Linux Unified Key Setup (LUKS)

> I messed up editing this page and some of the information is missing and in the wrong order.

> All the examples below assume wanting to setup a btrfs pool on two disks `/dev/sdX` and `/dev/sdY` that will be used just for additional storage.

Prepare Disks

> Before encrypting a drive, it is recommended to perform a secure erase of the disk by overwriting the entire drive with random data. To prevent cryptographic attacks or unwanted file recovery, this data is ideally indistinguishable from data later written by dm-crypt.

[Source](https://wiki.archlinux.org/title/Dm-crypt/Drive_preparation)

There are multiple ways to prepare a disk and some other potentially better ones listed on the page linked to above. Because I want to wipe my disks as quickly as possible and they were both the same size I am using a slightly more complicated method. This method creates the equivalent of filling the disks with the output from `/dev/urandom` but does so faster by using the output of encrypting `/dev/zero` and writing that to the disks instead. I save even more time by using `tee` and `process substitution` to redirect the output to both drives at once. Just for good measure I am using `pv` to measure the speed at which I am writing and to track my progress.

```
PASS=$(tr -cd '[:alnum:]' < /dev/urandom | head -c128)
openssl enc -aes-256-ctr -pass pass:"$PASS" -nosalt < /dev/zero | dd ibs=4K | pv | tee >(dd obs=64K oflag=direct of=/dev/sdX) | dd obs=64K oflag=direct of=/dev/sdY
```

Partition

Although LUKS can be layered on top of redundant storage (btrfs -or- mdadm + dm-integrity) for my usages it almost always makes sense to layer those things on top of LUKS instead. My goal is just to have an encrypted filesystem for storage of data so I only need to create one partition on each disk.

```
sgdisk --clear --new=0:0:0 /dev/sdX
sgdisk --clear --new=0:0:0 /dev/sdY
```

Encrypt

Setup LUKS with passphrase encrypted drives.

```
cryptsetup luksFormat /dev/sdX1 cryptbtrpool_1
cryptsetup luksFormat /dev/sdY1 cryptbtrpool_2
```

Create encryption key.

```
dd if=/dev/urandom bs=512 count=4 of=/etc/keyfile
```

Add keyfile as optional decryption key.

```
cryptsetup luksAddKey /dev/sdX1 /etc/keyfile
cryptsetup luksAddKey /dev/sdY1 /etc/keyfile
```

Unlock Devices

```
cryptsetup open /dev/sdX1 cryptbtrpool_1 --key-file=/etc/keyfile
cryptsetup open /dev/sdY1 cryptbtrpool_2 --key-file=/etc/keyfile
```

Create btrfs Pool

> You can use LUKS devices like any other block device and format them any way you want. However below I am combining them into a RAID1 btrfs filesystem that spans both disks and utilizes the underlying LUKS devices for encryption since btrfs doesn't support this natively.

```
mkfs.btrfs --data raid1 --metadata raid1 --label btrpool /dev/mapper/cryptbtrpool_1 /dev/mapper/cryptbtrpool_2
```

Add to `crypttab`

It is best practice to reference drives in `/etc/fstab` or `/etc/crypttab` using something more constant than just the dev name like `/dev/sdX1`. I reference the drives by the UUID of the LUKS partition but another good option is to reference the drives by their "disk-id" found under `/dev/disk/by-id/...`.

I can find the UUID for the LUKS partitions by using `blkid` and `grep` to filter the output.

```
blkid | grep LUKS
/dev/sdX1: UUID="99fc46af-1048-4c50-bc38-2085aee78579" TYPE="crypto_LUKS" PARTLABEL="Linux filesystem" PARTUUID="8cbdf3b0-7ba0-4b7b-8639-15ea3029c72e"
/dev/sdY1: UUID="507033de-5eb5-4baf-8875-6595fbb260af" TYPE="crypto_LUKS" PARTLABEL="Linux filesystem" PARTUUID="14d8331c-9a82-4e5d-8ea8-4d1a6d8025fe"
```

Now with those UUIDs I can use them in `/etc/crypttab` to automatically open my LUKS partition during boot.

```
# <target name> <source device>         <key file>      <options>
+ cryptbtrpool_1  UUID=507033de-5eb5-4baf-8875-6595fbb260af       /etc/keyfile
+ cryptbtrpool_2  UUID=99fc46af-1048-4c50-bc38-2085aee78579       /etc/keyfile
```

Add to `fstab`

Entries in `/etc/crypttab` will all have completed by the time entries in `/etc/fstab` are attempted. So knowing that the LUKS devices will have been automatically opened I can then mount that filesystem as I would any other block device. Since the btrfs filesystem above was labeled `btrpool` it is possible to mount subvolumes using a combination of that label and the names of any subvolumes that were created.

```
# /etc/fstab
# <file system> <mount point> <type> <options> <dump> <pass>
+ LABEL=btrpool   /storage/btrpool        btrfs   x-mount.mkdir=0755,defaults,subvol=@,compress=zstd      0 0
+ LABEL=btrpool   /storage/btrpool/services       btrfs   defaults,subvol=@services,compress=zstd,X-mount.mkdir=0755      0 0
+ LABEL=btrpool   /storage/btrpool/media  btrfs   defaults,subvol=@media,compress=zstd,X-mount.mkdir=0755      0 0
```

Revision #6
Created 20 October 2021 16:37:38 by dustin@swigg.net
Updated 9 January 2023 03:11:20 by dustin@swigg.net