LUKS2 Tutorial (External HDD/SSD/USB) on Linux

Goal

Encrypt an external HDD/SSD/USB with **LUKS2**, give it a friendly name/label, and be able to unlock + mount it both on **your laptop** and on a **foreign PC**.

> **WARNING:** This erases data if you format a device/partition. Back up first.

Assumptions

- You are on Linux (Ubuntu/Debian-like examples).
- Package needed: `cryptsetup`. Install once:

  ```
  sudo apt update
  sudo apt install cryptsetup
  ```

- You will encrypt a **partition** (recommended), e.g. `/dev/sdb1`, not the whole disk.

1 Steps (no commentary)

1) Identify the external drive and its partition

```
lsblk -o NAME,SIZE,TYPE,FSTYPE,LABEL,UUID,MOUNTPOINT
```

2) Unmount anything currently mounted from it

```
sudo umount /dev/sdX1 2>/dev/null || true
```

3) (Optional) Wipe the partition before LUKS2 (zero-fill or random-fill)

```
# ZERO-FILL (fast-ish, but still writes the whole partition)
sudo dd if=/dev/zero of=/dev/sdX1 bs=16M status=progress conv=fsync

# RANDOM-FILL (much slower; can take a very long time)
sudo dd if=/dev/urandom of=/dev/sdX1 bs=16M status=progress conv=fsync
```

4) Create the LUKS2 container (force LUKS2)

```
sudo cryptsetup luksFormat --type luks2 /dev/sdX1
```

5) Open (unlock) it with the name you want

```
sudo cryptsetup open /dev/sdX1 MyVault
```

6) Create a filesystem inside the unlocked mapping

(Example: ext4)

```
sudo mkfs.ext4 -L MyVaultFS /dev/mapper/MyVault
```

7) Close it

```
sudo cryptsetup close MyVault
```

8) Use it on your laptop (unlock + mount)

```
sudo cryptsetup open /dev/sdX1 MyVault
sudo mkdir -p /mnt/MyVault
sudo mount /dev/mapper/MyVault /mnt/MyVault
```

9) Safely unmount + lock

```
sudo umount /mnt/MyVault
sudo cryptsetup close MyVault
```

10) Verify it is LUKS2

```
sudo cryptsetup luksDump /dev/sdX1 | head
```

11) Use it on a foreign PC (unlock + mount)

Same as steps 8–9, but you may need to install `cryptsetup` first.

12) Make the desktop prompt for password when plugged in (typical GUI behaviour)

Enable auto-mount for removable media in your desktop settings, then plug the drive in and click it in the file manager; it should prompt for the LUKS passphrase.

---

2 Steps again (with comments)

1) Identify the correct device/partition

```
lsblk -o NAME,SIZE,TYPE,FSTYPE,LABEL,UUID,MOUNTPOINT
```

Comment: Find your external disk by size and by the fact it appears when you plug it in. You typically want a **partition** like `/dev/sdb1` (replace `sdX1` below with yours). If you see `/dev/sdb` and `/dev/sdb1`, encrypt `/dev/sdb1`.

2) Unmount any mounted partition from that drive

```
sudo umount /dev/sdX1 2>/dev/null || true
```

Comment: LUKS formatting will fail or be unsafe if the partition is mounted.

3) (Optional) Wipe the partition to hide old data patterns

```
# ZERO-FILL: overwrites with 0x00. Often enough for "normal" reuse.
sudo dd if=/dev/zero of=/dev/sdX1 bs=16M status=progress conv=fsync

# RANDOM-FILL: overwrites with random bytes. Much slower.
sudo dd if=/dev/urandom of=/dev/sdX1 bs=16M status=progress conv=fsync
```

Comment:

- `/dev/sdX1` must be the **partition**, not your OS disk.
- Random-fill can take *hours* or more, especially on large HDDs.
- **NOTE:** For SSDs, wear-levelling means a full overwrite is not a perfect "secure erase". For SSDs, consider the drive's built-in secure erase / sanitize features instead.

4) Encrypt the partition with *forced* LUKS2

```
sudo cryptsetup luksFormat --type luks2 /dev/sdX1
```

Comment:

- `-type luks2` forces the header format to be **LUKS2**.
- You will be asked to confirm and then set a passphrase.
- You can add additional passphrases later with:

  ```
  sudo cryptsetup luksAddKey /dev/sdX1
  ```

5) Open it with your chosen mapping name ("name when unlocked")

```
sudo cryptsetup open /dev/sdX1 MyVault
```

Comment: The name `MyVault` becomes `/dev/mapper/MyVault`. This name is chosen at open-time; it is not permanently stored on the drive.

6) Create a filesystem inside the unlocked mapping and set a friendly filesystem label

```
sudo mkfs.ext4 -L MyVaultFS /dev/mapper/MyVault
```

Comment:

- `MyVaultFS` is the filesystem label many file managers display.
- ext4 is a good default for Linux-to-Linux use.

7) Close it after setup

```
sudo cryptsetup close MyVault
```

Comment: This removes the unlocked device mapping until you unlock again.

8) Unlock + mount on your laptop (manual method that works everywhere)

```
sudo cryptsetup open /dev/sdX1 MyVault
sudo mkdir -p /mnt/MyVault
sudo mount /dev/mapper/MyVault /mnt/MyVault
```

Comment: Unlocking creates `/dev/mapper/MyVault`. Mounting attaches it to `/mnt/MyVault`.

9) Unmount + lock (always do this before unplugging)

```
sudo umount /mnt/MyVault
sudo cryptsetup close MyVault
```

Comment: Unmount first, then close, then unplug.

10) Foreign PC (not yours): what changes?

Minimal checklist:

- Install `cryptsetup` if missing (Debian/Ubuntu): `sudo apt install cryptsetup`.
- Identify the partition again (`lsblk`).
- Unlock + mount:

  ```
  sudo cryptsetup open /dev/sdX1 MyVault
  sudo mkdir -p /mnt/MyVault
  sudo mount /dev/mapper/MyVault /mnt/MyVault
  ```

- Unmount + close when done.

11) Verify the on-disk format is LUKS2

```
sudo cryptsetup luksDump /dev/sdX1 | head
```

Comment: Look for `Version:  2`. That confirms LUKS2.

12) "Recognised instantly" and prompts for password on plug-in

Reality check: LUKS volumes cannot be unlocked without a secret (passphrase/key). But on most Linux desktops, when you click the drive in the file manager (or auto-mount triggers), you will be prompted for the LUKS passphrase.

What to do on your laptop:

- Enable "auto-mount removable media" in your desktop settings.
- Plug the drive in and click it in the file manager; you should get a passphrase prompt.

If you want truly automatic unlocking on your laptop: That requires a key file stored on your laptop (or TPM/smartcard). This is a different security model, so it is not enabled here by default.

Quick template (copy/paste)

Replace `/dev/sdX1` and names.

```
# optional wipe (DESTROYS DATA ON /dev/sdX1)
sudo dd if=/dev/zero   of=/dev/sdX1 bs=16M status=progress conv=fsync
# OR (very slow)
sudo dd if=/dev/urandom of=/dev/sdX1 bs=16M status=progress conv=fsync

# one-time setup (DESTROYS DATA ON /dev/sdX1)
sudo cryptsetup luksFormat --type luks2 /dev/sdX1
sudo cryptsetup open /dev/sdX1 MyVault
sudo mkfs.ext4 -L MyVaultFS /dev/mapper/MyVault
sudo cryptsetup close MyVault

# daily use
sudo cryptsetup open /dev/sdX1 MyVault
sudo mkdir -p /mnt/MyVault
sudo mount /dev/mapper/MyVault /mnt/MyVault

# done
sudo umount /mnt/MyVault
sudo cryptsetup close MyVault
```

2.1 Basically

2.1.1 Names when opening

- Use `lsblk` to identify the correct partition (`/dev/sdX1`).
- `Name1` is the *temporary* unlocked mapping name (`/dev/mapper/Name1`); you choose it at open-time (it is not stored on disk).
- Create a mount point directory (e.g. `/mnt/Name2`).
- **Specification:** the mapper name and mount point name can be different, e.g.

  ```
  # mapper name = Name1, mount point = /mnt/Name2
  sudo cryptsetup open /dev/sdX1 Name1
  sudo mkdir -p /mnt/Name2
  sudo mount /dev/mapper/Name1 /mnt/Name2
  ```

- Mounting makes the drive's filesystem appear *at* that directory; files you read/write in `/mnt/Name2` are stored on the encrypted drive.
- After `umount`, the directory remains but the drive contents are no longer accessible there.

2.1.2 Initialisation (labels vs names)

- **`MyVaultFS`** = filesystem label set by `mkfs.ext4 -L ...` (what file managers usually show after unlocking/mounting).
- **`MyVault`** = mapper name under `/dev/mapper/` chosen at unlock-time (not stored on disk).