



# FortiSIEM - Disk Encryption of Data on FortiSIEM Supervisor

Version 5.3.3

**FORTINET DOCUMENT LIBRARY**

<https://docs.fortinet.com>

**FORTINET VIDEO GUIDE**

<https://video.fortinet.com>

**FORTINET BLOG**

<https://blog.fortinet.com>

**CUSTOMER SERVICE & SUPPORT**

<https://support.fortinet.com>

**FORTINET TRAINING & CERTIFICATION PROGRAM**

<https://www.fortinet.com/support-and-training/training.html>

**NSE INSTITUTE**

<https://training.fortinet.com>

**FORTIGUARD CENTER**

<https://fortiguard.com/>

**END USER LICENSE AGREEMENT**

<https://www.fortinet.com/doc/legal/EULA.pdf>

**FEEDBACK**

Email: [techdoc@fortinet.com](mailto:techdoc@fortinet.com)



---

# TABLE OF CONTENTS

Change Log .....	4
<b>Disk Encryption of Data on FortiSIEM Supervisor .....</b>	<b>5</b>
Detailed Steps to Encrypt Disk Data .....	5
Step 1: Import VA VM image and start the VM .....	5
Step 2: Unmount /cmdb filesystem .....	6
Step 3: Wipe the /cmdb disk of previous filesystem and partition info .....	6
Step 4: Setup LUKS format disk encryption for /cmdb disk .....	6
Step 5: Add a new key to LUKS disk for /cmdb .....	7
Step 6: Open the encrypted CMDB disk .....	7
Step 7: Configure automatic encrypted disk “open” at boot up .....	7
Step 8: Create an ext4 filesystem on the “opened” encrypted disk .....	8
Step 9: Replace /etc/fstab entry to mount the encrypted disk instead of the original .....	8
Step 10: Remount new /cmdb .....	8
Steps to Encrypt Disk Data for /svn .....	9
Steps to Encrypt Disk Data for /data (local) .....	9

---

## Change Log

Date	Change Description
05/13/2019	Initial version of Disk Encryption of Data on FortiSIEM Supervisor
11/20/2019	Release of Disk Encryption of Data on FortiSIEM Supervisor for 5.2.6.
03/30/2020	Release of Disk Encryption of Data on FortiSIEM Supervisor for 5.3.0.

# Disk Encryption of Data on FortiSIEM Supervisor

The steps below show how to encrypt `/cmdb`, `/svn`, and `/data` disks on a FortiSIEM supervisor VM node with local disk for EventDB (`/data`). If you are using NFS or Elastic storage, you must perform additional steps for the actual data directories on these servers in addition to supervisor (`/cmdb`, `/svn`).

**Note 1:** We do not recommend encrypting the root disk which presents an operational challenge during boot up to provide a passphrase. The root disk contains binaries and some internal system and application logs, not data.

**Note 2:** Disk encryption key management is an operational challenge. If you want strong security, then you must protect encryption keys with a passphrase and that requires a human to type them and mount the “opened” disks. The less secure alternative is to use keys that are not protected by a passphrase and stored in a file on the root partition.

It is best to perform these steps on a fresh installation prior to initializing the product. This avoids the need for additional disks. If you have to perform these on an existing installation of FortiSIEM, then you must have additional disks of the same capacity to encrypt and copy the data to it. Here is a quick overview of the steps:

1. [Import the latest FortiSIEM VA image on one of the supported platforms.](#)
2. For each disk, (`/cmdb`, `/svn`, `/data`) complete the following steps:
  - a. [Unmount the filesystem.](#)
  - b. [Wipe the filesystem of exiting filesystem and partition table.](#)
  - c. [Setup a LUKS formatted disk which prompts you for a passphrase that protect the encryption keys in Slot 0 – Your disk is now encrypted.](#)
  - d. [Add another LUKS key into Slot 1 and save this into a file on `/etc`. \(There are a total of 8 key slots\).](#)
  - e. [Open the encrypted disk after providing a passphrase for further operations which provides a `/dev/mapper/XYZ` device.](#)
  - f. [Create an entry in `/etc/crypttab` which will open the encrypted disk using the Slot 1 key file you saved above at boot time.](#)
  - g. [Create an `ext4` filesystem on the opened disk in `/dev/mapper/XYZ`.](#)
  - h. [Create an `/etc/fstab` entry to mount the opened disk above to a named path \(`/cmdb`, `/svn`, or `/data` as the case may be\).](#)

## Detailed Steps to Encrypt Disk Data

### Step 1: Import VA VM image and start the VM

Import VA VM image on one of the supported platforms as per user guide. Add a 4th `/data` disk if you plan on using local storage. Then, start the VM. Once you login by using `ssh`, run the `df -h` command to list the mounted filesystems. You will see both `/cmdb` and `/svn`, for example:

```
[root@localhost ~]# df -h
Filesystem      Size  Used Avail Use% Mounted on
/dev/sda3        55G   6.7G   46G  13% /
tmpfs            7.8G     0   7.8G   0% /dev/shm
```

```
/dev/sda1      124M   43M   76M   36% /boot
/dev/sdb1      60G   180M   56G    1% /cmdb
/dev/sdc1      60G   180M   56G    1% /svn
[root@localhost ~]#
```

If you had added a 4th disk for /data, you should be able to list this as well by using the `fdisk` command. These device names will vary on other platforms such as AWS, KVM, etc.

```
[root@localhost ~]# fdisk -l /dev/sdd
Disk /dev/sdd: 53.7 GB, 53687091200 bytes
255 heads, 63 sectors/track, 6527 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0x00000000
```

### Step 2: Unmount /cmdb filesystem

Use the `umount` command to unmount the CMDB filesystem, for example:

```
umount /cmdb
```

### Step 3: Wipe the /cmdb disk of previous filesystem and partition info

Use the `wipefs` command to clear the existing filesystem and partition information from the CMDB disk, for example:

```
[root@localhost ~]# wipefs --all /dev/sdb1
2 bytes [53 ef] erased at offset 0x438 (ext3)
[root@localhost ~]# wipefs --all /dev/sdb
wipefs: WARNING: /dev/sdb: appears to contain 'dos' partition table
```

### Step 4: Setup LUKS format disk encryption for /cmdb disk

When you run the `cryptsetup luksFormat` command, you must provide a passphrase that protects the encryption key for the disk. This key is stored in Slot 0 of the keys.

```
[root@localhost ~]# cryptsetup luksFormat /dev/sdb
WARNING!
=====

This will overwrite data on /dev/sdb irrevocably.
Are you sure? (Type uppercase yes): YES
Enter LUKS passphrase:
Verify passphrase:
```

There are a total of 8 slots for additional keys which can be used to provide multiple admins the ability to unlock the disks or can be used for periodic rotation of keys. The following command can be used to dump information about different slots.

```
[root@localhost ~]# cryptsetup luksDump /dev/sdb|grep Slot
Key Slot 0: ENABLED
Key Slot 1: DISABLED
Key Slot 2: DISABLED
Key Slot 3: DISABLED
Key Slot 4: DISABLED
Key Slot 5: DISABLED
Key Slot 6: DISABLED
Key Slot 7: DISABLED
```

### Step 5: Add a new key to LUKS disk for /cmdb

Use the `cryptsetup` command with the `luksAddKey` option to add the LUKS key for the CMDB disk, for example:

```
[root@localhost ~]# cryptsetup luksAddKey /dev/sdb /etc/enccmdbkey
```

Enter any passphrase.

If you enter the `cryptsetup` command with the `luksAddKey` option again, you will see that two slots have enabled keys. Slot 0 contains the key that was generated when you first created the encrypted disk. Slot 1 contains the new key you just added above which was copied to a file. Additional commands for LUKS keys are in this document.

```
[root@localhost ~]# cryptsetup luksDump /dev/sdb|grep Slot
Key Slot 0: ENABLED
Key Slot 1: ENABLED
Key Slot 2: DISABLED
Key Slot 3: DISABLED
Key Slot 4: DISABLED
Key Slot 5: DISABLED
Key Slot 6: DISABLED
Key Slot 7: DISABLED
```

### Step 6: Open the encrypted CMDB disk

Use the `cryptsetup` command with the `luksOpen` option to open the encrypted CMDB disk, for example:

```
[root@localhost ~]# cryptsetup luksOpen /dev/sdb encryptedCmdb --key-file
/etc/enccmdbkey
[root@localhost ~]#
```

### Step 7: Configure automatic encrypted disk “open” at boot up

Create an entry in `/etc/crypttab` which will open the encrypted disk using the Slot 1 key file you saved above at boot time.

```
echo "encryptedCmdb /dev/sdb /etc/enccmdbkey luks" > /etc/crypttab
```

## Step 8: Create an ext4 filesystem on the “opened” encrypted disk

Use the `mkfs.ext4` command to create an ext4 file system on the disk, for example:

```
[root@localhost ~]# mkfs.ext4 /dev/mapper/encryptedCmdb
mke2fs 1.41.12 (17-May-2010)
Filesystem label=
OS type: Linux
Block size=4096 (log=2)
Fragment size=4096 (log=2)
Stride=0 blocks, Stripe width=0 blocks
3932160 inodes, 15728128 blocks
786406 blocks (5.00%) reserved for the super user
First data block=0
Maximum filesystem blocks=4294967296
480 block groups
32768 blocks per group, 32768 fragments per group
8192 inodes per group
Superblock backups stored on blocks:
32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632, 2654208,
4096000, 7962624, 11239424
Writing inode tables: done
Creating journal (32768 blocks): done
Writing superblocks and filesystem accounting information: done
```

This filesystem will be automatically checked every 37 mounts or 180 days, whichever comes first. You can override this by using the `tune2fs -c` or `-i` command.

## Step 9: Replace `/etc/fstab` entry to mount the encrypted disk instead of the original

Enter these commands to mount the encrypted disk instead of the non-encrypted disk.

```
[root@localhost ~]# grep cmdb /etc/fstab
/dev/mapper/encryptedCmdb /cmdb ext4 defaults,nodev 0 1
```

## Step 10: Remount new `/cmdb`

Use the `mount` and `df` commands to remount the new CMDB disk, for example:

```
[root@localhost ~]# mount /cmdb
[root@localhost ~]# df -h /cmdb
Filesystem      Size  Used Avail Use% Mounted on
/dev/mapper/encryptedCmdb
59G   52M   56G   1% /cmdb
```



## Steps to Encrypt Disk Data for /svn

Encrypting disk data for `/svn` is very similar. Replace the device names, encryption key file name, and device mapper device names appropriately.

## Steps to Encrypt Disk Data for /data (local)

Encrypting disk data for local `/data` is very similar. Once you have created the encrypted disk, do not create the filesystem. That will be done when you setup storage. You must provide a `/dev/mapper/encryptedData` device name in the storage setup.



**FORTINET®**



Copyright© (Undefined variable: FortinetVariables.Copyright Year) Fortinet, Inc. All rights reserved. Fortinet®, FortiGate®, FortiCare® and FortiGuard®, and certain other marks are registered trademarks of Fortinet, Inc., in the U.S. and other jurisdictions, and other Fortinet names herein may also be registered and/or common law trademarks of Fortinet. All other product or company names may be trademarks of their respective owners. Performance and other metrics contained herein were attained in internal lab tests under ideal conditions, and actual performance and other results may vary. Network variables, different network environments and other conditions may affect performance results. Nothing herein represents any binding commitment by Fortinet, and Fortinet disclaims all warranties, whether express or implied, except to the extent Fortinet enters a binding written contract, signed by Fortinet's General Counsel, with a purchaser that expressly warrants that the identified product will perform according to certain expressly-identified performance metrics and, in such event, only the specific performance metrics expressly identified in such binding written contract shall be binding on Fortinet. For absolute clarity, any such warranty will be limited to performance in the same ideal conditions as in Fortinet's internal lab tests. In no event does Fortinet make any commitment related to future deliverables, features or development, and circumstances may change such that any forward-looking statements herein are not accurate. Fortinet disclaims in full any covenants, representations, and guarantees pursuant hereto, whether express or implied. Fortinet reserves the right to change, modify, transfer, or otherwise revise this publication without notice, and the most current version of the publication shall be applicable.