *INSTANCE* environment variable.

Wait for the Instance to be Ready

You can monitor the status of the instance via the Management Console, but you can

You can monitor the status of the instance via the Management Console, but you can also query the status via the AWS CLI.

38. Paste this command:

```
aws ec2 describe-instances --instance-ids $INSTANCE
```

All information related to the instance will be displayed in JSON format. Amongst this information is the instance status.

Specific information can be obtained by using the **query** parameter.

39. Paste this command:

```
aws ec2 describe-instances --instance-ids $INSTANCE --query  
'Reservations[].Instances[].State.Name' --output text
```

This is the same command but, rather than displaying all information about the instance, only displays the name of the instance *State*.

This will display a status of **pending** or **running**.

Repeat the above command until it returns a status of **running**.

Test the Web Server

You can now test that the web server is working. You can retrieve a URL to the instance via the AWS CLI.

40. Paste this command:

```
aws ec2 describe-instances --instance-ids $INSTANCE --query  
Reservations[].Instances[].PublicDnsName --output text
```

This returns the **DNS Name** of the instance.

41. Copy the DNS name that is displayed.

It should look similar to: *ec2-35-11-22-33.us-west-2.compute.amazonaws.com*

42. On your own computer, open a new web browser tab, paste the DNS name, then press Enter.

A web page should be displayed, demonstrating that the web server was successfully launched and configured.

A web page should be displayed, demonstrating that the web server was successfully launched and configured.

● If a page does not appear, **do not panic!** The new Windows instance will take several minutes to boot and install IIS. Simply keep trying the page every couple of minutes until the page displays. It will eventually appear!

You can also see the instance in the EC2 management console.

43. Return to the web browser tab containing the EC2 management console.

44. Click  Refresh.

The list should now include the *Web Server* instance that was launched via CLI command.

As seen in this task, the AWS CLI makes it possible to programmatically access and control AWS services. These commands can be placed in a script and run as a standard process to deploy consistent, reliable infrastructure with minimal scope for human error.

Which method should you use?

- **Launch from the management console** when you quickly need to launch a one-off or temporary instance.
- **Launch via a script** when you need to automate the creation of an instance in a repeatable, reliable manner.
- **Launch via CloudFormation** when you wish to launch related resources together.

AWS Tools for PowerShell

In addition to the AWS CLI, you can use Powershell commands to access AWS.

Here are some example commands available to do the equivalent to the commands you ran earlier:

- Get-SSMParameter
- Get-EC2Subnet
- Get-EC2SecurityGroup
- Get-EC2KeyPair
- New-EC2Instance

For more information, see: [AWS Tools for Windows PowerShell](#)

Challenge 1: Connect to an Amazon EC2 Instance

💡 This challenge is **optional** and is provided in case you still have lab time remaining.

In this challenge, your mission is to troubleshoot an instance called *Misconfigured Web Server*.

45. Your tasks are:

- Obtain the DNS name of the **Misconfigured Web Server**
- Try to establish an **RDP connection** to the instance
- Diagnose why this does not work and **fix the misconfiguration**

At the end of the lab, your instructor will ask you:

- What was the problem?
- What did you do to fix the problem?

Challenge 2: Fix the Web Server Installation

In this challenge, your mission is to troubleshoot the web server installation on the *Misconfigured Web Server*.

46. Your tasks are:

- Point your web browser to the *Misconfigured Web Server* (the IP address is shown to the left of these instructions)
- Why does the web site not appear?
- Diagnose the problem and try to fix it on the instance, or use **Launch More Like This** to launch another instance with a fixed configuration

At the end of the lab, your instructor will ask you:

- What was the problem?
- What did you do to fix the problem?

Lab Complete

Congratulations! You have completed the lab.

End Lab

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

47. Return to the AWS Management Console.

48. On the navigation bar, click **awsstudent@<AccountNumber>**, and then click **Sign Out**.

49. Click  **End Lab**

50. Click 

51. (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.