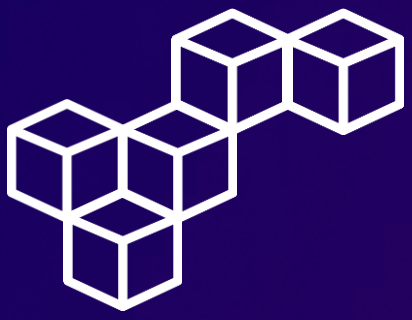
AWS re:Start LAB

Managing Storage

WEEK 10

aws re/start



Overview

Managing storage in AWS involves various strategies to ensure data persistence, backup, and recovery. One essential practice is creating and maintaining snapshots for Amazon EC2 instances. Snapshots capture a point-in-time state of an EBS volume, allowing you to back up data and restore it when needed. These snapshots can be used to create new EBS volumes or to recover from data loss, providing a robust solution for maintaining data integrity and availability.

Additionally, using Amazon S3 sync to copy files from an EBS volume to an S3 bucket is a powerful method for data backup and replication. The aws s3 sync command ensures that files are efficiently transferred and synchronized between storage solutions. Furthermore, enabling Amazon S3 versioning allows you to retrieve deleted files and previous versions, enhancing data protection. With versioning, every modification or deletion is tracked, making it possible to recover accidentally deleted or overwritten files. These combined strategies offer a comprehensive approach to managing and safeguarding your data in the AWS cloud environment.

Topics covered

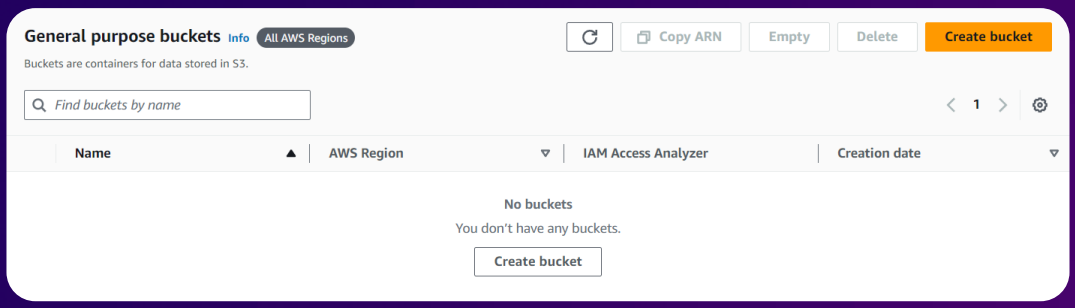
- Create and maintain snapshots for Amazon EC2 instances.
- Use Amazon S3 sync to copy files from an EBS volume to an S3 bucket.
- Use Amazon S3 versioning to retrieve deleted files.

Task 1

Creating and configuring resources

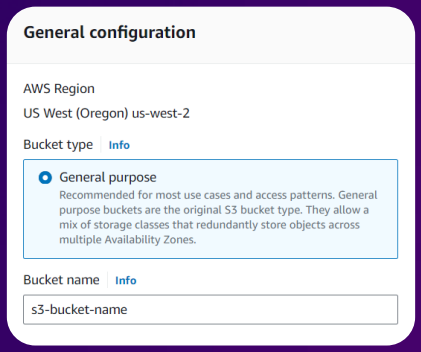
Step 1: Create bucket

On the S3 Management Console, navigate to the **Buckets** section, and select Create bucket.



Step 2: General configuration

In the **General configuration** section, configure the following settings.



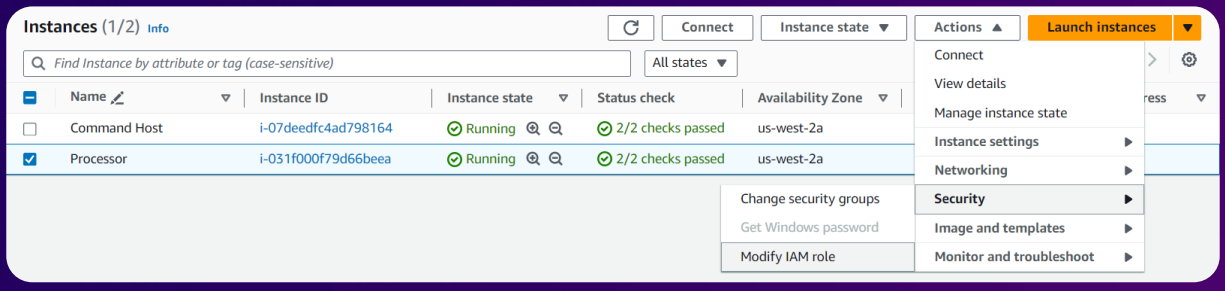
aws re/start

Task 1

Creating and configuring resources

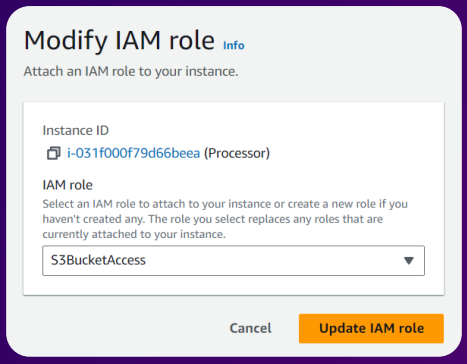
Step 3: Modify IAM role

On the EC2 Management Console, navigate to the **Instances** section, select the **Processor** instance, and choose Actions > Security > Modify IAM role.



Step 4: Update IAM role

In the **Modify IAM role** page, choose the **S3BucketAccess** role in the IAM role dropdown list. Choose Update IAM role.



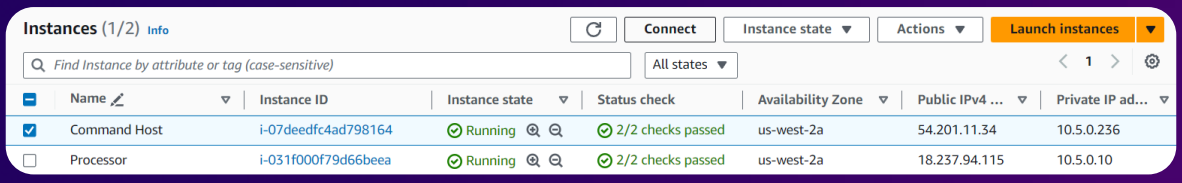
aws re/start

Task 2

Taking snapshots of your instance

Step 1: Connect to the Command Host instance

Navigate to the **Instances** section, select the **Command Host**, and connect to the instance using EC2 Instance Connect.



Step 2: Display Volume ID and Instance ID

To display the EBS Volume ID and the **Processor** Instance ID, run the following aws ec2 describe-instances commands.

```
[ec2-user@ip-10-5-0-236 ~]$ aws ec2 describe-instances \
> --filter 'Name=tag:Name,Values=Processor' \
> --query 'Reservations[0].Instances[0].BlockDeviceMappings[0].Ebs.{VolumeId:VolumeId}'
{
    "VolumeId": "vol-0004bb73c39e5fd40"
}
[ec2-user@ip-10-5-0-236 ~]$ aws ec2 describe-instances \
> --filters 'Name=tag:Name,Values=Processor' \
> --query 'Reservations[0].Instances[0].InstanceId'
"i-031f000f79d66beea"
[ec2-user@ip-10-5-0-236 ~]$
```

aws re/start

Task 2

Taking snapshots of your instance

Step 3: Stop the instance

To stop the **Processor** instance, run the following aws ec2 stop-instances command. To verify that the **Processor** instance stopped, run the following aws ec2 wait instance-stopped command.

```
[ec2-user@ip-10-5-0-236 ~]$ aws ec2 stop-instances --instance-ids i-031f000f79d66beea
{
    "StoppingInstances": [
        {
            "CurrentState": {
                "Code": 64,
                "Name": "stopping"
            },
            "InstanceId": "i-031f000f79d66beea",
            "PreviousState": {
                "Code": 16,
                "Name": "running"
            }
        }
    ]
}
[ec2-user@ip-10-5-0-236 ~]$ aws ec2 wait instance-stopped --instance-id i-031f000f79d66beea
[ec2-user@ip-10-5-0-236 ~]$
```

Step 4: Create a snapshot

To create your first snapshot of the volume of the **Processor** instance, run the following aws ec2 create-snapshot command. To check the status of your snapshot, run the following aws ec2 wait snapshot-completed command.

```
[ec2-user@ip-10-5-0-236 ~]$ aws ec2 create-snapshot --volume-id vol-0004bb73c39e5fd40
{
    "Description": "",
    "Encrypted": false,
    "OwnerId": "654654423682",
    "Progress": "",
    "SnapshotId": "snap-096f2442a947998da",
    "StartTime": "2024-05-31T02:10:41.717Z",
    "State": "pending",
    "VolumeId": "vol-0004bb73c39e5fd40",
    "VolumeSize": 8,
    "Tags": []
}
[ec2-user@ip-10-5-0-236 ~]$ aws ec2 wait snapshot-completed --snapshot-id snap-096f2442a947998da
[ec2-user@ip-10-5-0-236 ~]$
```

Task 2

Taking snapshots of your instance

Step 5: Restart the instance

To restart the **Processor** instance, run the following aws ec2 start-instances command. After a couple minutes, the instance will be in the running state.

```
[ec2-user@ip-10-5-0-236 ~]$ aws ec2 start-instances --instance-ids i-031f000f79d66beea
{
    "StartingInstances": [
        {
            "CurrentState": {
                "Code": 0,
                "Name": "pending"
            "PreviousState": {
                "Code": 80,
                "Name": "stopped"
            }
        }
    ]
}
[ec2-user@ip-10-5-0-236 ~]$
```

Step 6: Create a cron job

To create and schedule a cron entry that runs a job every minute, run the following commands.

```
[ec2-user@ip-10-5-0-236 ~]$ echo "* * * * *  aws ec2 create-snapshot --volume-id vol-0004bb73c39e5fd40 2>&1 >> /tmp/cronlog" > cronjob
[ec2-user@ip-10-5-0-236 ~]$ crontab cronjob
[ec2-user@ip-10-5-0-236 ~]$
```

Task 2

Taking snapshots of your instance

Step 7: Review snapshots

To verify that subsequent snapshots are being created, run the following aws ec2 describe-snapshots command.

```
[ec2-user@ip-10-5-0-236 ~]$ aws ec2 describe-snapshots \
--filters "Name=volume-id,Values=vol-0004bb73c39e5fd40"
{
    "Snapshots": [
        {
            "SnapshotId": "snap-0e60901411a2a8fc3",
            "State": "completed",
            "VolumeId": "vol-0004bb73c39e5fd40",
        },
        {
            "SnapshotId": "snap-012a6440230830a42",
            "State": "completed",
            "VolumeId": "vol-0004bb73c39e5fd40",
        },
        {
            "SnapshotId": "snap-0aaefc464bd38ac49",
            "State": "completed",
            "VolumeId": "vol-0004bb73c39e5fd40",
        },
        {
            "SnapshotId": "snap-00ff90d852238c5d0",
            "State": "completed",
            "VolumeId": "vol-0004bb73c39e5fd40",
        },
        {
            "SnapshotId": "snap-096f2442a947998da",
            "State": "completed",
            "VolumeId": "vol-0004bb73c39e5fd40",
        },
        {
            "SnapshotId": "snap-02ba21162518410c4",
            "State": "completed",
            "VolumeId": "vol-0004bb73c39e5fd40",
        }
    ]
}
[ec2-user@ip-10-5-0-236 ~]$
```

Step 8: Stop the cron job

To stop the cron job, run the command crontab –r.

```
[ec2-user@ip-10-5-0-236 ~]$ crontab -r
[ec2-user@ip-10-5-0-236 ~]$
```

aws re/start

Task 2

Taking snapshots of your instance

Step 9: Examine the Python script

The Python script **snapshotter_v2.py** finds all EBS volumes that are associated with the current user's account and takes snapshots. It then examines the number of snapshots that are associated with the volume, sorts the snapshots by date, and removes all but the two most recent snapshots.

```
[ec2-user@ip-10-5-0-236 ~]$ more /home/ec2-user/snapshotter_v2.py
#!/usr/bin/env python

import boto3

MAX_SNAPSHOTS = 2   # Number of snapshots to keep

# Create the EC2 resource
ec2 = boto3.resource('ec2')

# Get a list of all volumes
volume_iterator = ec2.volumes.all()

# Create a snapshot of each volume
for v in volume_iterator:
  v.create_snapshot()

  # Too many snapshots?
  snapshots = list(v.snapshots.all())
  if len(snapshots) > MAX_SNAPSHOTS:

    # Delete oldest snapshots, but keep MAX_SNAPSHOTS available
    snap_sorted = sorted([(s.id, s.start_time, s) for s in snapshots], key=lambda k: k[1])
    for s in snap_sorted[:-MAX_SNAPSHOTS]:
      print("Deleting snapshot", s[0])
      s[2].delete()
[ec2-user@ip-10-5-0-236 ~]$
```

Step 10: Review snapshot IDs

The following aws ec2 describe-snapshots command returns the multiple snapshot IDs that were returned for the volume. These snapshots were created by your cron job before you stopped it.

```
[ec2-user@ip-10-5-0-236 ~]$ aws ec2 describe-snapshots \
> --filters "Name=volume-id, Values=vol-0004bb73c39e5fd40" \
> --query 'Snapshots[*].SnapshotId'
[
    "snap-0e60901411a2a8fc3",
    "snap-012a6440230830a42",
    "snap-0aaefc464bd38ac49",
    "snap-00ff90d852238c5d0",
    "snap-096f2442a947998da",
    "snap-02ba2116251841Oc4"
]
[ec2-user@ip-10-5-0-236 ~]$
```

aws re/start

Task 2

Taking snapshots of your instance

Step 11: Run the Python script

Run the **snapshotter_v2.py** script using following command. The script runs for a few seconds, and then it returns a list of all of the snapshots that it deleted.

```
[ec2-user@ip-10-5-0-236 ~]$ python3 snapshotter_v2.py
/usr/local/lib/python3.7/site-packages/boto3/compat.py:82: PythonDeprecationWarning: Boto3 will no longer support Python 3.7 starting December 13, 2023.
To continue receiving service updates, bug fixes, and security updates please upgrade to Python 3.8 or later. More information can be found here:
https://aws.amazon.com/blogs/developer/python-support-policy-updates-for-aws-sdks-and-tools/
  warnings.warn(warning, PythonDeprecationWarning)
Deleting snapshot snap-096f2442a947998da
Deleting snapshot snap-00ff90d852238c5d0
Deleting snapshot snap-012a6440230830a42
Deleting snapshot snap-0e60901411a2a8fc3
Deleting snapshot snap-0aaefc464bd38ac49
[ec2-user@ip-10-5-0-236 ~]$
```

Step 12: Review snapshots

To examine the new number of snapshots for the current volume, re-run the following aws ec2 describe-snapshots command from an earlier step. The command returns only two snapshot IDs.

```
[ec2-user@ip-10-5-0-236 ~]$ aws ec2 describe-snapshots \
> --filters "Name=volume-id, Values=vol-0004bb73c39e5fd40" \
> --query 'Snapshots[*].SnapshotId'
[
    "snap-07cbfc1fcd495bfae",
    "snap-02ba21162518410c4"
]
[ec2-user@ip-10-5-0-236 ~]$
```

Task 3

Challenge: Synchronize files with Amazon S3

Step 1: Download and unzip sample files

Connect to the **Processor** instance using EC2 Instance Connect. Then, to download and unzip the sample files on the **Processor** instance, run the following commands from within your instance.

```
[ec2-user@ip-10-5-0-236 ~]$ wget https://aws-tc-largeobjects.s3.us-west-2.amazonaws.com/CUR-TF-100-RSJAWS-1-23732/183-lab-JAWS-managing-storage/s3/files.zip
--2024-05-31 03:15:13--  https://aws-tc-largeobjects.s3.us-west-2.amazonaws.com/CUR-TF-100-RSJAWS-1-23732/183-lab-JAWS-managing-storage/s3/files.zip
Resolving aws-tc-largeobjects.s3.us-west-2.amazonaws.com (aws-tc-largeobjects.s3.us-west-2.amazonaws.com)... 3.5.76.132, 52.218.217.81, 52.218.232.177, ...
Connecting to aws-tc-largeobjects.s3.us-west-2.amazonaws.com (aws-tc-largeobjects.s3.us-west-2.amazonaws.com)|3.5.76.132|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 72110 (70K) [application/zip]
Saving to: 'files.zip'

100%[=====================================================================================================================>] 72,110      --.-K/s   in 0.001s

2024-05-31 03:15:13 (130 MB/s) - 'files.zip' saved [72110/72110]

[ec2-user@ip-10-5-0-236 ~]$ unzip files.zip
Archive:  files.zip
  inflating: files/file1.txt
  inflating: files/file2.txt
  inflating: files/file3.txt
[ec2-user@ip-10-5-0-236 ~]$
```

Step 2: Activate versioning

To activate versioning on your bucket, run the following aws s3api put-bucket-versioning command.

```
[ec2-user@ip-10-5-0-236 ~]$ aws s3api put-bucket-versioning \
> --bucket s3-bucket-name \
> --versioning-configuration Status=Enabled
[ec2-user@ip-10-5-0-236 ~]$
```

aws re/start

Task 3

Challenge: Synchronize files with Amazon S3

Step 3: Sync files

To sync the contents of the files folder with your Amazon S3 bucket, run the following aws s3 sync command. The command confirms that three files were uploaded to your S3 bucket.

```
[ec2-user@ip-10-5-0-236 ~]$ aws s3 sync files s3://s3-bucket-name/files/
upload: files/file1.txt to s3://s3-bucket-name/files/file1.txt
upload: files/file2.txt to s3://s3-bucket-name/files/file2.txt
upload: files/file3.txt to s3://s3-bucket-name/files/file3.txt
[ec2-user@ip-10-5-0-236 ~]$
```

Step 4: Review files state

To confirm the state of your files, run the following aws s3 ls command.

```
[ec2-user@ip-10-5-0-236 ~]$ aws s3 ls s3://s3-bucket-name/files/
2024-05-31 03:18:56      30318 file1.txt
2024-05-31 03:18:56      43784 file2.txt
2024-05-31 03:18:56      96675 file3.txt
[ec2-user@ip-10-5-0-236 ~]$
```

Task 3

Challenge: Synchronize files with Amazon S3

Step 5: Delete a file from the local drive

To delete one of the files on the local drive, run the following command.

```
[ec2-user@ip-10-5-0-236 ~]$ rm files/file1.txt
[ec2-user@ip-10-5-0-236 ~]$
```

Step 6: Delete the file from the bucket

To delete the same file from the S3 bucket, use the --delete option with the aws s3 sync command.

```
[ec2-user@ip-10-5-0-236 ~]$ aws s3 sync files s3://s3-bucket-name/files/ --delete
delete: s3://s3-bucket-name/files/file1.txt
[ec2-user@ip-10-5-0-236 ~]$
```

Task 3

Challenge: Synchronize files with Amazon S3

Step 7: Verify file deletion

To verify that the file was deleted from the bucket, run the following aws s3 ls command.

```
[ec2-user@ip-10-5-0-236 ~]$ aws s3 ls s3://s3-bucket-name/files/
2024-05-31 03:18:56      43784 file2.txt
2024-05-31 03:18:56      96675 file3.txt
[ec2-user@ip-10-5-0-236 ~]$
```

Step 8: Review Versions

Try to recover the old version of **file1.txt**. To view a list of previous versions of this file, run the following aws s3api list-object-versions command. The DeleteMarkers block indicates where the delete marker is. The Versions block contains a list of all available versions. You should have only a single versions entry. Make note of the value for VersionId for use later.

```
[ec2-user@ip-10-5-0-236 ~]$ aws s3api list-object-versions --bucket s3-bucket-name --prefix files/file1.txt
{
    "Versions": [
        {
            "ETag": "\"b76b2b775023e60be16bc332496f8409\"",
            "Size": 30318,
            "StorageClass": "STANDARD",
            "Key": "files/file1.txt",
            "VersionId": "4vLDlj96_cGYM.Tvrf9Wt6rqWtxIOYMr",
            "IsLatest": false,
            "LastModified": "2024-05-31T03:18:56.000Z",
            "Owner": {
                "DisplayName": "awslabsc0w7642617t1711026801",
                "ID": "2628126b45b7bf4aa97e9f9e49a2efe046a0bf45d7767082fad949e3920b6d25"
            }
        }
    ],
    "DeleteMarkers": [
        {
            "Owner": {
                "DisplayName": "awslabsc0w7642617t1711026801",
                "ID": "2628126b45b7bf4aa97e9f9e49a2efe046a0bf45d7767082fad949e3920b6d25"
            },
            "Key": "files/file1.txt",
            "VersionId": "zJXWGGGIHvaIT3N8D5VeDLAaFTSzHx8E",
            "IsLatest": true,
            "LastModified": "2024-05-31T03:21:08.000Z"
        }
    ],
    "RequestCharged": null
}
[ec2-user@ip-10-5-0-236 ~]$
```

Task 3

Challenge: Synchronize files with Amazon S3

Step 9: Download older version

Because there's no direct command to restore an older version of an Amazon S3 object to its own bucket, you need to re-download the old version and sync again to Amazon S3. To download the previous version of **file1.txt**, run the following aws s3api get-object command.

```
[ec2-user@ip-10-5-0-236 ~]$ aws s3api get-object --bucket s3-bucket-name --key files/file1.txt --version-id 4vLDlj96_cGYM.Tvrf9Wt6rqWtxIOYMr files/file1.txt
{
    "AcceptRanges": "bytes",
    "LastModified": "Fri, 31 May 2024 03:18:56 GMT",
    "ContentLength": 30318,
    "ETag": "\"b76b2b775023e60be16bc332496f8409\"",
    "VersionId": "4vLDlj96_cGYM.Tvrf9Wt6rqWtxIOYMr",
    "ContentType": "text/plain",
    "ServerSideEncryption": "AES256",
    "Metadata": {}
}
[ec2-user@ip-10-5-0-236 ~]$
```

Step 10: Verify file restoration

To verify that the file was restored locally, run the following ls command. The command shows all three files listed.

```
[ec2-user@ip-10-5-0-236 ~]$ ls files
file1.txt  file2.txt  file3.txt
[ec2-user@ip-10-5-0-236 ~]$
```

Task 3

Challenge: Synchronize files with Amazon S3

Step 11: Re-sync files

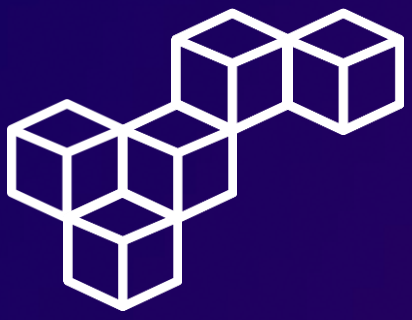
To re-sync the contents of the **files/** folder to Amazon S3, run the following aws s3 sync command.

```
[ec2-user@ip-10-5-0-236 ~]$ aws s3 sync files s3://s3-bucket-name/files/
upload: files/file1.txt to s3://s3-bucket-name/files/file1.txt
[ec2-user@ip-10-5-0-236 ~]$
```

Step 12: Review files state

To verify that a new version of **file1.txt** was pushed to the S3 bucket, run the following aws s3 ls command .

```
[ec2-user@ip-10-5-0-236 ~]$ aws s3 ls s3://s3-bucket-name/files/
2024-05-31 03:25:16      30318 file1.txt
2024-05-31 03:18:56      43784 file2.txt
2024-05-31 03:18:56      96675 file3.txt
[ec2-user@ip-10-5-0-236 ~]$
```



Conclusions

Creating Snapshots
Utilize the AWS CLI to create snapshots of Amazon EBS volumes, providing point-in-time backups and data recovery options.

Bucket Versioning
Enable versioning on Amazon S3 buckets to track changes to objects, protect against accidental deletion, and retrieve previous versions when needed.

The aws s3 sync command
Use aws s3 sync to synchronize files and directories between a local system or an EBS volume and an S3 bucket, ensuring data consistency and backup.

The aws s3 ls command
Use aws s3 ls to list objects in an S3 bucket, view their metadata, and verify the existence of files or directories.

The aws s3api commands
Utilize aws s3api commands to interact with Amazon S3 programmatically, enabling fine-grained control over bucket and object operations, such as tagging, lifecycle policies, and access permissions.

aws re/start

Cristhian Becerra

cristhian-becerra-espinoza

+51 951 634 354

cristhianbecerra99@gmail.com

Lima, Peru