



DRBD Encryption for ESC Active/Standby and Active/Active HA Data Replication

This chapter contains the following sections:

- [DRBD Encryption for ESC HA Data Replication](#) , on page 1
- [ESC HA with DRBD Encryption](#), on page 1

DRBD Encryption for ESC HA Data Replication

ESC uses DRBD for data replication across different nodes in an HA cluster environment. DRBD layers logical block device over existing local block devices on cluster nodes.

The written data to the active node is transferred to the lower-level block device and simultaneously propagated to the secondary node(s). Currently ESC mounts DRBD device directly on `/opt/cisco/esc/esc_database`.

For Example:

```
# df
Filesystem            1K-blocks    Used Available Use% Mounted on
devtmpfs              2961760      0   2961760  0% /dev
tmpfs                 2972164      4   2972160  1% /dev/shm
tmpfs                 2972164    8748  2963416  1% /run
...
tmpfs                 594436      0   594436  0% /run/user/1004
/dev/mapper/esc_crypt 3028620    57212  2797848  3% /opt/cisco/esc/esc_database
```

Block device encryption encrypts or decrypts the data transparently as it is written/read from block devices, the underlying block device sees only encrypted data.

Security is enhanced with the `dm-crypt/LUKS` layer to encrypt the data in DRBD partition, between filesystem and DRBD device. LUKS (Linux Unified Key Setup) is a specification for block device encryption.

ESC HA with DRBD Encryption

The following bootvm commands boot ESC HA with DRBD encrypted:

Select DRBD LUKS encryption through `bootvm.py`. There are 4 variations that result in the equivalent ESC `day-0 user-data/esc-config.yaml` when passed to the ESC VM instance.

```
bootvm.py --fs_encryption_type luks --fs_luks_key_prompt
bootvm.py --fs_encryption_type luks --fs_luks_key 'LuksKeyValue'
=> injects the luks key into default file location /opt/cisco/esc/esc-config/luks_key
```

```
bootvm.py --fs_encryption_type luks --file
root:0400:/opt/cisco/esc/esc-config/luks_key:path-to-local-luks-key-file
=> injects a local file containing the luks key
```

The following command shows an advanced usage to manage the luks key file at a different path on the ESC VM filesystem:

```
bootvm.py --fs_encryption_type luks --fs_luks_key_file path-on-esc-vm-luks-key-file
--fs_luks_key_prompt
bootvm.py --fs_encryption_type luks --fs_luks_key_file path-on-esc-vm-luks-key-file
--fs_luks_key 'LuksKeyValue'
=> injects the luks key into a different file location
```

```
bootvm.py --fs_encryption_type luks --fs_luks_key_file path-on-esc-vm-luks-key-file --file
root:0400:path-on-esc-vm-luks-key-file:path-to-local-luks-key-file
=> injects the luks key as read from a local file into a different file location
```

Use the following commands if you are installing ESC with a custom user-data, such as ESC Active/Active deployment with heat templates:

The luks key is specified as a day-0 file and an attribute under `esc-config.yaml / filesystem`.

Encode the luks key as base64:

```
base64 <<<'LuksKeyValue'
THVrc0tleVZhbHVlCg==
```

Then, insert the previous luks key into the user-data / cloud-config file:

```
write_files:
- path: /opt/cisco/esc/esc-config/luks_key
  owner: root:root
  permissions: '400'
  encoding: b64
  content: THVrc0tleVZhbHVlCg==

- path: /opt/cisco/esc/esc-config/esc-config.yaml
  owner: root:esc-user
  permissions: '0640'
  content: |
    resources:
      filesystem:
        depend_on: drbd:active
        encryption_type: luks
        luks_key_file: /opt/cisco/esc/esc-config/luks_key
```