Encrypted volumes

When using Barbican for block storage encryption, you ensure that data in persistent storage volumes is stored in an encrypted fashion.

That encryption is transparent to virtual machines (instances) that you attach the volume to.

Creating an encrypted volume

For the creation of an encrypted volume, you need to provide a specific *volume type*. You can retrieve the list of available volume types with the following command:

```
openstack volume type list
```

```
+--------------------------------------+----------------------+-----------+
| ID                                   | Name                 | Is Public |
+--------------------------------------+----------------------+-----------+
| a479a6b0-b283-41a5-b38b-5b08e7f902ca | volumes_hdd_encrypted | True      |
| d9dfa98a-238d-4ca0-9abf-701fceb05623 | __DEFAULT__          | True      |
| 86796611-fb12-4628-b6b1-e09469e301d7 | volumes_hdd          | True      |
+--------------------------------------+----------------------+-----------+
```

> In Cleura, all volume types that support encryption use the suffix `_encrypted`.

To create a volume with encryption, you need to explicitly specify the `--type` option to the `openstack volume create` command. The following example creates a volume using the `volumes_hdd_encrypted` type, naming it `enc_drive` and setting its size to 10 GiB:

```
openstack volume create \
  --type volumes_hdd_encrypted \
  --size 10 \
  enc_drive
```

```
+---------------------+--------------------------------------+
| Field               | Value                                |
+---------------------+--------------------------------------+
| attachments         | []                                   |
| availability_zone   | nova                                 |
| bootable            | false                                |
| consistencygroup_id | None                                 |
| created_at          | 2021-04-27T13:52:10.000000           |
| description         | None                                 |
```

```
| encrypted          | True                                 |
| id                 | 33211b21-8d4f-48e9-b76f-ec73ffd19def |
| multiattach        | False                                |
| name               | enc_drive                            |
| properties         |                                      |
| replication_status | None                                 |
| size               | 10                                   |
| snapshot_id        | None                                 |
| source_volid       | None                                 |
| status             | creating                             |
| type               | volumes_hdd_encrypted                |
| updated_at         | None                                 |
| user_id            | 966ad341f4e14920b5f589f900246ccc     |
+--------------------+--------------------------------------+
```

Upon volume creation, this will create a one-off encryption key, which is stored in Barbican and applies to this one volume only. In other words, the key created for this volume will be unable to decrypt any other volumes except the one it was created for.

Retrieving a volume's encryption key

Once you have created an encrypted volume, you may retrieve a reference to the Barbican secret that represents its encryption key. You do this with the following command:

```
openstack volume show \
  --os-volume-api-version 3.66 \
  -f value \
  -c encryption_key_id \
  enc_drive
```

Instead of the volume name, you can of course also specify its UUID:

```
openstack volume show \
  --os-volume-api-version 3.66 \
  -f value \
  -c encryption_key_id \
  33211b21-8d4f-48e9-b76f-ec73ffd19def
```

Deleting an encrypted volume

When you decide you no longer need an encrypted volume and want to delete it, you can do so with the `openstack volume delete` command. As long as you do this with the same user account as the one that created the volume, this will succeed without further intervention.

However, if you are trying to delete a volume that was created by a different user, you'll run into the limitation that the *secret* associated with the volume is owned by that user. As a result, the deletion of the encrypted volume using your own user credentials will fail.

There are two options to work around this limitation:

1. You can switch to the user credentials of the user that created the volume (if you have access to them), and proceed with the deletion.
2. You can ask the user that created the volume to add you to the Access Control List (ACL) for the secret. This will enable you to read the secret, and to delete the volume using your own credentials.

Block device encryption caveats

Once a volume is configured to use encryption and is also attached to an instance in Cleura, some caveats apply that you might want to keep in mind.

Sometimes, automatically or through administrator intervention, we move one of your instances to another physical machine. This process is known as *live migration,* and it normally does not interrupt the instance's functionality at all — typically, neither you nor the application users notice that live migration has even happened. This is a very common occurrence when we do routine upgrades of the Cleura platform, during our pre-announced maintenance windows.

The same considerations apply to physical node failure. If the physical machine running your instance fails, we can automatically recover it onto another machine — an action known as *evacuation.*

Live migration or evacuation *including encrypted volumes* does, however, require that whoever does the migration also has at least read access to the volume's encryption secret.

This means that you have two options:

1. If you *do* trust us to include your instances in live migrations and evacuations, even if they attach encrypted volumes, then you can add our administrative account to the Access Control List (ACL) for your secrets.
2. If you *don't* want to share your secrets but you still want to use encrypted volumes, you should build your own mechanism or process (preferably automated) so that your instances recover in case they become non-functional.

Last update: 2022-11-07
Created: 2022-03-08
Authors: Florian Haas