

Developing on AWS – Lab 2 – Developing Storage Solutions with Amazon S3

2 hours 30 minutes

Free

★★★★★ [Rate Lab](#)



.Net version

© 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. All trademarks are the property of their owners.

Corrections, feedback, or other questions? Contact us at [AWS Training and Certification](#).



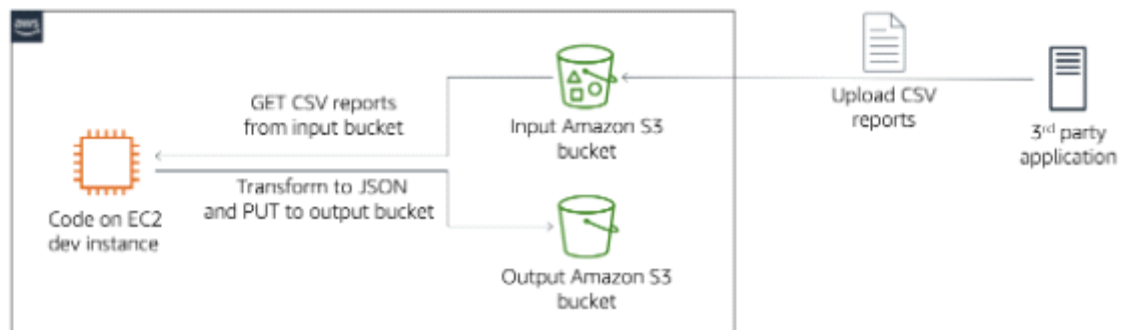
Overview

In this lab, you will learn how to use the AWS SDK to get and put objects in an S3 bucket, restrict access by using pre-signed URLs, secure data with server-side encryption, and update an object's user metadata.

The terms *file* and *object* are used interchangeably when referring to the contents of S3 buckets.

The story is a third-party application drops CSV-formatted reports into an S3 bucket. Another internal application needs to process this data but can only process JSON-formatted content.

You will develop an application that converts every file in the input bucket from CSV to JSON format and drops the content into an output bucket. You will encrypt the converted files, restrict access to them by using pre-signed URLs, and add a contact person in the user-defined metadata.



Objectives

After completing this lab, you will be able to:

- Create an S3 bucket and manage its properties.

- Create an S3 bucket and manage its properties.
- Retrieve objects from an S3 bucket.
- Upload objects to an S3 bucket.
- Use pre-signed URLs to restrict access.
- Request server-side encryption when uploading objects.
- Add user metadata to objects.

Prerequisites

This lab requires:

- Access to a notebook computer with Wi-Fi running Microsoft Windows or macOS.
- An Internet browser such as Chrome, Firefox, or IE9+. (previous versions of Internet Explorer are not supported)
- You will need either an SSH client, such as PuTTY, or a Microsoft Remote Desktop client to connect to your development EC2 instance.


Note

You can use an iPad or tablet device to access these directions in the lab console.

Duration

This lab will require around **60 minutes** to complete.

Start Lab

1. At the top of your screen, launch your lab by choosing 

1. At the top of your screen, launch your lab by choosing **Start Lab**

This starts the process of provisioning your lab resources. An estimated amount of time to provision your lab resources is 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 choosing **Open Console**

This opens an AWS Management Console sign-in page.

3. On the sign-in page, configure:

- **IAM user name:** `awsstudent`
- **Password:** Paste the value of **Password** from the left side of the lab page
- Choose **Sign In**

⚠ 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:**

- Choose **click here**
- Close your browser tab to return to your initial lab window

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

Task 1: Connecting to Your Development Environment

4. Your first step to develop this code is to connect to the **Windows Dev Instance**. If you remember how to do so from the previous lab, continue with the next task. If you don't remember, you can find the details in here: [Appendix: Connect to your Development Environment](#).

Task 2: Developing Your Application

You will develop an application that retrieves objects from an S3 bucket, transforms content from CSV to JSON format, and uploads the transformed content to another S3 bucket.

Your application will also implement the following features:

- Pre-signed URLs for objects in the output bucket
- Server-side encryption
- User metadata

Lab Skeleton Code

Lab Skeleton Code

Your lab skeleton code has been set up in the Amazon EC2 instance under the `<yourWorkDir>` directory.

The base working directory, which is referred to as `<yourWorkDir>`, is at the following location:

- Windows Dev Instance: `c:\temp\workdir`

See the [Appendix: Working with Lab Skeleton Code in .NET](#) for instructions on working with lab skeleton code in **Visual Studio IDE**.

Note

The Lab skeleton code includes comments beginning with `// TODO` to help you quickly locate the sections of code you must modify.

Solution Code

You can refer to the solution code in `Solution.cs`.

Task 2.1: Develop Code to Transform Files and Upload Output

In this section, you will develop code to retrieve files from input S3 bucket and post the transformed files to the output S3 bucket.

5. In Windows Explorer, navigate to `c:\temp\workdir\s3CSharpLab`. Open the `s3Solution` solution file in Visual Studio.
6. **Build** the Solution in Visual Studio by pressing **Ctrl+Shift+B**. This will restore dependencies using the NuGet Package Manager and allow IntelliSense to operate correctly. You may see errors when you open Visual Studio, but once you have built the Solution, you shouldn't see any errors.

operate correctly. You may see errors when you open Visual Studio, but once you have built the Solution, you shouldn't see any errors.

7. Open the `DataTransformer.cs` file. You will need to update the code in this file to implement the features described in the following instructions, unless noted otherwise.
8. The code creates the input and output buckets to be used in the lab. The input bucket will be used to store CSV formatted files. The output bucket will be used to store the transformed JSON formatted files.

Specify unique values for the `InputBucketName` and `OutputBucketName` variables which will be used to name the buckets.

Make sure to pick names that following the [Rules for Bucket Naming in the S3 Developer Guide](#).

TODO in the code

```
// TODO 1: Set input bucket name (must be globally unique)
// TODO 2: Set output bucket name (must be globally unique)
```

9. In the `CreateS3Client` method, to handle operations on buckets, create an instance of the `AmazonS3Client` object.

See the sample code snippet below. You may also find an [example in the documentation](#). You do not have to specify a region endpoint if you specify the region.

```
s3ForStudentBuckets = new AmazonS3Client();
```

TODO in the code

```
// TODO 3: Replace the solution with your own code
```

10. In the `GetS3Object` method, retrieve a list of objects from the input S3 bucket by using the bucket name and object key.

TODO in the code

```
// TODO 4: Replace the solution with your own code
```

For information about retrieving objects from S3 buckets, see [Getting Objects](#).

11. In the `PutObjectBasic` method, store the result of `TransformText` in a variable and upload the object by using the `ContentBody` parameter.

TODO in the code

```
// TODO 5: Replace the solution with your own code
```

For information about uploading objects to S3 buckets, see [Uploading Objects](#).

12. Save the `DataTransformer.cs` file.
13. Build the solution in Visual Studio by pressing **Ctrl+Shift+B** and ensure no errors are displayed.
14. Run the solution by **selecting Start**.
15. Look at the output to make sure that your code builds and runs successfully. You should see a list of files that were transformed. In case of errors, correct the code and run the solution again.
16. Within the **Test Explorer** (*Test > Windows > Test Explorer*), run the `TestAllFilesExistAndModifiedInOutput` test method to verify the files were transformed. If you encounter any errors, address them in your code.

`TestAllFilesExistAndModifiedInOutput` test method to verify the files were transformed. If you encounter any errors, address them in your code, rebuild and run the solution, and re-run the test method.

Task 2.2: Develop Code to Generate Pre-signed URLs

In this section, you will generate pre-signed URLs for the transformed files. The pre-signed URLs enable users to retrieve objects for a limited time from the output S3 bucket.

17. Continue to edit the `DataTransformer.cs` file in Visual Studio. You will need to update the code in this file to implement the features described in the following instructions, unless noted otherwise.
18. In the `GeneratePresignedUrl` method, generate a pre-signed URL for each S3 object.

The Pre-signed URL must have the following characteristics:

- can be used only to retrieve objects.
- must expire after 15 minutes.

TODO in the code

```
// TODO 6: Replace the solution with your own code
```

For information about generating a pre-signed URL, see [Generating a Pre-signed URL](#). Note that if you do not specify an [HttpVerb](#), the default will be GET.

19. Save the `DataTransformer.cs` file.
20. Build the solution in Visual Studio and ensure no errors are displayed.

20. Build the solution in Visual Studio and ensure no errors are displayed.

21. Run the solution.

22. Check the output to make sure that your code builds and runs successfully. In case of errors, correct the code and run this command again.

You should see a pre-signed URL for each transformed file.

23. Copy one of the pre-signed URLs printed from the debug trace window into a text editor and remove any hard-line breaks.

24. Copy and paste the clean pre-signed URL in a web browser. You must be able to view the transformed file containing text in JSON format.

Task 2.3: (Optional Challenge) Develop Code to Request Server-Side Encryption and Add User Metadata

In this section, you will specify server-side encryption and add user metadata before uploading the object to the output S3 bucket.

25. Continue to edit the `DataTransformer.cs` file in Visual Studio. You will need to update the code in this file to implement the features described in the following instructions, unless noted otherwise.

26. In the `main` method, comment out the line that invokes the `PutObjectBasic` method and uncomment the line that invokes the `PutObjectEnhanced` method.

TODO in the code

```
// TODO 7: Switch to enhanced file upload
```

27. In the `PutObjectEnhanced` method, enable server-side encryption and

-
27. In the `PutObjectEnhanced` method, enable server-side encryption and upload the object to the output S3 bucket.

For more information, see [Specifying Server-Side Encryption](#).

Add the following name-value pair to the object's user metadata.

- **Name:** `contact`
- **Value:** `John Doe`

For more information, see Uploading an [Object Using .NET SDK](#) reference documentation.

TODO in the code

```
// TODO 8: Replace the solution with your own code
```

28. Save the `DataTransformer.cs` file.
29. Build the solution in Visual Studio and ensure no errors are displayed.
30. Run the solution.
31. Check the output to make sure that your code builds and runs successfully. You should see the object's metadata listed with your user metadata and server-side encryption enabled.

In case of errors, correct the code and run this command again.


End Lab

End Lab

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

32. Return to the AWS Management Console.

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

34. Choose  **End Lab**

35. Choose  **OK**

36. (Optional):

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

You may close the window if you don't want to provide feedback.
Congratulations! You are done!

Additional Resources

Additional Resources

For more information about AWS Training and Certification, see

<http://aws.amazon.com/training/>.

Your feedback is welcome and appreciated.

If you would like to share any feedback, suggestions, or corrections, please provide the details in our [AWS Training and Certification Contact Form](#).

Appendix: Connecting to Your Development Environment

You can connect to your Dev instance by using one of the following methods:

- Use Apache Guacamole to connect to your Windows Dev instance
- Use Remote Desktop to connect to your Windows Dev instance

To connect to the **Windows EC2 instance** by using Guacamole (Recommended), see the following directions:

- [Connect to Your Windows Dev Instance by Using Apache Guacamole](#)

To connect to the **Windows EC2 instance** by using RDP, see the following directions:

- [Connect to Your Windows Dev Instance from a Windows Machine](#)
- [Connect to Your Windows Dev Instance from a macOS Machine](#)

-
- [Connect to Your Windows Dev Instance from a macOS Machine](#)

Connect to Your Windows Dev Instance by Using Apache Guacamole

37. In the **Connection Details** section in the lab console, go to the bottom for the Guacamole information. Copy the **GuacamoleLink** and paste it into a browser.
38. Go back to the lab console and copy the **WindowsPassword** to the clipboard.
39. Go to the Apache Guacamole sign in in the browser. Sign in by using the following steps:
 - For **Username**, enter: `student`
 - For **Password**, paste the **WindowsPassword** from the clipboard.
 - Select **Log In**.

Your connection to your remote instance should start momentarily. Once you open a connection, you will see an image of the Dev instance desktop. You can interact with this image just as you would your normal desktop, or any remote desktop client.

You are now connected to your Windows Dev instance in the browser via Guacamole.

Tip Web browsers don't provide access to clipboard data, which means synchronization between your local clipboard and the remote clipboard is impossible. To copy and paste when using Guacamole, you must use the Clipboard editor. To open the Clipboard editor, press **Ctrl -> Alt -> Shift**.

impossible. To copy and paste when using Guacamole, you must use the Clipboard editor. To open the Clipboard editor, press **Ctrl -> Alt -> Shift**.

Copy your text and paste it to the Clipboard editor. This will set the clipboard of your Dev instance to what you just pasted. You can also edit the text that you place in the Clipboard editor before pasting into your remote desktop. To close the Clipboard editor, press **Ctrl -> Alt -> Shift**.

To continue this lab, move on to [Task 2: Developing Your Application](#).

Connect to Your Windows Dev Instance from a Windows Machine

In this task, you will connect to a Windows EC2 instance from your Windows machine.

Note

Perform the steps in this task only if you are connecting to **Windows Dev Instance** from a Windows machine.

40. In the lab console, go to the **Connection Details** section and copy the **WindowsInstanceIP** to the clipboard.
41. Open the Remote Desktop Connection application on your computer.
 - On Windows 7, select the **Start** icon, and in the **Search programs and files** textbox, enter: `Remote Desktop Connection`. Select the application when it appears in the **Programs** list.
 - On Windows 8, activate the Charms menu by moving the cursor into the lower right corner of the screen, and select the **Search** icon. Enter: `Remote Desktop Connection`. Select the application when it appears in

lower right corner of the screen, and select the **Search** icon. Enter:

`Remote Desktop Connection`. Select the application when it appears in the **Programs** list.

- On Windows 10, select the **Start** icon, and then, select the **Search** icon. Enter: `Remote Desktop Connection`. Select the application when it appears in the **Programs** list.

42. In Remote Desktop Connection, for **Computer**, paste the IP of your Windows instance that you copied.

43. Select **Connect**.

44. 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, select **Use another account**.

Note

On Windows 10, select **More Choices** before selecting **Use a different account**.

45. Go back to the lab console and copy the **WindowsPassword** to the clipboard.

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

- For **User name**, enter: `\Administrator`
- For **Password**, paste the password from the clipboard.

Note

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.

47. To connect to your instance, select **OK**. If you receive a prompt that the certificate used to verify the connection was not a known, trusted root certificate, select **Yes**.

Result

Result

Your connection to your remote instance should start momentarily. When lab instructions in subsequent sections require a command window, open or use a Powershell window.

To continue this lab, move on to [Task 2: Developing Your Application](#).

Connect to Your Windows Dev Instance from a macOS Machine

In this section, you will connect to a Windows EC2 instance from your macOS machine.

48. In the lab console, go to the **Connection Details** section and copy the **WindowsInstanceIP** to the clipboard.
49. Install Microsoft Remote Desktop if it is not already installed.
 - From the Dock, launch **App store**.
 - Search for the following string: `Microsoft Remote Desktop`
 - Select **Install**.
50. To open **Microsoft Remote Desktop**, on the Dock, select **Launchpad**. Then, select **Microsoft Remote Desktop**.
51. To create a new connection, select **New**.
52. Use the following values:

52. Use the following values:

- For **Connection name**, enter: `Windows Dev Instance`
- For **PC Name**, paste in the IP address of your Windows Server instance that you copied to the clipboard.
- For **User name**, enter: `\Administrator`

53. Go back to the lab console and copy the **WindowsPassword** to the clipboard.

54. Go back to your Microsoft Remote Desktop connection window and enter the following value:

- For **Password**, paste in the password that you copied to the clipboard.

Note

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.

55. Close the *Edit Remote Desktops* window by selecting the button on the top left corner.

56. In the *Microsoft Remote Desktop* window, select the connection titled **Windows Dev Instance** and select **Start**.

57. In the *Verify Certificate* dialog, select **Continue** to complete the connection.

Result

Your connection to your remote instance should start momentarily. When lab instructions in subsequent sections require a command window, open or use a Powershell window.

To continue this lab, move on to [Task 2: Developing Your Application](#).

Appendix: Configuring AWS SDK for .NET Credentials

The following resources are excerpts from the [AWS SDK for .NET Developer Guide](#).

AWS Credentials

You can manage credentials for your AWS SDK for .NET application in the following ways:

- [Using the SDK Store](#)
- [Using a Credentials File](#) (`C:\Users\<username>\.aws\credentials`)

For more information, see [AWS Access Keys Best Practices](#).

Appendix: Working with Lab Skeleton Code in .NET

Follow the instructions in this section to import your lab skeleton code into the **Visual Studio IDE**.

58. Launch the Visual Studio IDE by using the desktop shortcut in **Windows Dev Instance**.

59. Select through the prompts to setup Visual Studio for first time use

59. Select through the prompts to setup Visual Studio for first time use.

60. In the **File** menu, select `Open -> Project/Solution` .

61. In the **Open Project** dialog, navigate to the `C:\temp\workdir` folder. Select the lab folder. Examples of lab folder names include:

- `readySetGoCSharp`
- `s3CSharpLab` .

62. Select the solution file and choose **Open**.

To continue this lab, move on to [Task 2: Developing Your Application](#).
