Managing Storage

This lab is divided into two parts:

- In the **Task** portion of this lab, you will create:
  - Amazon EC2 instance
  - Setup AWS CLI
  - Create snapshots
  - Upload log files to Amazon S3
- In the **Challenge** portion of this lab, you will be challenged to synchronize contents of a local directory to an Amazon S3 bucket

Objectives
After completing this lab, you will be able to:

- Create and maintain snapshots for Amazon EC2 instances
- Upload files to and download files from Amazon S3

Duration

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

Accessing the AWS Management Console

1. At the top of these instructions, click Start Lab to launch your lab.

   A Start Lab panel opens displaying the lab status.

2. Wait until you see the message "**Lab status: ready**", then click the **X** to close the Start Lab panel.
3. At the top of these instructions, click AWS

   This will open the AWS Management Console in a new browser tab. The system will automatically log you in.

   Tip: If a new browser tab does not open, there will typically be a banner or icon at the top of your browser indicating that your browser is preventing the site from opening pop-up windows. Click on the banner or icon and choose "Allow pop ups."

4. Arrange the AWS Management Console tab so that it displays along side these instructions. Ideally, you will be able to see both browser tabs at the same time, to make it easier to follow the lab steps.

   ⚠ Please do not change the Region during this lab.

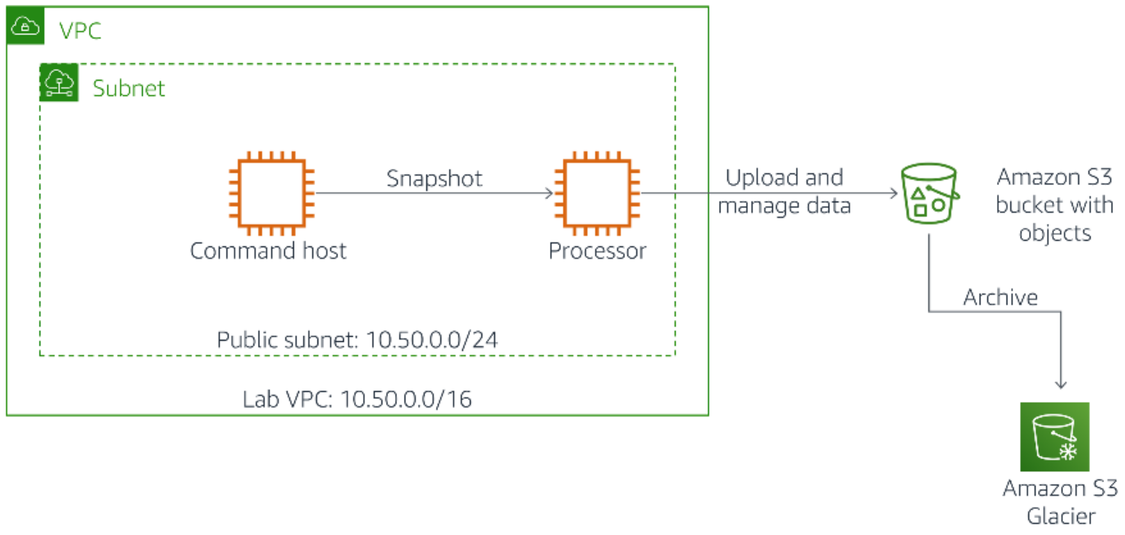
Task 1: Creating and Configuring Resources

Scenario

Your lab environment (pictured below) consists of an Amazon VPC instance called Lab VPC, which currently contains a single public subnet. Amazon EC2 instances named *Command Host* and *Processor* have already been created for you as part of this lab.

The *Command Host* will be used to administer AWS resources including the *Processor*.

In this task, you will configure AWS CLI installed on the *Command Host* to create Amazon EBS volume Snapshots for an instance labelled as *Processor*. You will then set up processes on the instance for retrieving data from and uploading data to Amazon S3.



Create an Amazon S3 bucket

In this subtask, you will create an Amazon S3 bucket

5. On the **AWS Management Console**, on the **Services** menu, click **S3**.
6. Click **Create bucket**.
7. In the **Create bucket** dialog box, configure:
   - **Bucket name**: Type a bucket name that will be unique across Amazon S3. This value will be referred to as *s3-bucket-name* in subsequent procedures. Make a note of the *s3-bucket-name* for future use.
   - **Region**: Leave as default.
8. Click **Create bucket**.

Attach Instance Profile to Processor

In this section you will attach a pre-created IAM Role as an Instance Profile to the Processor Host, giving it the permissions to interact with your Amazon S3 bucket.

9. On the **Services** menu, click **EC2**.

10. In the navigation pane, click **Instances**.
11. Select the **Processor**.
12. Click on **Actions** then **Security**, followed by **Modify IAM role**.
13. Select the `S3BucketAccess` role under **IAM role**.
14. Click **Save**.

Task 2: Taking Snapshots of Your Instance

In this section, you will learn how to use the AWS Command Line Interface (CLI) to manage the processing of snapshots of an instance.

Your AWS account is limited in any region to holding 10,000 snapshots. Furthermore, you are charged every month per gigabyte of snapshot data that you store. This charge is minimized by the fact that AWS takes incremental snapshots of your instances after the first snapshot, and also by the fact that snapshot data is compressed. However, to optimize both maintenance and cost, we recommend that you monitor the number of snapshots stored for each instance and routinely delete old snapshots that you no longer need.

Connect to the Command Host

The following instructions now vary slightly depending on whether you are using Windows or Mac/Linux.

Windows Users: Using SSH to Connect

These instructions are for Windows users only.

If you are using macOS or Linux, skip to the next section.

15. In the **AWS Management Console**, on the **Services** menu, click **EC2**.
16. In the left navigation pane, click **Instances**.
17. Select the **Command Host**.
18. Copy the **IPv4 Public IP** from the Description in the lower pane.
19. Read through the three bullet points in this step before you start to complete the actions, because you will not be able see these instructions when the Details panel is open.
    - Click on the Details drop down menu above these instructions you are currently reading, and then click Show. A Credentials window will open.
    - Click on the **Download PPK** button and save the **labsuser.ppk** file. Typically your browser will save it to the Downloads directory.
    - Then exit the Details panel by clicking on the **X**.
20. Download needed software.
    - You will use **PuTTY** to SSH to Amazon EC2 instances. If you do not have PuTTY installed on your computer, download it here.

21. Open **putty.exe**
22. Configure PuTTY to not timeout:
    - Click **Connection**
    - Set **Seconds between keepalives** to `30`

    This allows you to keep the PuTTY session open for a longer period of time.
23. Configure your PuTTY session:
    - Click **Session**
    - **Host Name (or IP address):** Paste the **Public IPv4** value you copied to your clipboard earlier in the lab.
    - Back in PuTTy, in the **Connection** list, expand ⊞ **SSH**
    - Click **Auth** (don't expand it)
    - Click **Browse**
    - Browse to and select the lab#.ppk file that you downloaded
    - Click **Open** to select it
    - Click **Open**
24. Click **Yes**, to trust the host and connect to it.
25. When prompted **login as**, enter: `ec2-user`

    This will connect you to the EC2 instance.
26. Windows Users: Click here to skip ahead to the next task.

Mac and Linux Users

These instructions are for Mac/Linux users only. If you are a Windows user, skip ahead to the next task.

27. In the **AWS Management Console**, on the **Services ⌄** menu, click **EC2**.
28. In the left navigation pane, click **Instances**.
29. Select the **Command Host**.
30. Copy the **IPv4 Public IP** from the Description in the lower pane.
31. Read through the three bullet points in this step before you start to complete the actions, because you will not be able see these instructions when the Details panel is open.
    - Click on the Details drop down menu above these instructions you are currently reading, and then click Show. A Credentials window will open.
    - Click on the **Download PEM** button and save the **labsuser.pem** file.
    - Then exit the Details panel by clicking on the **X**.
32. Open a terminal window, and change directory `cd` to the directory where the labsuser.pem file was downloaded.

    For example, run this command, if it was saved to your Downloads directory:

    ```
    cd ~/Downloads
    ```

33. Change the permissions on the key to be read only, by running this command:

    ```
    chmod 400 labsuser.pem
    ```

34. Return to the terminal window and run this command (replace **<public-ip>** with the **Public IPv4** value you copied to your clipboard earlier in the lab):

    ```
    ssh -i labsuser.pem ec2-user@<public-ip>
    ```

35. Type `yes` when prompted to allow a first connection to this remote SSH server.

    Because you are using a key pair for authentication, you will not be prompted for a password.

Taking an Initial Snapshot

In this procedure, you will take an initial snapshot of the Processor instance.

To take a snapshot, you will use the **aws ec2 create-snapshot** command. Because this command takes a volume ID, you will first need to find the volume ID for the Amazon EBS volume attached to your Processor instance. To do this, use the **aws ec2 describe-instances** command.

The **aws ec2 create-snapshot** command will take a snapshot of your disk at the time that the command was issued; subsequent writes to the disk are not included in the snapshot. However, due to application and OS write caching, a snapshot on a running instance might be inconsistent and result in missing or corrupted data. Therefore, before taking the snapshot of the Processor instance, you will shut it down. This ensures a consistent snapshot.

If you are taking a snapshot of a secondary (non-root) Amazon EBS volume, you can also unmount the volume before taking a snapshot to ensure that you get a consistent copy. To back up database systems (e.g., MySQL), you can freeze the file system to suspend write operations or enable replication and take periodic backups of your read replica.

36. To get a full description of the Processor instance, copy the following command and run it from within your instance:

    ```
    aws ec2 describe-instances --filter 'Name=tag:Name,Values=Processor'
    ```

    This command uses the **--filter** tag to limit the results description to the new instance that you created in the previous section. The command will respond with a full, JSON-based description of the instance and all of its attributes. You will now modify this command to return just the subset of data—the Amazon EBS volume information—that you are interested in.

37. To narrow down the results of the previous command further, copy the following command and run it from within your instance. If the command returns *null*, change *Reservations[0]* to *Reservations[1]*:

```
aws ec2 describe-instances --filter 'Name=tag:Name,Values=Processor' --query
'Reservations[0].Instances[0].BlockDeviceMappings[0].Ebs.{VolumeId:VolumeId}'
```

This modified command uses the --query attribute to specify a JMESPath query that returns only the volume ID of the only volume (the root volume) attached to the Processor instance. You should receive a response similar to this:

```
{

  "VolumeId": "vol-1234abcd"

}
```

This value will be referred to as volume-id in subsequent commands.

38. Before taking a snapshot, you will shut down the Processor instance, which requires its instance ID. To obtain the instance ID, copy the following command and run it from within your instance. If *Reservations[0]* was changed to *Reservations[1]* in the previous step, *Reservations[0]* will need to be changed to *Reservations[1]* for the next command:

    ```
    aws ec2 describe-instances --filters 'Name=tag:Name,Values=Processor' --query
    'Reservations[0].Instances[0].InstanceId'
    ```

    This value will be referred to as instance-id in subsequent commands.

39. To shut down the Processor instance, copy the following command, replace *INSTANCE-ID* with your instance id, and run it from within your instance:

    ```
    aws ec2 stop-instances --instance-ids INSTANCE-ID
    ```

40. Before moving to the next step in this procedure, verify that the Processor instance has stopped by running the following command, replacing *INSTANCE-ID* with your instance id. When the Processor instance has stopped, the command will return to a prompt.

    ```
    aws ec2 wait instance-stopped --instance-id INSTANCE-ID
    ```

41. To create your first snapshot of the root volume of your Processor instance, copy the following command, replace VOLUME-ID_ with your volume id, and run it in your SSH window:

    ```
    aws ec2 create-snapshot --volume-id VOLUME-ID
    ```

    The command will return a set of information that includes a **SnapshotId** value that uniquely identifies the new snapshot. This value will be referred to as **snapshot-id** in subsequent commands.

42. To check the status of your snapshot, copy the following command, replace *SNAPSHOT-ID* your **snapshot-id**, and run it in your SSH window:

```
aws ec2 wait snapshot-completed --snapshot-id SNAPSHOT-ID
```

Continue with the below procedure when the command completes.

43. To restart the Processor instance, copy the following command, replace the *INSTANCE-ID* to your instance id and run it in your SSH window:

```
aws ec2 start-instances --instance-ids INSTANCE-ID
```

44. To check on the status of the restart operation, copy the following command, replace *INSTANCE-ID* with your instance id, and run it in your SSH window:

```
aws ec2 wait instance-running --instance-id INSTANCE-ID
```

Schedule Creation of Subsequent Snapshots

Using the Linux scheduling system (cron), you can easily set up a recurring snapshot process so that new snapshots of your data are taken automatically.

For the purposes of this lab, you will schedule snapshot creation every minute so that you can verify the results of your work. In the next procedure, you will use automation to manage the number of snapshots that are maintained for a volume.

Note This section of the lab does not stop the instance in order to create a large number of snapshots for the next procedure. If you need to guarantee consistency, you can develop a fuller automation script that shuts down the instance or quiesces the disk first, as discussed in Task 2.

45. To create a cron entry that will schedule a job that runs every minute, copy the following command, replace *VOLUME-ID* with your volume-id and run it from within your instance:

```
echo "* * * * *  aws ec2 create-snapshot --volume-id VOLUME-ID 2>&1 >>
/tmp/cronlog" > cronjob
```

46. To schedule this cron task, copy the following command and run it from within your instance:

```
crontab cronjob
```

Note: This will take 1-2 minutes

47. To verify that subsequent snapshots are being created, copy the following command, replace **<volume-id>** with your volume-id and run it from within your instance:

```
aws ec2 describe-snapshots --filters "Name=volume-id,Values=<volume-id>"
```

After a few minutes, you should ideally see one or more Snapshots. If this is not working as expected, please request assistance from your instructor.

48. Wait a few minutes so that a few more snapshots will be generated before beginning the next task.

Retaining Only Last Two EBS Volume Snapshots

In this procedure, you will run a Python script that maintains only the last two snapshots for any given Amazon EBS volume associated with your account.

As discussed at the beginning of this section, aggressive snapshot management both limits your costs and simplifies management over time. Using a few lines of code, you can leverage one of the many AWS Software Development Kits (SDKs) to create a program that deletes unnecessary snapshots.

49. Use the following command to stop the cron job that you previously created:

    ```
    crontab -r
    ```

50. In the home directory of your CommandHost instance is a file named *snapshotter_v2.py*. Examine it with the following command:

    ```
    more snapshotter_v2.py
    ```

    This command is a simple script written in the Python programming language using Boto (version 3), the Python SDK for AWS. The AWS CLI is also written in Boto, which makes writing Python-powered AWS scripts very convenient because Boto is pre-installed on most Amazon EC2 Linux instances.

    The script finds all Amazon EBS volumes associated with the current user's account and takes snapshots of them. It then examines the number of snapshots associated with the volume, sorts the snapshots by date, and removes all but the two most recent snapshots.

51. Before running snapshotter_v2.py, copy the following command and run it from within your instance (replacing *VOLUME-ID* with your volume-id):

    ```
    aws ec2 describe-snapshots --filters "Name=volume-id, Values=VOLUME-ID"
    --query 'Snapshots[*].SnapshotId'
    ```

    You should see multiple snapshot IDs returned for the volume. These are the snapshots that were created by your cron job before you terminated it.

52. Run the snapshotter_v2.py script:

    ```
    python3 snapshotter_v2.py
    ```

    The script should run for a few seconds, and then return a list of all of the snapshots that it deleted:

    ```
    [ec2-user@ip-\*]$ python3 snapshotter_v2.py
    ```

```
Deleting snapshot snap-e8128a20

Deleting snapshot snap-d0d34818

Deleting snapshot snap-ded14a16

Deleting snapshot snap-e8d74c20

Deleting snapshot snap-25d54eed

Deleting snapshot snap-4acb5082
```

53. To examine the new number of snapshots for the current volume, re-run the command from the procedure above:

    ```
    aws ec2 describe-snapshots --filters "Name=volume-id, Values=<volume-id>"
    --query 'Snapshots[*].SnapshotId'
    ```

    You should see only two snapshot IDs returned.

54. Quit your SSH connection of **Command Host**.

Task 3: Challenge: Synchronize Files With Amazon S3

In this section, you will be challenged to synchronize the contents of a directory with your Amazon S3 bucket.

Note If you are already familiar with AWS, we recommend that you try this challenge yourself using the information provided in this section **before** reading the detailed solution provided in the next section. When you have completed the challenge, check your work by reviewing the detailed solution.

Challenge Description

Run this command on the EC2 instance to download a sample set of files:

```
wget https://aws-tc-largeobjects.s3.us-west-2.amazonaws.com/CUR-TF-100-RESTRT-
1/183-lab-JAWS-managing-storage/s3/files.zip
```

Unzip these files, and then, using the AWS CLI as much as possible, figure out how to accomplish the following:

- Activate versioning for your Amazon S3 bucket.
- Use a single AWS CLI command to synchronize (sync) the contents of your unzipped folder with your Amazon S3 bucket.
- Modify the command so that it deletes a file from Amazon S3 when the corresponding file is deleted locally on your instance.
- Recover the deleted file from Amazon S3 using versioning.

Hints: You can use the `aws s3api` command to enable versioning on an Amazon S3 bucket.

Solution Summary

The solution involves the following steps:

- To enable versioning for the bucket, use the `aws s3api put-bucket-versioning` command.
- To synchronize the local files with Amazon S3, use the `aws s3 sync` command on the local folder.
- Delete a local file.
- To force Amazon S3 to delete any files not present on the local drive but present in Amazon S3, use the `--delete` option to `aws s3 sync`.
- Because there is no direct command in Amazon S3 to restore an old version of a file, to download the old version of the deleted file from Amazon S3, use the `aws s3api list-object-versions` and `aws s3api get-object` commands. You can then restore the file to Amazon S3 by using another call to `aws s3 sync`.

Downloading and Unzipping Sample files

The sample file package contains a folder with three text files: file1.txt, file2.txt, and file3.txt. These are the files that you will synchronize with your Amazon S3 bucket.

55. Login to the **Processor** instance.
56. To download the sample files on the Processor instance, copy the following command and run it from within your instance:

    ```
    wget https://aws-tc-largeobjects.s3.us-west-2.amazonaws.com/CUR-TF-100-
    RESTRT-1/183-lab-JAWS-managing-storage/s3/files.zip
    ```

57. To unzip the directory, use the following command:

    ```
    unzip files.zip
    ```

Synchronizing Files

58. Before synchronizing content with your Amazon S3 bucket, you will need to enable versioning on your bucket. To enable versioning, copy the following command (replacing *S3-BUCKET-NAME* with your bucket name) and run it from within your instance:

    ```
    aws s3api put-bucket-versioning --bucket S3-BUCKET-NAME --versioning-
    configuration Status=Enabled
    ```

59. To synchronize the contents of the files folder with your Amazon S3 bucket, copy the following command (replacing *S3-BUCKET-NAME* with your bucket name) and run it from within your

instance:

```
aws s3 sync files s3://S3-BUCKET-NAME/files/
```

The command should confirm that it has copied each of the three files to your Amazon S3 bucket.

60. To confirm the state of your files, use the following command (replacing *S3-BUCKET-NAME* with your bucket name):

```
aws s3 ls s3://S3-BUCKET-NAME/files/
```

61. To delete one of the files on the local drive, use the following command:

```
rm files/file1.txt
```

62. To delete the same file from the server, use the `--delete` option to the `aws s3 sync` command. Copy the following command (replacing *S3-BUCKET-NAME* with your bucket name) and run it from within your instance:

```
aws s3 sync files s3://S3-BUCKET-NAME/files/ --delete
```

Note Depending on the version of the AWS CLI that you are using, you may see the following error:

```
delete failed: s3://custombucketname/files/file2.txt 'str' object has no
attribute 'text'
```

This is simply a parsing response error that exists in a single version of the AWS CLI; as you will confirm in the next step, the file has successfully been deleted in spite of this error.

63. Verify that the file was deleted remotely on the server:

```
aws s3 ls s3://S3-BUCKET-NAME/files/
```

64. Now, try to recover the old version of file1.txt. To view a list of past versions of this file, use the `aws s3api list-object-versions` command:

```
aws s3api list-object-versions --bucket S3-BUCKET-NAME --prefix
files/file1.txt
```

The output will contain a `DeleteMarkers` and a `Versions` block. `DeleteMarkers` indicates where the delete marker is; i.e., if you perform an `aws s3 rm` operation (or an `aws s3 sync` operation with the `--delete` option), this is the next version that the file will revert to.

The Versions block contains a list of all available versions. You should have only a single Versions entry. Find the field `VersionId` and copy its value; we will refer to this as **version-id** in the next step.

65. Because there is no direct command to restore an older version of an Amazon S3 object to its own bucket, you will need to re-download the old version and then sync again to Amazon S3. To download the previous version of *file1.txt*, copy the following command (replacing *VERSION-ID* with your version-id and *S3-BUCKET-NAME* with your bucket name) and run it from within your instance:

```
aws s3api get-object --bucket S3-BUCKET-NAME --key files/file1.txt --version-
id VERSION-ID files/file1.txt
```

66. To verify that the file has been restored locally, use the following command:

```
ls files
```

67. To re-sync the contents of the files/ folder to Amazon S3, copy the following command (replacing *S3-BUCKET-NAME* with your bucket name) and run it from within your instance:

```
aws s3 sync files s3://S3-BUCKET-NAME/files/
```

68. Finally, to verify that a new version of *file1.txt* has been pushed to Amazon S3, copy the following command (replacing *S3-BUCKET-NAME* with your bucket name) and run it from within your instance:

```
aws s3 ls s3://S3-BUCKET-NAME/files/
```

Lab Complete

Congratulations! You have completed the lab.

69. Click End Lab at the top of this page and then click **Yes** to confirm that you want to end the lab.

    A panel will appear, indicating that "DELETE has been initiated... You may close this message box now."

70. Click the **X** in the top right corner to close the panel.