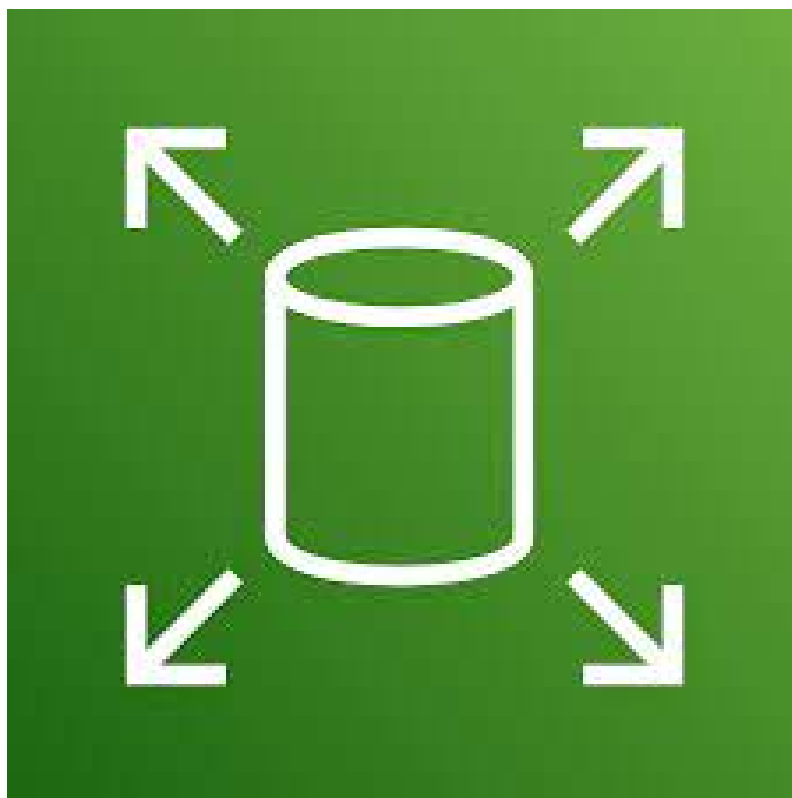




54°

Lab - AWS re/Start

Administración del almacenamiento



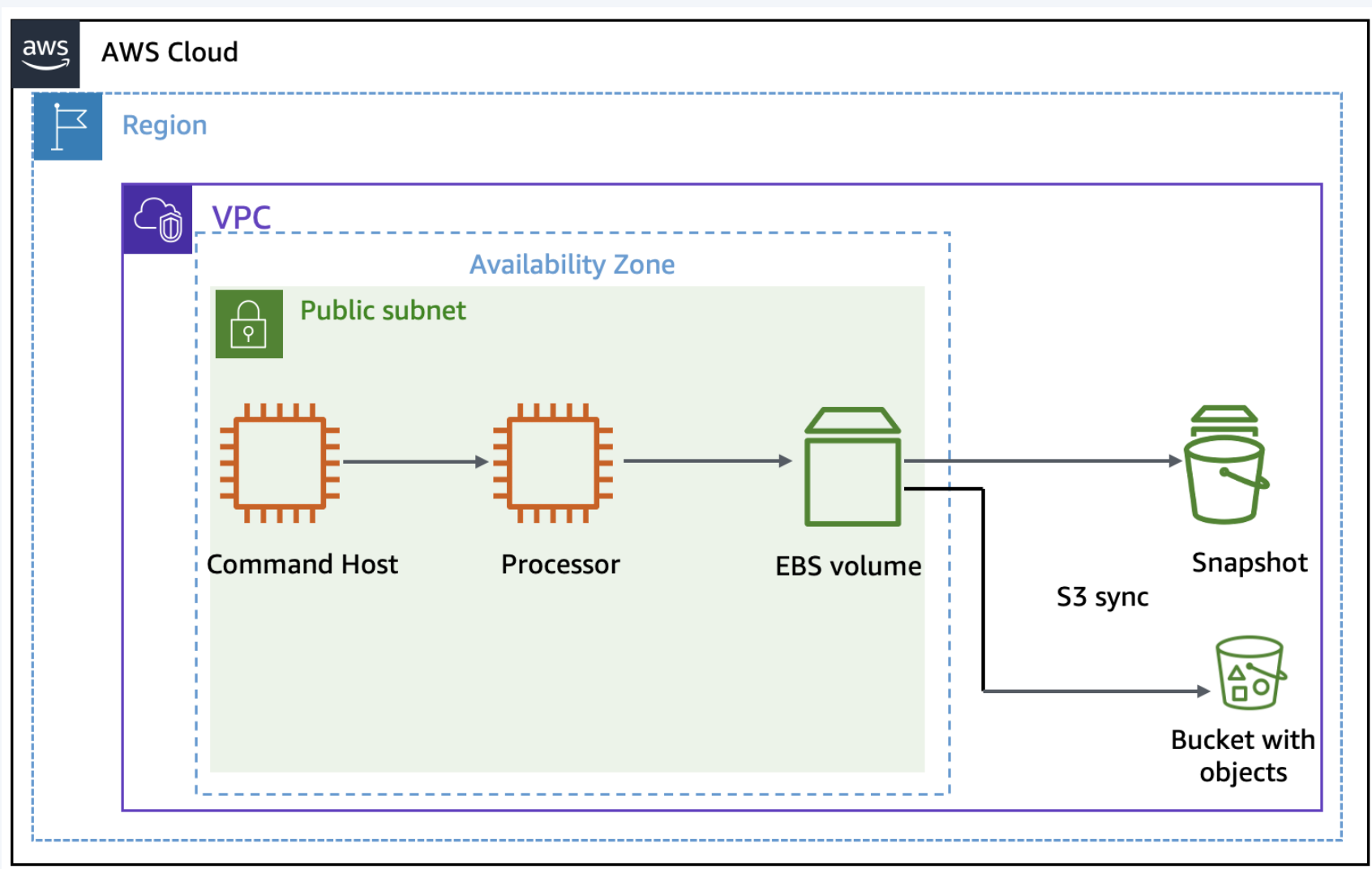
Tarea 01



Interactuando con Amazon EBS

Los objetivos son:

- Crear y mantener snapshots para instancias de EC2.
- Utilizar la sincronización de Amazon S3 para copiar archivos de un volumen de EBS a un bucket de S3.
- Utilizar el versionado de S3 para recuperar archivos eliminados.



Tarea 01



Empezamos creando un bucket de S3

[Amazon S3](#) > [Buckets](#) > Create bucket

Create bucket [Info](#)

Buckets are containers for data stored in S3.

General configuration

AWS Region
US West (Oregon) us-west-2

Bucket type [Info](#)

☒ **General purpose**
Recommended for most use cases and access patterns. General purpose buckets are the original S3 bucket type. They allow a mix of storage classes that redundantly store objects across multiple Availability Zones.

☐ **Directory - New**
Recommended for low-latency use cases. These buckets use only the S3 Express One Zone storage class, which provides faster processing of data within a single Availability Zone.

Bucket name [Info](#)
lab-mrv-volumes

Bucket name must be unique within the global namespace and follow the bucket naming rules. [See rules for bucket naming](#)

Copy settings from existing bucket - *optional*
Only the bucket settings in the following configuration are copied.

Format: s3://bucket/prefix

Asimismo, asignamos un rol a la instancia Processor que le permita interactuar con el bucket de S3

[EC2](#) > [Instances](#) > [i-0e2f8293ed0541ce5](#) > Modify IAM role

Modify IAM role [Info](#)

Attach an IAM role to your instance.

Instance ID
 i-0e2f8293ed0541ce5 (Processor)

IAM role
Select an IAM role to attach to your instance or create a new role if you haven't created any. The role you select replaces any roles that are currently attached to your instance.

S3BucketAccess

Después, procederemos a tomar el snapshot al volumen EBS que está montado en la instancia Processor

Tarea 01



Esto en la AWS CLI, OJO la instancia debe ser **detenida**

```
aws
[ec2-user@ip-10-5-0-106 ~]$ aws ec2 stop-instances --instance-ids i-0e2f8293ed0541ce5
{
  "StoppingInstances": [
    {
      "CurrentState": {
        "Code": 64,
        "Name": "stopping"
      },
      "InstanceId": "i-0e2f8293ed0541ce5",
      "PreviousState": {
        "Code": 16,
        "Name": "running"
      }
    }
  ]
}
[ec2-user@ip-10-5-0-106 ~]$ aws ec2 wait instance-stopped --instance-id i-0e2f8293ed0541ce5
[ec2-user@ip-10-5-0-106 ~]$ aws ec2 create-snapshot --volume-id vol-0827aed06e8a254dc
{
  "Description": "",
  "Encrypted": false,
  "OwnerId": "058264155507",
  "Progress": "",
  "SnapshotId": "snap-0f1033999a3755120",
  "StartTime": "2024-03-06T22:50:55.784Z",
  "State": "pending",
  "VolumeId": "vol-0827aed06e8a254dc",
  "VolumeSize": 8,
  "Tags": []
}
[ec2-user@ip-10-5-0-106 ~]$ aws ec2 wait snapshot-completed --snapshot-id vol-0827aed06e8a254dc
Waiter SnapshotCompleted failed: An error occurred (InvalidParameterValue): Value (vol-0827aed06e8a254dc) for parameter snapshotId is invalid. Expected: 'snap-...'.
[ec2-user@ip-10-5-0-106 ~]$ aws ec2 wait snapshot-completed --snapshot-id vol-0827aed06e8a254dc
Waiter SnapshotCompleted failed: An error occurred (InvalidParameterValue): Value (vol-0827aed06e8a254dc) for parameter snapshotId is invalid. Expected: 'snap-...'.
[ec2-user@ip-10-5-0-106 ~]$ aws ec2 wait snapshot-completed --snapshot-id snap-0f1033999a3755120
[ec2-user@ip-10-5-0-106 ~]$
```

Luego, volvemos a iniciar la instancia EC2. A continuación, programaremos la creación de los siguientes snapshots

```
[ec2-user@ip-10-5-0-106 ~]$ echo "* * * * * aws ec2 create-snapshot --volume-id vol-0827aed06e8a254dc 2>&1 >> /tmp/cronlog" > cronjob
[ec2-user@ip-10-5-0-106 ~]$ crontab cronjob
[ec2-user@ip-10-5-0-106 ~]$ aws ec2 describe-snapshots --filters "Name=volume-id,Values=vol-0827aed06e8a254dc"
{
  "Snapshots": [
    {
      "Description": "",
      "Encrypted": false,
      "OwnerId": "058264155507",
      "Progress": "11%",
      "SnapshotId": "snap-06f5d6531867f8e80",
      "StartTime": "2024-03-06T22:55:03.747Z",
      "State": "pending",
      "VolumeId": "vol-0827aed06e8a254dc",
      "VolumeSize": 8,
      "StorageTier": "standard"
    },
    {
      "Description": "",
      "Encrypted": false,
      "OwnerId": "058264155507",
      "Progress": "100%",
      "SnapshotId": "snap-0f1033999a3755120",
      "StartTime": "2024-03-06T22:50:55.784Z",
      "State": "completed",
      "VolumeId": "vol-0827aed06e8a254dc",
      "VolumeSize": 8,
      "StorageTier": "standard"
    }
  ]
}
[ec2-user@ip-10-5-0-106 ~]$
```

Esto lo hacemos mediante un *cron job*

Tarea 01



Para mantener los dos últimos *snapshots*:

```
[ec2-user@ip-10-5-0-106 ~]$ 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.delete()

[ec2-user@ip-10-5-0-106 ~]$ 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-0f1033999a3755120
Deleting snapshot snap-06f5d6531867f8e80
Deleting snapshot snap-0fe4acb6c85821306
[ec2-user@ip-10-5-0-106 ~]$ aws ec2 describe-snapshots --filters "Name=volume-id, Values=vol-0827aed06e8a254dc" --query 'Snapshots[*].SnapshotId'
[
  "snap-0bad7a06419211f9f",
  "snap-08de8b390087e2672"
]
[ec2-user@ip-10-5-0-106 ~]$
```

Ahora sincronizaremos estos respaldos para que se almacenen en un bucket de S3.

```
us-west-2.console.aws.amazon.com/ec2-instance-connect/ssh?connType=standard&instanceId=i-0f9ee2686242f5318&osUser=ec2-user&region=us-west-2&sshPort=22#/?
ec2-user@ip-10-5-0-106 ~]$ 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
--2024-03-06 23:03:56-- 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)... 52.92.138.10, 3.5.78.105, 52.218.170.74, ...
Connecting to aws-tc-largeobjects.s3.us-west-2.amazonaws.com (aws-tc-largeobjects.s3.us-west-2.amazonaws.com) [52.92.138.10]:443... connected.
HTTP request sent, awaiting response... 200 OK
length: 72110 (70K) [application/zip]
Saving to: 'files.zip'

00K[=====] 72,110 --.-K/s in 0s

2024-03-06 23:03:57 (197 MB/s) - 'files.zip' saved [72110/72110]

ec2-user@ip-10-5-0-106 ~]$ unzip files.zip
Archive: files.zip
  inflating: files/file1.txt
  inflating: files/file2.txt
  inflating: files/file3.txt
ec2-user@ip-10-5-0-106 ~]$ aws s3api put-bucket-versioning --bucket lab-mrv-volumesE --versioning-configuration Status=Enabled
An error occurred (NoSuchBucket) when calling the PutBucketVersioning operation: The specified bucket does not exist
ec2-user@ip-10-5-0-106 ~]$ aws s3api put-bucket-versioning --bucket lab-mrv-volumes --versioning-configuration Status=Enabled
ec2-user@ip-10-5-0-106 ~]$ aws s3 sync files s3://lab-mrv-volumes/files/
upload: files/file1.txt to s3://lab-mrv-volumes/files/file1.txt
upload: files/file3.txt to s3://lab-mrv-volumes/files/file3.txt
upload: files/file2.txt to s3://lab-mrv-volumes/files/file2.txt
ec2-user@ip-10-5-0-106 ~]$ aws s3 ls s3://lab-mrv-volumes/files/
2024-03-06 23:05:05      30318 file1.txt
2024-03-06 23:05:05      43784 file2.txt
2024-03-06 23:05:05      96675 file3.txt
ec2-user@ip-10-5-0-106 ~]$ rm files/file1.txt
ec2-user@ip-10-5-0-106 ~]$ aws s3 sync files s3://lab-mrv-volumes/files/ --delete
delete: s3://lab-mrv-volumes/files/file1.txt
ec2-user@ip-10-5-0-106 ~]$ aws s3 ls s3://lab-mrv-volumes/files/
2024-03-06 23:05:05      43784 file2.txt
2024-03-06 23:05:05      96675 file3.txt
ec2-user@ip-10-5-0-106 ~]$
```

Y también podemos ver como el versionado nos ayuda a recuperar versiones antiguas que hemos eliminado

Tarea 01



Veamos

```
[ec2-user@ip-10-5-0-106 ~]$ aws s3api list-object-versions --bucket lab-mrv-volumes --prefix files/file1.txt
{
  "Versions": [
    {
      "ETag": "\"b76b2b775023e60be16bc332496f8409\"",
      "Size": 30318,
      "StorageClass": "STANDARD",
      "Key": "files/file1.txt",
      "VersionId": "RriyRV1mqkNRpmqIePPldyM4HkRfWDXI",
      "IsLatest": false,
      "LastModified": "2024-03-06T23:05:05.000Z",
      "Owner": {
        "DisplayName": "awslabsc0w7434404t1709383734",
        "ID": "3a1c9c4ea4ecc548519a581069c5dfe839abec1008cdf2f3c97f93aa9c28478d"
      }
    }
  ],
  "DeleteMarkers": [
    {
      "Owner": {
        "DisplayName": "awslabsc0w7434404t1709383734",
        "ID": "3a1c9c4ea4ecc548519a581069c5dfe839abec1008cdf2f3c97f93aa9c28478d"
      },
      "Key": "files/file1.txt",
      "VersionId": "Kw_QRygc.2wFmpJE0xfID01JJzQH0I_b",
      "IsLatest": true,
      "LastModified": "2024-03-06T23:05:54.000Z"
    }
  ],
  "RequestCharged": null
}
```

Y recuperar dicho archivo “eliminado”

```
[ec2-user@ip-10-5-0-106 ~]$ aws s3api get-object --bucket lab-mrv-volumes --key files/file1.txt --version-id RriyRV1mqkNRpmqIePPldyM4HkRfWDXI files/file1.txt
{
  "AcceptRanges": "bytes",
  "LastModified": "Wed, 06 Mar 2024 23:05:05 GMT",
  "ContentLength": 30318,
  "ETag": "\"b76b2b775023e60be16bc332496f8409\"",
  "VersionId": "RriyRV1mqkNRpmqIePPldyM4HkRfWDXI",
  "ContentType": "text/plain",
  "ServerSideEncryption": "AES256",
  "Metadata": {}
}
[ec2-user@ip-10-5-0-106 ~]$ ls files
file1.txt  file2.txt  file3.txt
[ec2-user@ip-10-5-0-106 ~]$ aws s3 sync files s3://lab-mrv-volumes/files/
upload: files/file1.txt to s3://lab-mrv-volumes/files/file1.txt
[ec2-user@ip-10-5-0-106 ~]$ aws s3 ls s3://lab-mrv-volumes/files/
2024-03-06 23:12:39      30318 file1.txt
2024-03-06 23:05:05      43784 file2.txt
2024-03-06 23:05:05      96675 file3.txt
[ec2-user@ip-10-5-0-106 ~]$
```