AWS Production Account

Tests

Dennis Catharina Johannes Kuijs

June 3, 2025

openremote

Contents

1. Context

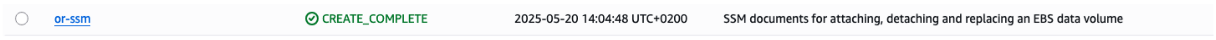
This document provides a detailed description how I tested my `EBS` data volume implementation on OpenRemote's AWS account.

2. Configure AWS account

Before running the `CI/CD` workflow, I configured the `AWS` account with the required `IAM` policies and provisioned/updated several `CloudFormation` stacks.

2.1. Provisioning / Updating CloudFormation Stacks

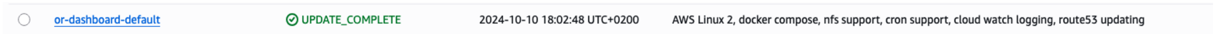
First, I provisioned the `or-ssm` CloudFormation stack to ensure the `SSM` documents are available in `Amazon Systems Manager (SSM)` . When an new AWS account is provisioned using the `provision_account` workflow, these documents are automatically created during workflow execution. In this case the AWS account was already created. Therefore, I need to add these documents manually to ensure the workflow can execute them to attach/mount the `EBS` data volume to the instance.



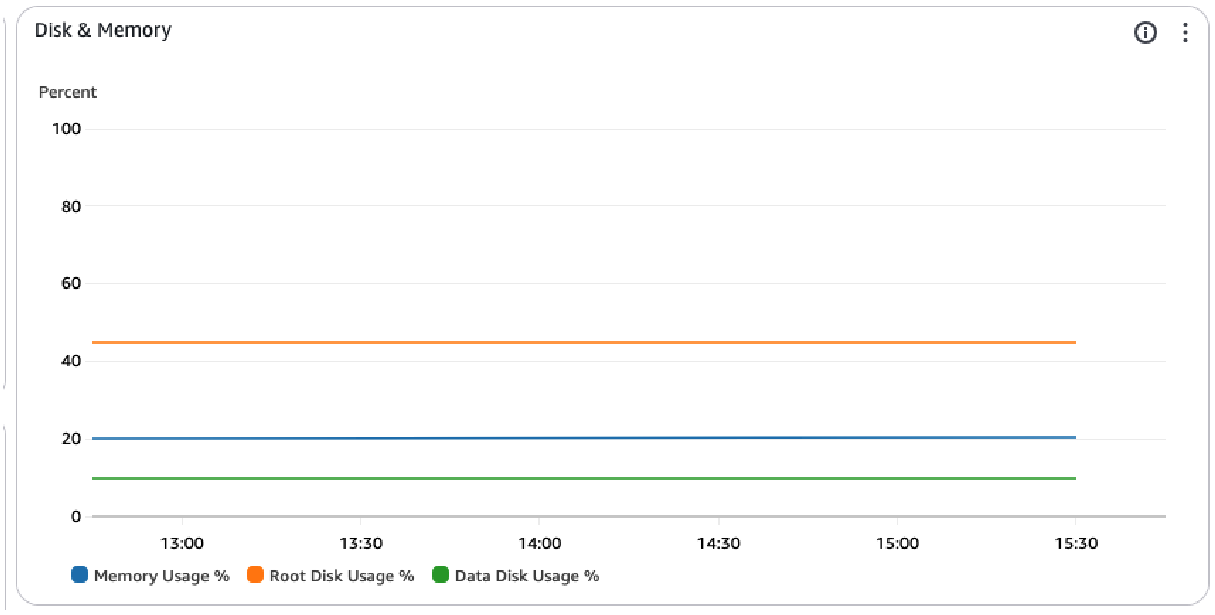
or-ssm	CREATE_COMPLETE	2025-05-20 14:04:48 UTC+0200	SSM documents for attaching, detaching and replacing an EBS data volume

Figure 1: The CloudFormation Stack or-ssm has been provisioned successfully

I also updated the existing `or-dashboard-default` `CloudFormation` stack to make the `EBS` data volume visible on the `CloudWatch` Dashboard.



or-dashboard-default	UPDATE_COMPLETE	2024-10-10 18:02:48 UTC+0200	AWS Linux 2, docker compose, nfs support, cron support, cloud watch logging, route53 updating

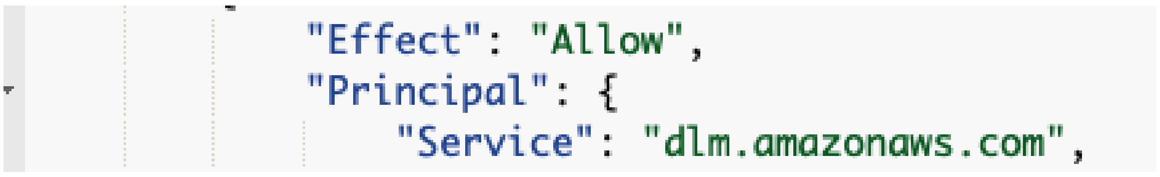


2.2. Updating IAM Roles

In the `CI/CD` workflow, I've added the feature to create an `DLM` policy for automatic snapshot creation. Before this can be provisioned, the `IAM` role that's assumed by the `CI/CD` runner needs to have the approriate permissions. I added the following permissions to the `developers-access-eu-west-1` role:

- `DLMPolicy (Inline)`
    - `dlm:CreateLifecyclePolicy` - To create the Amazon Data Lifecycle Manager policy for automatic snapshot creation.
    - `dlm:TagResource` - To tag the resources (volumes) that needs to be targeted by the `DLM` policy.
- `IAMPassRole (Inline)`
    - `arn:aws:iam::xxxxx:role/developers-access-eu-west-1` - To be able to pass this `IAM` role to the `DLM` service.
- `AWSDataLifecycleManagerServiceRole (Policy)` - To give `DLM` permissions to take actions on AWS resources, for example to create snapshots from the `EBS` data volume on behalf of the AWS user.

I also added the `DLM` service to the trusted entities to ensure `DLM` can assume this role.



```
        "Effect": "Allow",
        "Principal": {
            "Service": "dlm.amazonaws.com",
```

Figure 2: The DLM service is added to the trusted policies

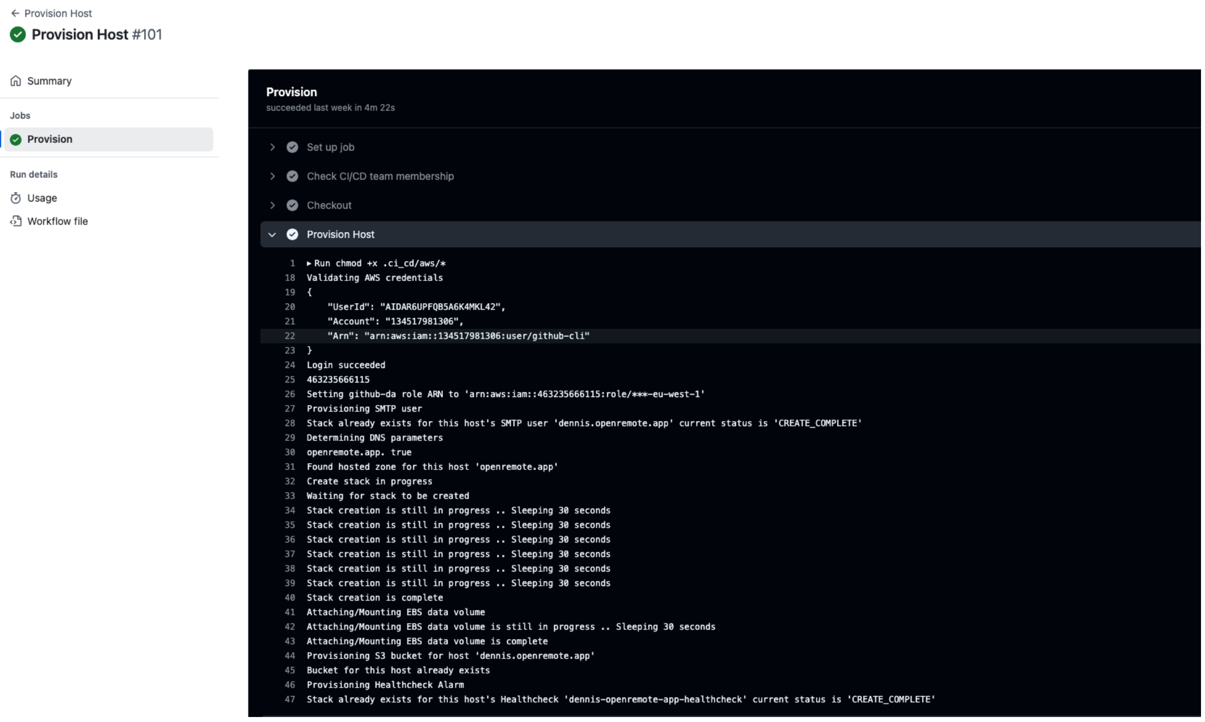
3. Provisioning host

After configuring the AWS account, I was be able to run the `provision_host` workflow with my changes. Since the implementation is not merged in the `master` branch, I need to use the following `GitHub` CLI command to run the workflow from an different branch:

```
gh workflow run "provision host" --ref feature/ebs-volume-creation --field
↪  ACCOUNT_NAME=openremote --field HOST=dennis.openremote.app
```

The workflow provisions a new host in the `openremote` AWS account with the hostname (FQDN) `dennis.openremote.app` . The following services are provisioned: - An `EC2` instance configured with `Docker` , `Docker-Compose` - An `EBS` Data Volume that's mounted to the `/var/lib/docker/volumes` directory - An `DLM` policy for automatically create snapshots from the `EBS` data volume - Several `CloudWatch` healthchecks to monitor the performance of the `EC2` instance and the OpenRemote platform - An `S3` bucket for storing the `PGDUMP` PostgreSQL backup file

After approximately 5 minutes, the workflow has finished execution and the host is ready to be used.



```
▸ Run chmod +x .ci_cd/aws/*
Validating AWS credentials
{
    "UserId": "AIDAR6UPFQB5A6K4MKL42",
    "Account": "134517981306",
    "Arn": "arn:aws:iam::134517981306:user/github-cli"
}
Login succeeded
463235666115
Setting github-da role ARN to 'arn:aws:iam::463235666115:role/***-eu-west-1'
Provisioning SMTP user
Stack already exists for this host's SMTP user 'dennis.openremote.app' current status is 'CREATE_COMPLETE'
Determining DNS parameters
openremote.app. true
Found hosted zone for this host 'openremote.app'
Create stack in progress
Waiting for stack to be created
Stack creation is still in progress .. Sleeping 30 seconds
Stack creation is still in progress .. Sleeping 30 seconds
Stack creation is still in progress .. Sleeping 30 seconds
Stack creation is still in progress .. Sleeping 30 seconds
Stack creation is still in progress .. Sleeping 30 seconds
Stack creation is still in progress .. Sleeping 30 seconds
Stack creation is complete
Attaching/Mounting EBS data volume
Attaching/Mounting EBS data volume is still in progress .. Sleeping 30 seconds
Attaching/Mounting EBS data volume is complete
Provisioning S3 bucket for host 'dennis.openremote.app'
Bucket for this host already exists
Provisioning Healthcheck Alarm
Stack already exists for this host's Healthcheck 'dennis-openremote-app-healthcheck' current status is 'CREATE_COMPLETE'
```

Figure 3: The Provision Host CI/CD workflow has been executed successfully and provisioned the EC2 instance in the AWS account

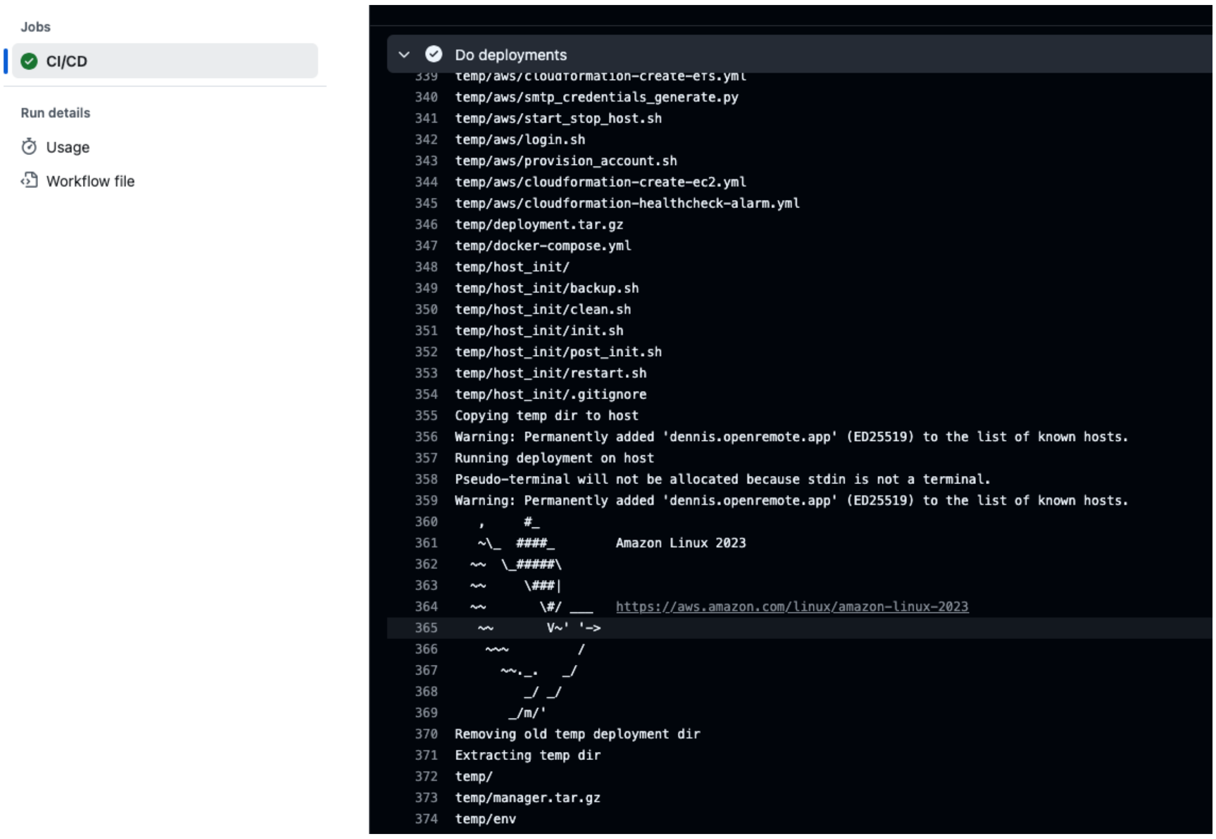
4. Tests in the AWS account

After provisioning the host in the AWS account I can start testing the `EBS` volume implementation.

4.1. Deploying OpenRemote to the new host

When the `provision_host` workflow is successfully executed, it creates an empty `EC2` instance. Before I can test my implementation I need to deploy OpenRemote on this virtual machine. I used the `CI/CD` workflow to deploy the branch `feature/edit-map-layers` to this instance. This takes around 10 minutes as it needs to build the `Docker` images first.

When this workflow is finished successfully, OpenRemote is running on the `EC2` instance and accessible using the hostname (`dennis.openremote.app`)

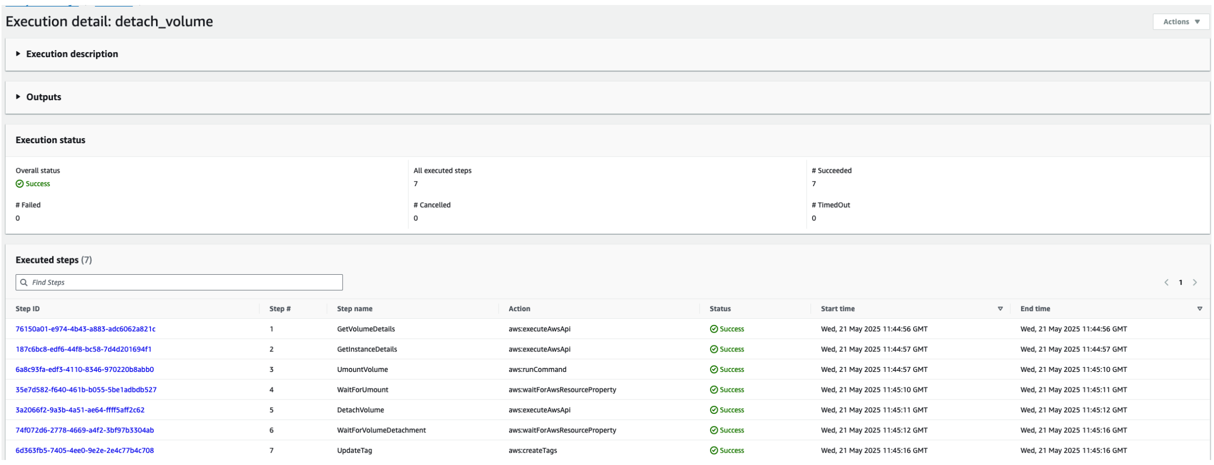


```
temp/aws/cloudformation-create-efs.yml
temp/aws/smtp_credentials_generate.py
temp/aws/start_stop_host.sh
temp/aws/login.sh
temp/aws/provision_account.sh
temp/aws/cloudformation-create-ec2.yml
temp/aws/cloudformation-healthcheck-alarm.yml
temp/deployment.tar.gz
temp/docker-compose.yml
temp/host_init/
temp/host_init/backup.sh
temp/host_init/clean.sh
temp/host_init/init.sh
temp/host_init/post_init.sh
temp/host_init/restart.sh
temp/host_init/.gitignore
Copying temp dir to host
Warning: Permanently added 'dennis.openremote.app' (ED25519) to the list of known hosts.
Running deployment on host
Pseudo-terminal will not be allocated because stdin is not a terminal.
Warning: Permanently added 'dennis.openremote.app' (ED25519) to the list of known hosts.
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
Removing old temp deployment dir
Extracting temp dir
temp/
temp/manager.tar.gz
temp/env
```

Figure 4: The CI/CD workflow has been executed successfully and deployed OpenRemote to the EC2 instance

4.2. Testing Detach Volume

First, I tested the option to detach the `EBS` volume by executing the `detach_volume` `SSM` document using the `volumeId` . After the document is successfully executed I manually checked every step to make sure the tasks are executed correctly.



Execution detail: detach_volume

Actions

Execution description

Outputs

Execution status

Overall status ✓ Success	All executed steps 7	# Succeeded 7
# Failed 0	# Cancelled 0	# TimedOut 0

Executed steps (7)

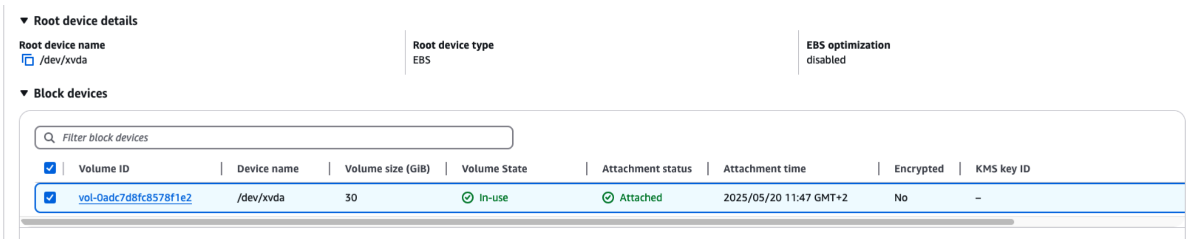
Find Steps

Step ID	Step #	Step name	Action	Status	Start time	End time
76150a01-e974-4b43-a883-adc6062a821c	1	GetVolumeDetails	aws:executeAwsApi	✓ Success	Wed, 21 May 2025 11:44:56 GMT	Wed, 21 May 2025 11:44:56 GMT
187c6bc8-edf6-44f8-bc58-7d4d201694f1	2	GetInstanceDetails	aws:executeAwsApi	✓ Success	Wed, 21 May 2025 11:44:57 GMT	Wed, 21 May 2025 11:44:57 GMT
6a8c93fa-edf3-4110-8346-970220b8abb0	3	UmountVolume	aws:runCommand	✓ Success	Wed, 21 May 2025 11:44:57 GMT	Wed, 21 May 2025 11:45:10 GMT
35e7d582-f640-461b-b055-5be1adbdb527	4	WaitForUmount	aws:waitForAwsResourceProperty	✓ Success	Wed, 21 May 2025 11:45:10 GMT	Wed, 21 May 2025 11:45:11 GMT
3a2066f2-9a3b-4a51-ae64-ffff5aff2c62	5	DetachVolume	aws:executeAwsApi	✓ Success	Wed, 21 May 2025 11:45:11 GMT	Wed, 21 May 2025 11:45:12 GMT
74f072d6-2778-4669-a4f2-3bf97b3304ab	6	WaitForVolumeDetachment	aws:waitForAwsResourceProperty	✓ Success	Wed, 21 May 2025 11:45:12 GMT	Wed, 21 May 2025 11:45:16 GMT
6d363fb5-7405-4ee0-9e2e-2e4c77b4c708	7	UpdateTag	aws:createTags	✓ Success	Wed, 21 May 2025 11:45:16 GMT	Wed, 21 May 2025 11:45:16 GMT

Figure 5: The Detach Volume SSM automation is successfully executed

4.2.1. Volume is detached from the EC2 instance

The `EBS` data volume is correctly detached from the EC2 instance. Only the `root` volume is still attached. The `EBS` data volume is also not showing up in the `block devices` list anymore.



Root device details

Root device name /dev/xvda	Root device type EBS	EBS optimization disabled

Block devices

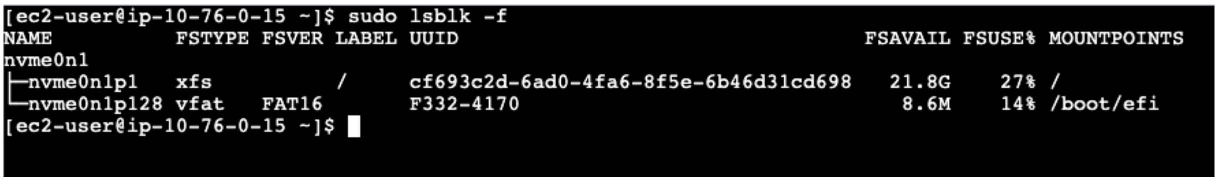
Filter block devices

✓	Volume ID	Device name	Volume size (GiB)	Volume State	Attachment status	Attachment time	Encrypted	KMS key ID
✓	vol-0adc7d8fc8578f1e2	/dev/xvda	30	✓ In-use	✓ Attached	2025/05/20 11:47 GMT+2	No	–

Figure 6: The EBS data volume is detached from the EC2 instance

4.2.2. Volume is umounted

The `EBS` data volume is successfully umounted from the `/var/lib/docker/volumes` directory. The `docker` persistent volumes are no longer available by the filesystem.

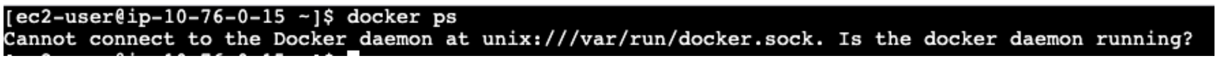


```
[ec2-user@ip-10-76-0-15 ~]$ sudo lsblk -f
NAME          FSTYPE FSVER LABEL UUID                                 FSAVAIL FSUSE% MOUNTPOINTS
nvme0n1
├─nvme0n1p1   xfs          /     cf693c2d-6ad0-4fa6-8f5e-6b46d31cd698   21.8G    27% /
└─nvme0n1p128 vfat   FAT16       F332-4170                               8.6M    14% /boot/efi
[ec2-user@ip-10-76-0-15 ~]$ 
```

Figure 7: The EBS data volume is umounted from the file system

4.2.3. Docker is successfully stopped

The `Docker` service and socket are successfully stopped. The `Docker` containers are no longer running and OpenRemote is shutdown safely.

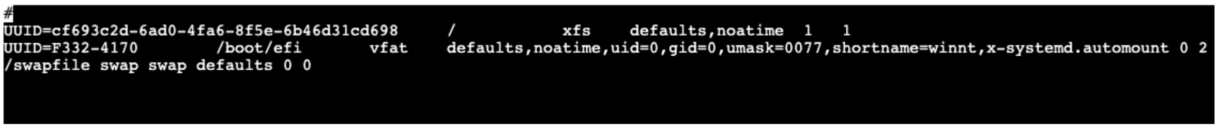


```
[ec2-user@ip-10-76-0-15 ~]$ docker ps
Cannot connect to the Docker daemon at unix:///var/run/docker.sock. Is the docker daemon running?
```

Figure 8: The Docker service and socket are successfully stopped

4.2.4. Entry in the File Systems Table

When the `EBS` volume is successfully detached, the system has removed the entry from the file systems table in the `/etc/fstab` file.

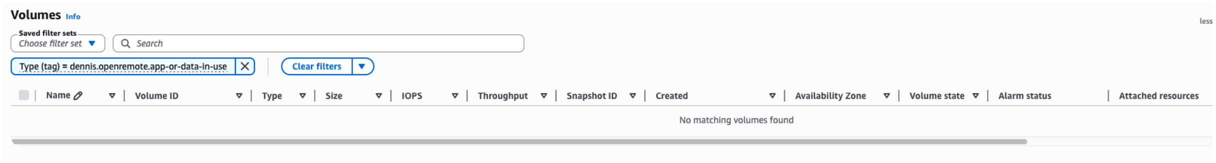


```
#
UUID=cf693c2d-6ad0-4fa6-8f5e-6b46d31cd698     /           xfs    defaults,noatime  1   1
UUID=F332-4170        /boot/efi      vfat    defaults,noatime,uid=0,gid=0,umask=0077,shortname=winnt,x-systemd.automount 0 2
/swapfile swap swap defaults 0 0
```

Figure 9: The EBS data volume is removed from the File Systems Table

4.2.5. Volume not targeted by DLM Policy

The tag gets updated to `or-data-not-in-use` to make sure the `EBS` data volume is no longer targeted by the `DLM` policy. The policy only needs to target the `EBS` data volume that is currently attached to the instance.



Volumes Info

Saved filter sets: Choose filter set

Type (tag) = dennis.openremote.app-or-data-in-use

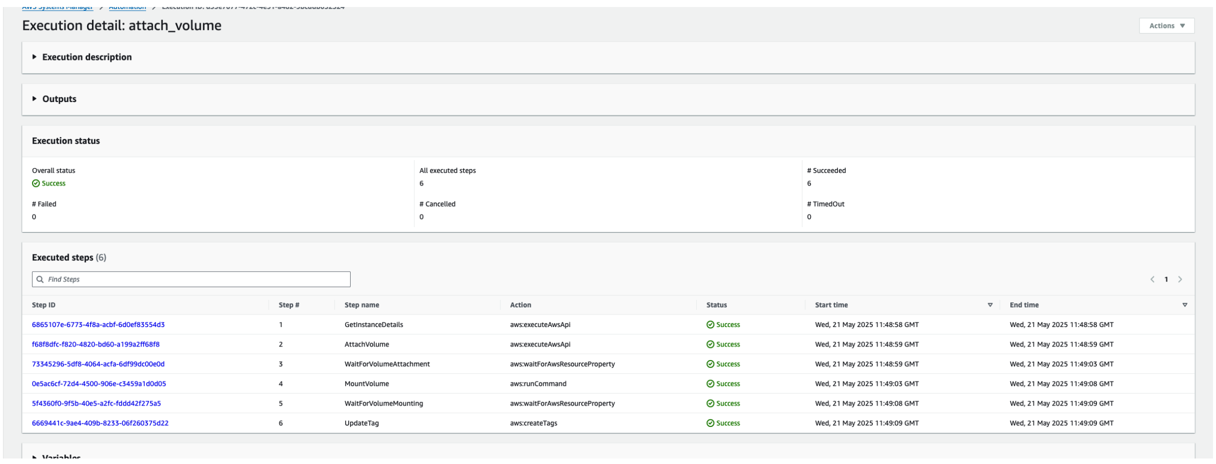
Name	Volume ID	Type	Size	IOPS	Throughput	Snapshot ID	Created	Availability Zone	Volume state	Alarm status	Attached resources

No matching volumes found

Figure 10: The EBS data volume is not targeted by the DLM policy anymore

4.3. Testing Attach Volume

When the `EBS` volume is successfully detached from the `EC2` instance I start testing the possibility to attach the `EBS` volume again using the `attach_volume` `SSM` document. After the document is successfully executed I manually go through every step to ensure it's processed correctly.



Execution detail: attach_volume

Execution description

Outputs

Execution status

Overall status: Success
All executed steps: 6
Succeeded: 6
Failed: 0
Cancelled: 0
TimedOut: 0

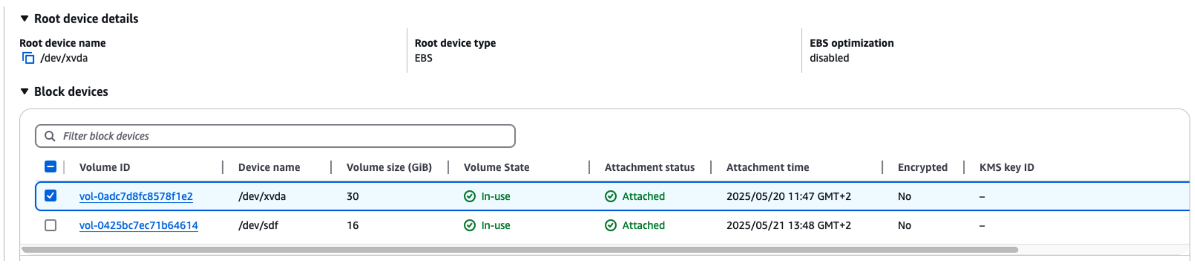
Executed steps (6)

Step ID	Step #	Step name	Action	Status	Start time	End time
6865107e-6773-4f8a-acbf-6d0ef83554d3	1	GetInstanceDetails	aws:executeAwsApi	Success	Wed, 21 May 2025 11:48:58 GMT	Wed, 21 May 2025 11:48:58 GMT
f68f8dfc-f820-4820-bd60-a199a2ff68f8	2	AttachVolume	aws:executeAwsApi	Success	Wed, 21 May 2025 11:48:59 GMT	Wed, 21 May 2025 11:48:59 GMT
73345296-5df8-4064-acfa-6df99dc00e0d	3	WaitForVolumeAttachment	aws:waitForAwsResourceProperty	Success	Wed, 21 May 2025 11:48:59 GMT	Wed, 21 May 2025 11:49:03 GMT
0e5ac6cf-72d4-4500-906e-c3459a1d0d05	4	MountVolume	aws:runCommand	Success	Wed, 21 May 2025 11:49:03 GMT	Wed, 21 May 2025 11:49:08 GMT
5f4360f0-9f5b-40e5-a2fc-fddd42f275a5	5	WaitForVolumeMounting	aws:waitForAwsResourceProperty	Success	Wed, 21 May 2025 11:49:08 GMT	Wed, 21 May 2025 11:49:09 GMT
6669441c-9ae4-409b-8233-06f260375d22	6	UpdateTag	aws:createTags	Success	Wed, 21 May 2025 11:49:09 GMT	Wed, 21 May 2025 11:49:09 GMT

Figure 11: The Attach Volume SSM automation is successfully executed

4.3.1. Volume is attached to the EC2 instance

The `EBS` data volume is successfully attached to the `EC2` instance.



Root device details

Root device name: /dev/xvda
Root device type: EBS
EBS optimization: disabled

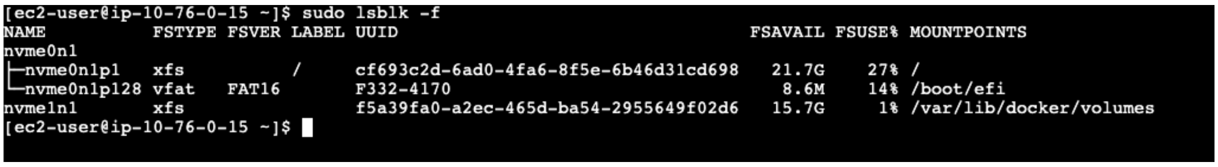
Block devices

Volume ID	Device name	Volume size (GiB)	Volume State	Attachment status	Attachment time	Encrypted	KMS key ID
vol-0adc7d8fc8578f1e2	/dev/xvda	30	In-use	Attached	2025/05/20 11:47 GMT+2	No	–
vol-0425bc7ec71b64614	/dev/sdf	16	In-use	Attached	2025/05/21 13:48 GMT+2	No	–

Figure 12: The EBS data volume is attached to the EC2 instance

4.3.2. Volume is mounted

The `EBS` data volume is successfully mounted to the `/var/lib/docker/volumes` directory.

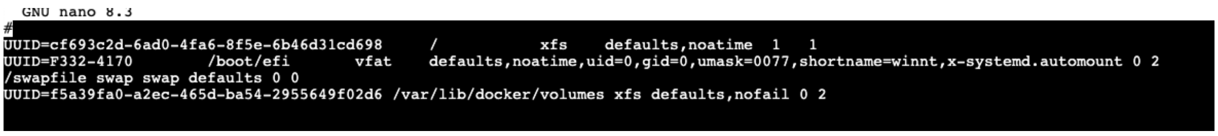


```
[ec2-user@ip-10-76-0-15 ~]$ sudo lsblk -f
NAME          FSTYPE FSVER LABEL UUID                                 FSAVAIL FSUSE% MOUNTPOINTS
nvme0n1
├─nvme0n1p1   xfs          /     cf693c2d-6ad0-4fa6-8f5e-6b46d31cd698   21.7G    27% /
└─nvme0n1p128 vfat   FAT16       F332-4170                               8.6M    14% /boot/efi
nvme1n1       xfs                f5a39fa0-a2ec-465d-ba54-2955649f02d6   15.7G     1% /var/lib/docker/volumes
[ec2-user@ip-10-76-0-15 ~]$ █
```

Figure 13: The EBS data volume is mounted to the Docker volumes directory

4.3.3. Entry in the File Systems Table

After successfully attaching the `EBS` data volume to the `EC2` instance, the script will add the `block device` to the file systems table in the `/etc/fstab` file.

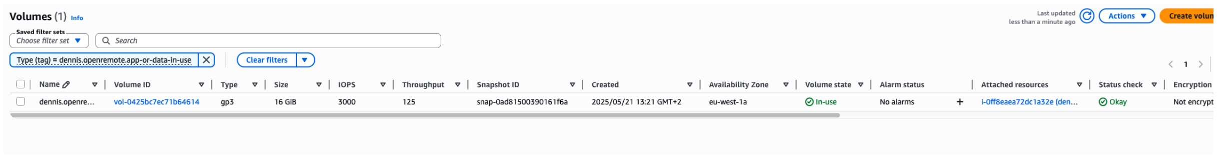


```
  GNU nano 8.3
#
UUID=cf693c2d-6ad0-4fa6-8f5e-6b46d31cd698     /           xfs    defaults,noatime  1    1
UUID=F332-4170        /boot/efi       vfat    defaults,noatime,uid=0,gid=0,umask=0077,shortname=winnt,x-systemd.automount 0 2
/swapfile swap swap defaults 0 0
UUID=f5a39fa0-a2ec-465d-ba54-2955649f02d6 /var/lib/docker/volumes xfs defaults,nofail 0 2
```

Figure 14: The EBS data volume is added to the File Systems Table

4.3.4. Volume is targeted by DLM Policy

The script has updated the tag to `or-data-in-use` to make sure the `EBS` volume is targeted by the `DLM` policy again. `DLM` will now create automatic snapshots for this volume.



Volumes (1) Info

Saved filter sets: Choose filter set

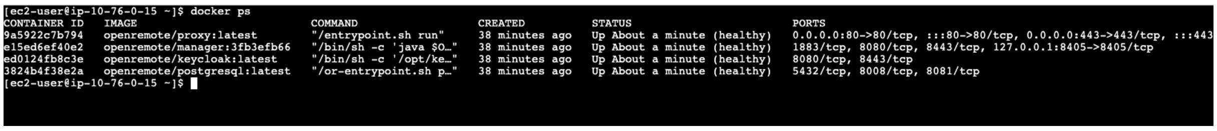
Type (tag) = dennis.openremote.app-or-data-in-use

Name	Volume ID	Type	Size	IOPS	Throughput	Snapshot ID	Created	Availability Zone	Volume state	Alarm status	Attached resources	Status check	Encryption
dennis.openre...	vol-0425bc7ec71b64614	gp3	16 GiB	3000	125	snap-0ad81500390161f6a	2025/05/21 13:21 GMT+2	eu-west-1a	In-use	No alarms	i-0ff8eaea72dc1a32e (den...	Okay	Not encrypt

Figure 15: The EBS Data volume is targeted by the DLM Policy

4.3.5. Docker is successfully started

The script enables the `Docker` socket and service. The existing containers are automatically trying to boot up. After a few minutes all the containers became healthy and OpenRemote is accesible.



```
[ec2-user@ip-10-76-0-15 ~]$ docker ps
CONTAINER ID   IMAGE                         COMMAND                  CREATED          STATUS                        PORTS
9a5922c7b794   openremote/proxy:latest       "/entrypoint.sh run"     38 minutes ago   Up About a minute (healthy)   0.0.0.0:80->80/tcp, :::80->80/tcp, 0.0.0.0:443->443/tcp, :::443
e15ed6ef40e2   openremote/manager:3fb3efb66  "/bin/sh -c 'java $O…"   38 minutes ago   Up About a minute (healthy)   1883/tcp, 8080/tcp, 8443/tcp, 127.0.0.1:8405->8405/tcp
ed0124fb8c3e   openremote/keycloak:latest    "/bin/sh -c '/opt/ke…"   38 minutes ago   Up About a minute (healthy)   8080/tcp, 8443/tcp
3824b4f38e2a   openremote/postgresql:latest  "/or-entrypoint.sh p…"   38 minutes ago   Up About a minute (healthy)   5432/tcp, 8008/tcp, 8081/tcp
[ec2-user@ip-10-76-0-15 ~]$ █
```

Figure 16: The Docker containers are healthy

When visiting the OpenRemote platform, all the data is visible and the platform is working as expected.

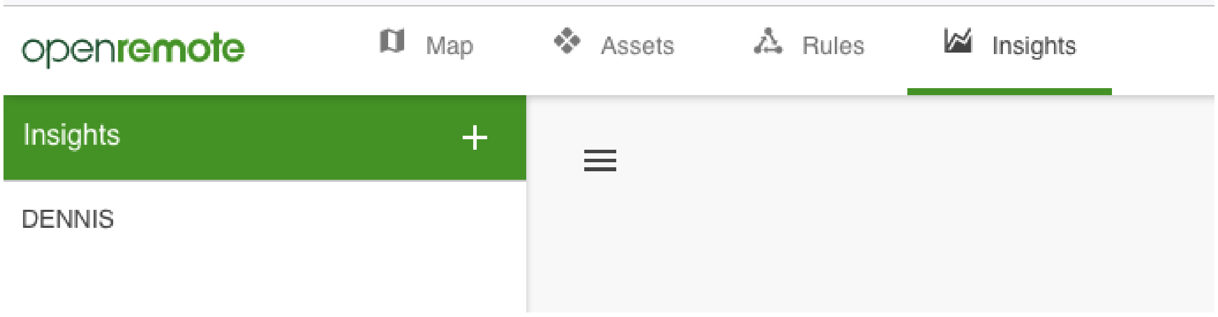
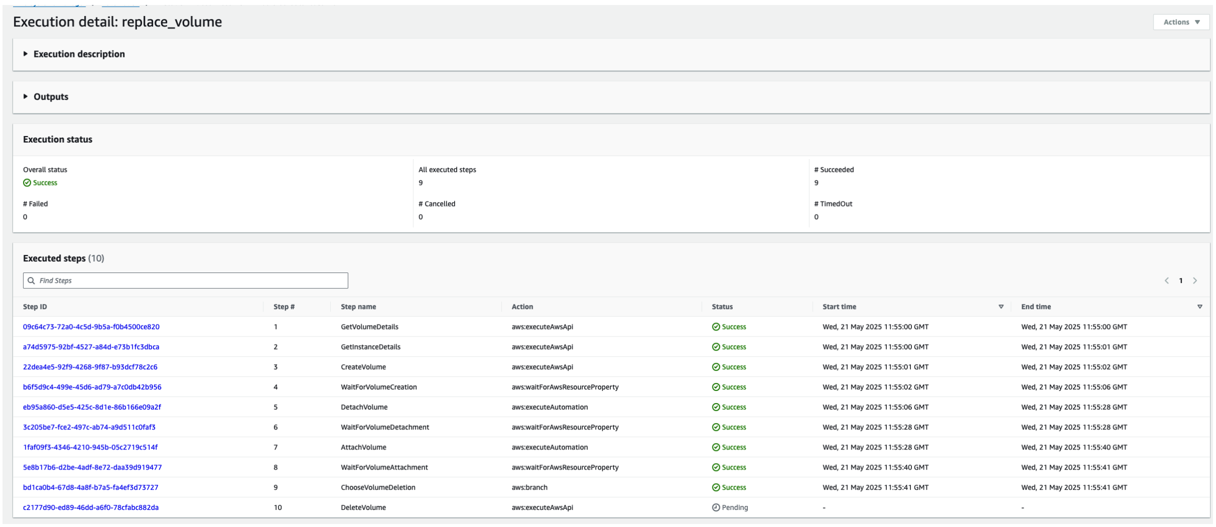


Figure 17: The IoT data is successfully loaded and available in the platform

4.4. Testing Replace Volume with/without Volume Deletion

In this section, I tested the option to replace an existing `EBS` data volume with an snapshot using the `replace_volume` `SSM` document. In this example, the script is configured to keep the original `EBS` data volume. After successfully executed the document, I checked every step manually to make sure all the tasks are executed properly.


Execution detail: replace_volume

Execution description

Outputs

Execution status

Overall status Success	All executed steps 9	# Succeeded 9
# Failed 0	# Cancelled 0	# TimedOut 0

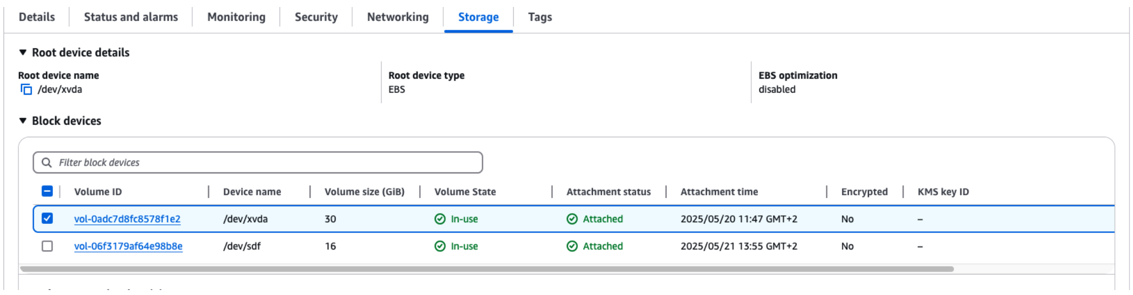
Executed steps (10)

Step ID	Step #	Step name	Action	Status	Start time	End time
09c64c73-72a0-4c5d-9b5a-f0b4500ce820	1	GetVolumeDetails	aws:executeAwsApi	Success	Wed, 21 May 2025 11:55:00 GMT	Wed, 21 May 2025 11:55:00 GMT
a74d5975-92bf-4527-a84d-e73b1fc3dbca	2	GetInstanceDetails	aws:executeAwsApi	Success	Wed, 21 May 2025 11:55:00 GMT	Wed, 21 May 2025 11:55:01 GMT
22dea4e5-92f9-4268-9f87-b93dcf78c2c6	3	CreateVolume	aws:executeAwsApi	Success	Wed, 21 May 2025 11:55:01 GMT	Wed, 21 May 2025 11:55:02 GMT
b6f5d9c4-499e-45d6-ad79-a7c0db42b956	4	WaitForVolumeCreation	aws:waitForAwsResourceProperty	Success	Wed, 21 May 2025 11:55:02 GMT	Wed, 21 May 2025 11:55:06 GMT
eb95a860-d5e5-425c-8d1e-86b166e09a2f	5	DetachVolume	aws:executeAutomation	Success	Wed, 21 May 2025 11:55:06 GMT	Wed, 21 May 2025 11:55:28 GMT
3c205be7-fce2-497c-ab74-a9d511c0faf3	6	WaitForVolumeDetachment	aws:waitForAwsResourceProperty	Success	Wed, 21 May 2025 11:55:28 GMT	Wed, 21 May 2025 11:55:28 GMT
1faf09f3-4346-4210-945b-05c2719c514f	7	AttachVolume	aws:executeAutomation	Success	Wed, 21 May 2025 11:55:28 GMT	Wed, 21 May 2025 11:55:40 GMT
5e8b17b6-d2be-4adf-8e72-daa39d919477	8	WaitForVolumeAttachment	aws:waitForAwsResourceProperty	Success	Wed, 21 May 2025 11:55:40 GMT	Wed, 21 May 2025 11:55:41 GMT
bd1ca0b4-67d8-4a8f-b7a5-fa4ef3d73727	9	ChooseVolumeDeletion	aws:branch	Success	Wed, 21 May 2025 11:55:41 GMT	Wed, 21 May 2025 11:55:41 GMT
c2177d90-ed89-46dd-a6f0-78cfabc882da	10	DeleteVolume	aws:executeAwsApi	Pending	-	-

Figure 18: The Replace Volume SSM automation is successfully executed without deleting the current EBS data volume

4.4.1. Create new volume from snapshot

The script creates an new `EBS` data volume based off an existing snapshot an attaches this volume to the `EC2` instance. The existing `EBS` data volume will be detached from the instance.



▼ Root device details

Root device name	Root device type	EBS optimization
/dev/xvda	EBS	disabled

▼ Block devices

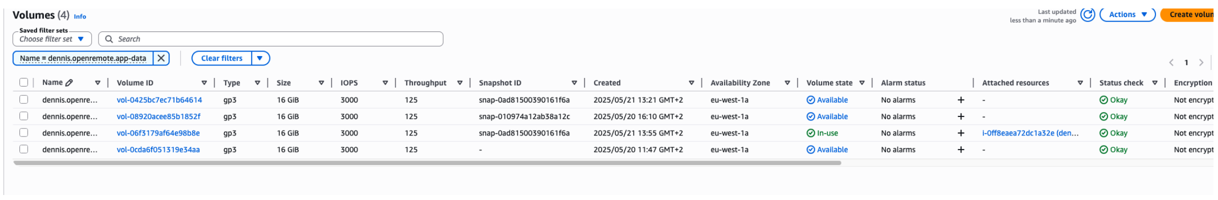
Filter block devices

	Volume ID	Device name	Volume size (GiB)	Volume State	Attachment status	Attachment time	Encrypted	KMS key ID
✓	vol-0adc7d8fc8578f1e2	/dev/xvda	30	In-use	Attached	2025/05/20 11:47 GMT+2	No	–
☐	vol-06f3179af64e98b8e	/dev/sdf	16	In-use	Attached	2025/05/21 13:55 GMT+2	No	–

Figure 19: The current EBS data volume is replaced with the new EBS data volume that is based off an existing snapshot

4.4.2. Old volume detached

The current `EBS` data volume is successfully detached from the `EC2` instance and is visible in the `volumes` overview

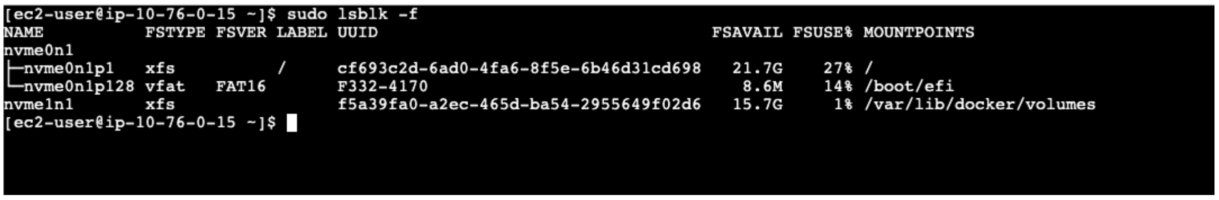


	Name	Volume ID	Type	Size	IOPS	Throughput	Snapshot ID	Created	Availability Zone	Volume state	Alarm status		Attached resources	Status check	Encryption
☐	dennis.openre...	vol-0425bc7ec71b64614	gp3	16 GiB	3000	125	snap-0ad81500390161f6a	2025/05/21 13:21 GMT+2	eu-west-1a	Available	No alarms	+	-	Okay	Not encrypt
☐	dennis.openre...	vol-08920acee85b1852f	gp3	16 GiB	3000	125	snap-010974a12ab38a12c	2025/05/20 16:10 GMT+2	eu-west-1a	Available	No alarms	+	-	Okay	Not encrypt
☐	dennis.openre...	vol-06f3179af64e98b8e	gp3	16 GiB	3000	125	snap-0ad81500390161f6a	2025/05/21 13:55 GMT+2	eu-west-1a	In-use	No alarms	+	i-0ff8eaea72dc1a32e (den...	Okay	Not encrypt
☐	dennis.openre...	vol-0cda6f051319e34aa	gp3	16 GiB	3000	125	-	2025/05/20 11:47 GMT+2	eu-west-1a	Available	No alarms	+	-	Okay	Not encrypt

Figure 20: The current EBS data volume is detached from the EC2 instance

4.4.3. New volume mounted

The newly created `EBS` data volume is mounted to the `/var/lib/docker/volumes` directory. The snapshot data (docker volumes) are available in this directory.

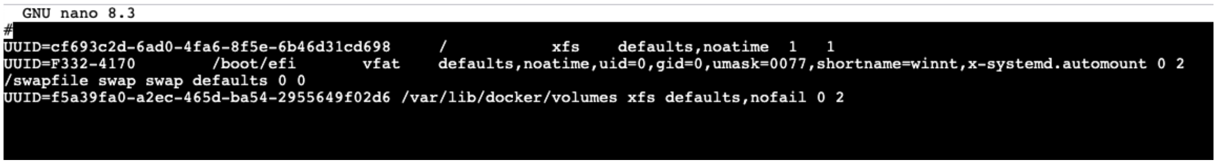


```
[ec2-user@ip-10-76-0-15 ~]$ sudo lsblk -f
NAME          FSTYPE FSVER LABEL UUID                                 FSAVAIL FSUSE% MOUNTPOINTS
nvme0n1
├─nvme0n1p1   xfs          /     cf693c2d-6ad0-4fa6-8f5e-6b46d31cd698   21.7G    27% /
└─nvme0n1p128 vfat   FAT16       F332-4170                               8.6M    14% /boot/efi
nvme1n1       xfs                f5a39fa0-a2ec-465d-ba54-2955649f02d6   15.7G     1% /var/lib/docker/volumes
[ec2-user@ip-10-76-0-15 ~]$ █
```

Figure 21: The newly created EBS data volume is mounted to the Docker volumes directory

4.4.4. Entry in the File Systems Table

The newly created `EBS` data volume is added to the file systems table in the `/etc/fstab` file. The old volume is removed from this table.

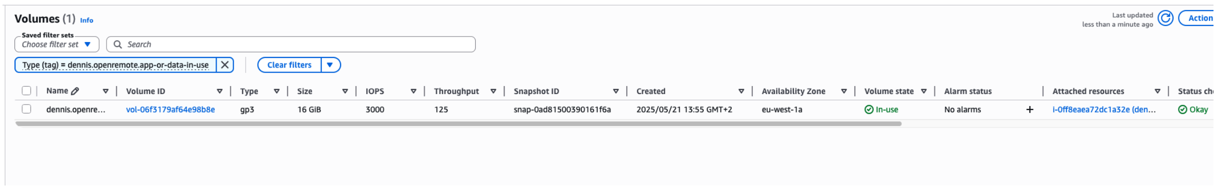


```
  GNU nano 8.3
#
UUID=cf693c2d-6ad0-4fa6-8f5e-6b46d31cd698     /            xfs    defaults,noatime  1   1
UUID=F332-4170        /boot/efi       vfat    defaults,noatime,uid=0,gid=0,umask=0077,shortname=winnt,x-systemd.automount 0 2
/swapfile swap swap defaults 0 0
UUID=f5a39fa0-a2ec-465d-ba54-2955649f02d6 /var/lib/docker/volumes xfs defaults,nofail 0 2
```

Figure 22: The newly created EBS data volume is added to the File Systems Table

4.4.5. New volume is targeted by DLM Policy

Only the newly created `EBS` data volume is targeted by the `DLM` policy using the tag `or-data-in-use` . The tag from the old volume is updated to `or-data-not-in-use` to ensure it's no longer targeted by the `DLM` policy.



Volumes (1) Info

Last updated less than a minute ago

Saved filter sets: Choose filter set

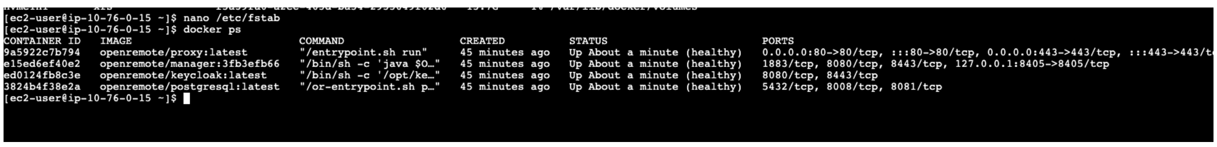
Type (tag) = dennis.openremote.app-or-data-in-use

Name	Volume ID	Type	Size	IOPS	Throughput	Snapshot ID	Created	Availability Zone	Volume state	Alarm status	Attached resources	Status ch
dennis.openre...	vol-06f3179af64e98b8e	gp3	16 GiB	3000	125	snap-0ad81500390161f6a	2025/05/21 13:55 GMT+2	eu-west-1a	In-use	No alarms	i-0ff8eaea72dc1a32e (den...	Okay

Figure 23: The EBS Data volume is targeted by the DLM Policy

4.4.6. Docker is starting

The scripts starts the `Docker` service and socket. The containers are booting up again using the existing `docker` volumes from the snapshot that are mounted to the `/var/lib/docker/volumes` directory.



```
[ec2-user@ip-10-76-0-15 ~]$ nano /etc/fstab
[ec2-user@ip-10-76-0-15 ~]$ docker ps
CONTAINER ID   IMAGE                         COMMAND                  CREATED          STATUS                        PORTS
9a5922c7b794   openremote/proxy:latest       "/entrypoint.sh run"     45 minutes ago   Up About a minute (healthy)   0.0.0.0:80->80/tcp, :::80->80/tcp, 0.0.0.0:443->443/tcp, :::443->443/t
e15ed6ef40e2   openremote/manager:3fb3efb66  "/bin/sh -c 'java $O…"   45 minutes ago   Up About a minute (healthy)   1883/tcp, 8080/tcp, 8443/tcp, 127.0.0.1:8405->8405/tcp
ed0124fb8c3e   openremote/keycloak:latest    "/bin/sh -c '/opt/ke…"   45 minutes ago   Up About a minute (healthy)   8080/tcp, 8443/tcp
3824b4f38e2a   openremote/postgresql:latest  "/or-entrypoint.sh p…"   45 minutes ago   Up About a minute (healthy)   5432/tcp, 8008/tcp, 8081/tcp
[ec2-user@ip-10-76-0-15 ~]$ █
```

Figure 24: The Docker containers are healthy

After a few minutes, the containers are healthy and OpenRemote is accessible again. The data from the snapshot is successfully loaded.

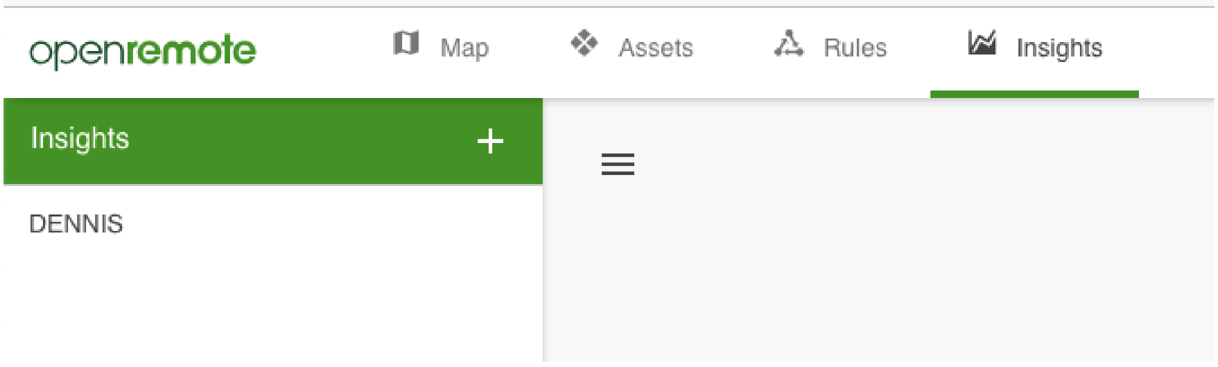
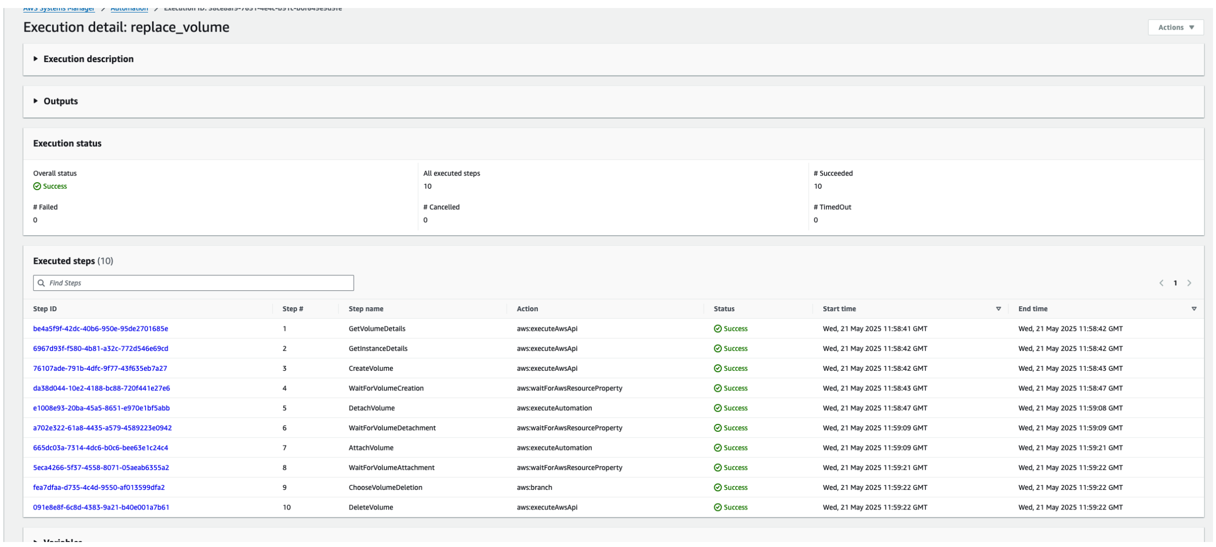


Figure 25: The IoT data is successfully loaded and available in the platform

4.4.7. Delete original EBS data volume

When the `DeleteVolume` parameter is configured to `true` this step will be executed and the original `EBS` data volume will be deleted


Execution detail: replace_volume

Execution status

Overall status: Success
All executed steps: 10
Succeeded: 10
Failed: 0
Cancelled: 0
TimedOut: 0

Executed steps (10)

Step ID	Step #	Step name	Action	Status	Start time	End time
be4a5f9f-42dc-40b6-950e-95de2701685e	1	GetVolumeDetails	aws:executeAwsApi	Success	Wed, 21 May 2025 11:58:41 GMT	Wed, 21 May 2025 11:58:42 GMT
6967d93f-f580-4b81-a32c-772d546e69cd	2	GetInstanceDetails	aws:executeAwsApi	Success	Wed, 21 May 2025 11:58:42 GMT	Wed, 21 May 2025 11:58:42 GMT
76107ade-791b-4dfc-9f77-43f635eb7a27	3	CreateVolume	aws:executeAwsApi	Success	Wed, 21 May 2025 11:58:42 GMT	Wed, 21 May 2025 11:58:43 GMT
da38d044-10e2-4188-bc88-720f441e27e6	4	WaitForVolumeCreation	aws:waitForAwsResourceProperty	Success	Wed, 21 May 2025 11:58:43 GMT	Wed, 21 May 2025 11:58:47 GMT
e1008e93-20ba-45a5-8651-e970e1bf5abb	5	DetachVolume	aws:executeAutomation	Success	Wed, 21 May 2025 11:58:47 GMT	Wed, 21 May 2025 11:59:08 GMT
a702e322-61a8-4435-a579-4589223e0942	6	WaitForVolumeDetachment	aws:waitForAwsResourceProperty	Success	Wed, 21 May 2025 11:59:09 GMT	Wed, 21 May 2025 11:59:09 GMT
665dc03a-7314-4dc6-b0c6-bee63e1c24c4	7	AttachVolume	aws:executeAutomation	Success	Wed, 21 May 2025 11:59:09 GMT	Wed, 21 May 2025 11:59:21 GMT
5eca4266-5f37-4558-8071-05aeab6355a2	8	WaitForVolumeAttachment	aws:waitForAwsResourceProperty	Success	Wed, 21 May 2025 11:59:21 GMT	Wed, 21 May 2025 11:59:22 GMT
fea7dfaa-d735-4c4d-9550-af013599dfa2	9	ChooseVolumeDeletion	aws:branch	Success	Wed, 21 May 2025 11:59:22 GMT	Wed, 21 May 2025 11:59:22 GMT
091e8e8f-6c8d-4383-9a21-b40e001a7b61	10	DeleteVolume	aws:executeAwsApi	Success	Wed, 21 May 2025 11:59:22 GMT	Wed, 21 May 2025 11:59:22 GMT

Figure 26: The Replace Volume SSM automation is successfully executed with the option to delete the original EBS data volume

5. Final changes to the implementation

After completing the tests and confirming everything is working as expected, I received one final comment from a team member to review. In the `provision_host` script, I added logic to create the default `DLM` `IAM` role if it doesn't already exist in the AWS account. The `ARN` of this role is then passed to the parameters section of the `create-ec2` `CloudFormation` stack, ensuring the `DLM` policy has the necessary permissions to create snapshots on behalf of the user.

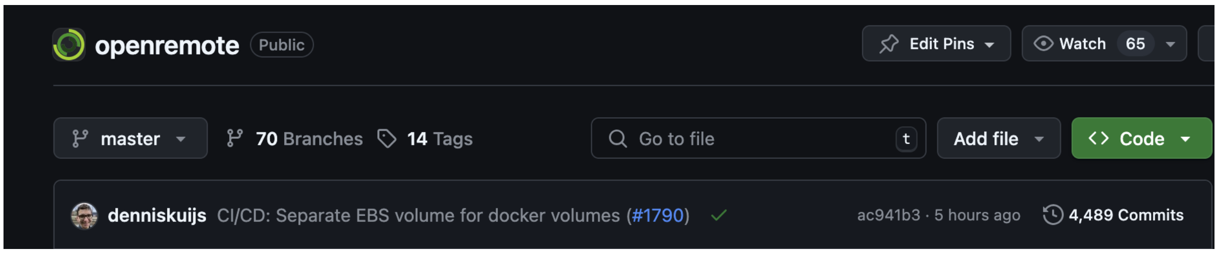
```bash
# Check for DLM IAM Role
echo "Check for DLM IAM Role"

ROLE_ARN=$(aws iam get-role --role-name AWSDataLifecycleManagerDefaultRole --query
↪  "Role.Arn" --output text $ACCOUNT_PROFILE 2>/dev/null)
if [ -z "$ROLE_ARN" ]; then
  ROLE=$(aws dlm create-default-role --resource-type snapshot --output text
  ↪  $ACCOUNT_PROFILE)

  if [ $? -ne 0 ]; then
    echo "IAM Role creation has failed"
    exit 1
  else
    echo "IAM Role creation is complete"
  fi

  ROLE_ARN=$(aws iam get-role --role-name AWSDataLifecycleManagerDefaultRole --query
  ↪  "Role.Arn" --output text $ACCOUNT_PROFILE 2>/dev/null)
fi

echo "DLM IAM Role found"
```

Instead of creating this role, I added the policy inside the role (`AWSDataLifecycleManagerServiceRole`) to the already existing `developers-access-eu-west-1` role. With this approach the check for the `DLM` `IAM` role can be removed from the `provision_host` script and the `developers-access-eu-west-1` `ARN` can be passed to the parameters section.

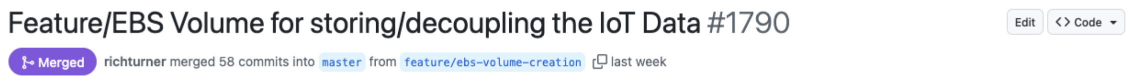
To make this work, I added the `DLM` service to the trusted entities in the role's trust policy, allowing it to assume the role. I also included the role's `ARN` in the inline `PassRole` policy to allow `CloudFormation` to assign this role to the `DLM` policy when provisioning the host.

Final moment: Work is merged into the main codebase

On Thursday May 22, 2025 at 5:24 PM the `EBS` data volume implementation is merged into the `master` branch.



denniskuijs CI/CD: Separate EBS volume for docker volumes (#1790) ✓ ac941b3 · 5 hours ago 4,489 Commits



Feature/EBS Volume for storing/decoupling the IoT Data #1790

Merged richturner merged 58 commits into `master` from `feature/ebs-volume-creation` last week