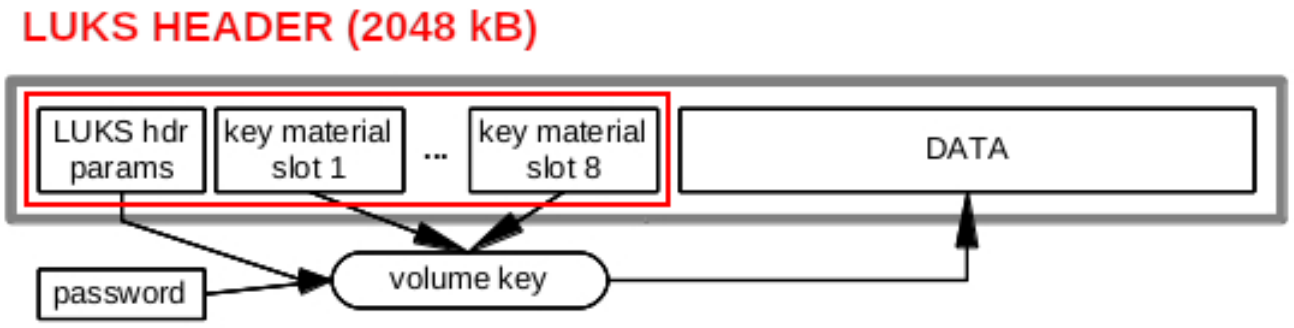
TüBix 2019: Verschlüsselung einer SSD mit LUKS ..und wie man sich sehr schnell selbst kompromittiert

- Hardware-Einschränkungen bei LUKS auf SSD?
  Keine. TRIM-Befehl (fstrim) & Co. nutzbar (sofern entsprechend konfiguriert)

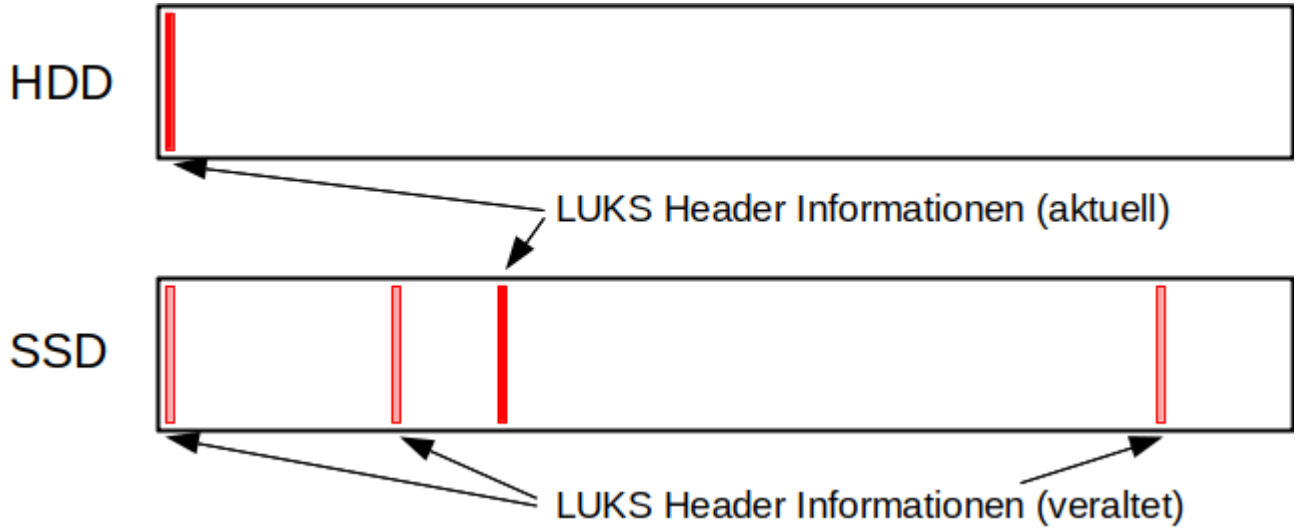
- Besonderheiten (im Vgl. zur HDD)?
  Datenträger zu Beginn nicht komplett mit Zufallszahlen überschreiben (→ schlechtere Performance)

- LUKS auf einem Blockdevice:



SSD vs. HDD

- Zus. Abstraktionsschicht durch WEAR-Leveling in SSD Firmware (Copy-on-write)
- Kein gezieltes Löschen von Sektoren möglich (kein „LUKS-Killswitch“)
- Alter LUKS Header resp. Teile davon verbleiben unverändert auf Datenträger bis betroffene Sektoren erneut überschrieben werden
- Auslesen alter LUKS Header (resp. Key Slots) bei forensischer Analyse möglich



Und nun?

Was man nicht tun sollte:

- Schwaches Passwort zu Beginn bei Systemeinrichtung vergeben, was man später „eh noch ändert“

Was man tun kann (und sollte):

- Starkes Passwort gleich bei Systemeinrichtung vergeben
- Bei Kompromittierung SSD-Datenträger aufgeben und komplett überschreiben (secure erase)
  → Einrichtung von vorne beginnen
- LUKS Header nicht auf SSD speichern (stattdessen HDD, USB-Stick, ..) - bietet zus. Vorteile:
  - Deniability
  - Zwei- resp. Mehrfaktorauthentifizierung/-absicherung (z.B. noch zus. GnuPG-Key nutzen)
- dmcrypt direkt nutzen (und damit verbundene Nachteile als Preis akzeptieren)
- Keine SSD nutzen (ernst gemeint!)

Übrigens:

- Bei einer SSHD (HDD+SSD) ist die Problematik grundsätzlich ähnlich
- Prinzipiell jede Art der Datenträgerverschlüsselung, die Informationen auf dem Datenträger speichert, ist betroffen. So z.B. auch ecryptFS.

Links:

- https://gitlab.com/cryptsetup/cryptsetup/wikis/FrequentlyAskedQuestions#5-security-aspects
- https://wiki.archlinux.org/index.php/Dm-crypt/Specialties#Discard/TRIM_support_for_solid_state_drives_(SSD)