Studienprogramm Sachbearbeiter:in Digitale Forensik
Praktikum Betriebssysteme
Dozenten: Tim Wetterau
wetterau@hs-mittweida.de
Stand: 26.06.2023

HOCHSCHULE MITTWEIDA
University of Applied Sciences

Betriebssysteme

Praktikum 3 - Festplatten und RAID-Systeme

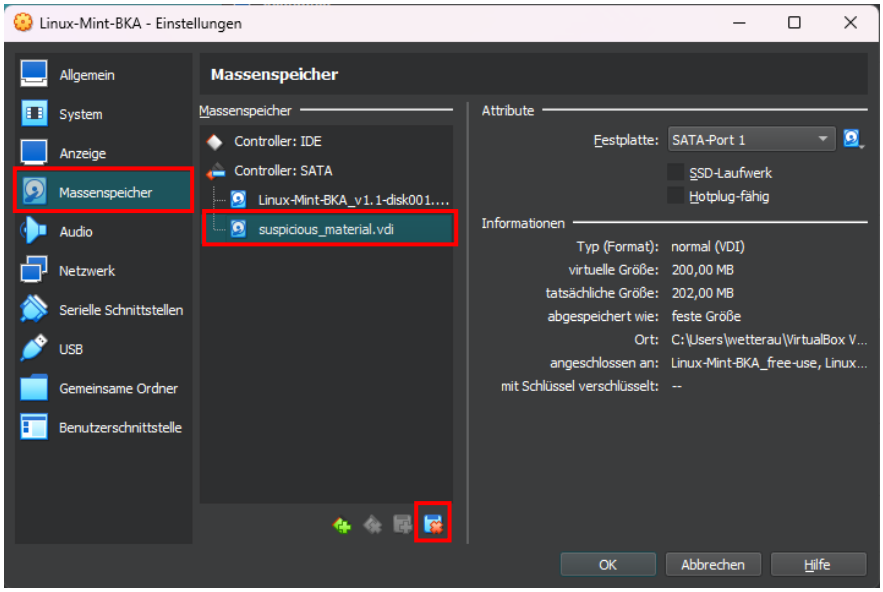
In diesem Praktikum lernen Sie wie Festplatten in Linux verwaltet werden. Darüber hinaus schauen wir dazu anfänglich das Erstellen neuer virtueller Festplatten in Virtual Box an. Weiterhin schauen wir uns an, wie wir Festplatten per LUKS verschlüsseln. Schließlich wollen wir aus weiteren Festplatten einen kleinen RAID-Verbund erstellen.

Inhalte des Praktikums:

- Festplatten formatieren und Partitionen erstellen
- Festplatten verschlüsseln
- Erstellen eines RAID-System mit **`mdadm`**

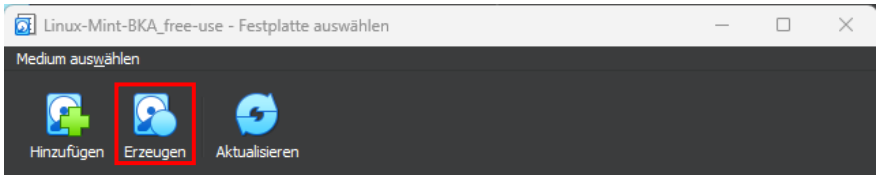
Vorbereitung

Zur Vorbereitung auf das dritte Praktikum erstellen wir zuerst neue Festplatten, an denen wir den Ablauf testen können. Dazu sollten Sie die VM ausgeschaltet haben. Die Ubuntu-Server VM aus Praktikum 2 brauchen wir an der Stelle nicht mehr. Gehen Sie zuerst wieder in die Einstellungen Ihrer Linux-Mint-BKA VM über “Ändern”. Anschließend rufen Sie den Menüpunkt “Massenspeicher” auf der linken Seite auf. Sollten Sie bis auf die Hauptfestplatte der VM weitere Festplatten eingebunden haben, können Sie diese per “Anschluss entfernen” von der VM entfernen.



Nachdem Sie alle nicht benötigten Festplatten entfernt haben, erstellen wir neue mit denen wir später arbeiten können. Dazu wählen Sie den SATA-Controller aus und klicken auf „Anschluss hinzufügen“. Im erscheinenden Dropdown-Menü wählen Sie „Platte“.

Im dem neuen Fenster wählen Sie nun oben die Option „Erzeugen“, worauf sich erneut ein Fenster mit den Einstellungen der zu erstellenden Festplatte öffnet.



Anschließend navigieren Sie mit den folgenden Einstellungen durch den Einrichtungsassistenten und klicken jeweils „Vorwärts“ und zuletzt auf „Fertigstellen“:

1. **Dateityp der virtuellen Festplatte:** VDI (VirtualBox Disk Image)
2. **Art der Speicherung:** nichts auswählen (dynamisch alloziert)
3. **Dateiname und Größe:**
    a. **Dateiname:** siehe unten
    b. **Größe:** 100 MB

Für den Dateinamen nutzen wir vier verschiedene, da wir vier neue Platten erzeugen wollen. Dazu müssen Sie den oben beschrieben Ablauf auch vier Mal wiederholen. Die Platten sollen folgende Namen bekommen:

- **`encrypted_HD.vdi`**
- **`RAID-disk_1.vdi`**
- **`RAID-disk_2.vdi`**
- **`RAID-disk_3.vdi`**

Nachdem Sie alle vier Platten erstellt haben, schließen wir zuerst nur die encrypted_HD.vdi an die VM an. Diese brauchen wir zuerst, um den Ablauf einer Partitionierung unter Linux nachzuvollziehen. Dazu klicken doppelt auf die Platte oder wählen diese aus und bestätigen unten mit „OK“. Anschließend werden Sie auf das Massenspeicher-Menü zurückgeschickt und müssten zwei Festplatten unter dem SATA-Controller sehen. Wenn das der Fall ist, bestätigen Sie erneut mit „OK“ und starten Sie anschließend wieder die VM.

Festplattenpartitionen erstellen

Im ersten Teil soll es um die einfache Arbeit der Partitionierung von Festplatten gehen. Dazu bieten sich zwei Tools an: **`fdisk`** und dessen Weiterentwicklung **`gdisk`**. Zuerst schauen wir uns an, wie wir eine Partition mit **`fdisk`** erstellen.

```
$ lsblk
$ /dev/sdb
```

Zudem schauen Sie sich die Partitionierung mit dem folgenden Befehl an:

```
$ sudo fdisk -l
```

Hier sehen wir schon mehr Informationen zur Festplatte. Starten Sie nun **`fdisk`** mit der Angabe der Bezeichnung der erstellten Festplatte aus der Ausgabe von **`lsblk`**.

```
$ sudo fdisk /dev/<Bezeichnung>
```

Anschließend startet **`fdisk`** in einen interaktiven Modus, welcher uns nun verschiedene Optionen bietet. Diese Optionen schauen wir uns mit der Eingabe von „**`m`**“ und der Bestätigung mit Enter an. Wir wollen an dieser Stelle eine neue Partition erstellen. Wie Sie dem Hilfemenü entnehmen können, müssen wir dafür „**`n`**“ eingeben. Wie Sie bereits wissen, können wir mit einem standardmäßigen MBR nur 4 Partitionen erstellen (hier gezeigt durch „4 frei“).

Erstellen Sie hier eine neue primäre Partition mit der Nummer 1 ab dem vorgegebenen Startsektor 2048. Die Partition soll die Hälfte des Datenträgers einnehmen. Berechnen Sie dazu den Endsektor selbst. Nehmen Sie Bezug auf die Ausgabe von ***fdisk****. Prüfen Sie anschließend die getätigte Einstellung erneut über die Eingabe „****p****“.*

```
$ Endsektor: (204800-2048)/2 = 101376+2048 = 103424
```

Die Einstellungen, die in einer Session von **fdisk** getätigt werden, werden nicht sofort auf den Datenträger geschrieben, sondern nur virtuell vorgehalten. Um die Einstellungen tatsächlich zu schreiben, geben sie ein „**w**“ (write) ein. Automatisch wird **fdisk** geschlossen und die Einstellungen übernommen.

Ähnlich können wir das mit **gdisk** machen. **gdisk** ist allerdings rein auf die Partitionierung mit GPT ausgerichtet, was wir direkt beim Aufrufen angezeigt bekommen. Der Aufruf dazu ist analog zum Aufruf von **fdisk**:

```
$ gdisk /dev/<Bezeichnung>
```

Mit dem Aufruf bekommen wir die Meldung, dass ein invalider GPT-Header gefunden wurde. Mit dem Bestätigen der Einstellungen überschreibt **gdisk** alle Daten des vorhandenen MBR automatisch in einen GPT-Header. Das ist für uns an der Stelle aber nicht weiter wichtig. *Bestätigen Sie einfach mit „****w****“ und anschließend „****Y****“ und rufen Sie erneut mit dem gleichen Aufruf* **gdisk** *erneut auf*. Sie sehen nun, dass die Umwandlung erfolgreich war und wir nun einen validen GPT-Header mit einem Schutz-MBR haben.

Rufen Sie **gdisk** *wieder mit der zweiten Festplatte auf und erstellen sie mit* **gdisk** *eine zweite Partition, welche den restlichen Platz auf der Platte ausfüllt. Nutzen Sie den von* **gdisk** *vorgeschlagenen Start- und Endsektor. Bestätigen Sie wieder mit „****w****“ und anschließend „****Y****“.*

```
$ sudo gdisk /dev/sdb
$ n
$ Nummer:          2
$ Startsektor:     104448
$ Endsektor:       204766
$ Filesystemtyp:   0x83
$ w
$ Y
$ q
```

Festplatten verschlüsseln

Zur Verschlüsselung von Festplatten ist das LUKS-Format in Linux Systemen üblich. Sinnvoll ist es besonders zur Verschlüsselung des ganzen Systems. Wir wollen uns auf eine der beiden erstellten Partitionen beschränken. Standardmäßig wird dazu in Linux-Systemen das Tool **cryptsetup** mit ausgeliefert. Dieses nutzen wir, um die Partition **/dev/sdb2** zu verschlüsseln. Dazu verwenden wir folgenden Befehl:

```
$ sudo cryptsetup luksFormat -c aes-xts-plain64 -s 512 -h sha512 -y /dev/sdb2
```

Der Befehl bewirkt grundlegend, dass die Gerätedatei **/dev/sdb2** im LUKS-Format verschlüsselt wird. Mit „**-c**“ übergeben wir dem Befehl den Verschlüsselungsmodus, in dem Fall AES im XTS-Modus. Alternativen sind z.B. aes-cbc-essive. Mittels „**-s 512**“ wird bestimmt, dass wir eine Schlüssellänge von 512 Bit verwenden, was für die Sicherheit von AES im XTS-Modus empfohlen wird. Mittels „**-h sha512**“ definieren wir den Hashalgorithmus, mit dem der LUKS-Header gehasht werden soll, um dessen Integrität zu prüfen. Schließlich wollen wir, dass die Passphrase zur Verschlüsselung zweimal eingeben werden muss, was wir mit „**-y**“ definieren. Mit der Bestätigung des Befehls werden wir dazu aufgefordert „**YES**“ einzugeben (und auch nur „**YES**“, nichts anderes). Dann vergeben wir eine Passphrase und bestätigen diese durch deren erneute Eingabe.

*Versuchen Sie anschließend, wie gelernt die Partition **/dev/sdb2** an einen Mountpoint zu mounten. Was stellen Sie fest?*

```
$ sudo mount /dev/sdb2 /mnt/image
$ Schlägt fehl mit der Fehlermeldung: Unbekannter Dateisystemtyp „crypto_LUKS“
```

Damit wir die verschlüsselte Partition verwenden können, müssen wir sie auf ein virtuelles Gerät mappen mittels eines Mappers. Dazu verwenden wir wieder cryptsetup mit der Aktion luksOpen. Nach der Eingabe des Passwortes wird die Partition entschlüsselt und wir können über das angegebene virtuelle Gerät auf die Partition zugreifen. Dazu nutzen wir:

```
$ sudo cryptsetup luksOpen <Gerätename> <virtuelles Gerät>
```

Nach dem Absetzen des Befehls bekommen wir das neue Gerät unter dem Verzeichnis **/dev/mapper/<virtuelles Gerät>** zur Verfügung gestellt. *Öffnen Sie nun die verschlüsselte Partition und überprüfen Sie den Inhalt des Verzeichnisses **/dev/mapper**. Erstellen Sie außerdem mit dem Befehl **mkfs** ein Dateisystem vom Typ ext4 auf dem virtuellen Datenträger. Versuchen Sie dieses erneut in Ihr Dateisystem einzuhängen. Belesen Sie sich zur Handhabung von **mkfs** selbst in den Manpages.*

```
$ sudo cryptsetup luksOpen /dev/sdb2 cryptPart
$ mkfs.ext4 /dev/mapper/cryptPart
$ mount /dev/mapper/cryptPart /mnt/image
```

Anschließend kann mit dem verschlüsselten Datenträger so umgegangen werden, wie mit jedem anderem eingehängten Dateisystem. Wollen wir den Datenträger wieder schließen, geben wir zuerst den **umount**-Befehl und nutzen dann:

```
$ sudo cryptsetup luksClose <virtuelles Gerät>
```

Vergleichen Sie die Ausgabe von lsblk mit dem geöffneten und geschlossenen virtuellen Gerät. Was fällt Ihnen auf?

- Wenn das Gerät geschlossen ist, dann ist die verschlüsselte Partition nicht direkt ersichtlich.
- Man muss also erstmal wissen, dass auf einer Partition einer LUKS-verschlüsselter Container vorhanden ist

Auch hier soll an der Stelle wieder darauf hingewiesen sein, dass die Verschlüsselung des Datenträgers nur so stark ist, wie die Passphrase, die Sie zur Verschlüsselung verwendet haben! Sollten Sie also Daten wirklich sicher verstauen wollen, dann wählen Sie eine entsprechend starke Passphrase.

Erstellen von RAID-Systemen

Schauen wir uns das Erstellen eines RAID-Systems in Linux an. Dazu nutzen wir das Tool **mdadm** (Multiple Devices Administration). Dazu muss das Tool aber erst einmal installiert werden. *Installieren Sie mdadm auf der Mint VM.*

```
$ sudo apt install mdadm -y
```

Frage: *Handelt es sich bei der Erstellung eines RAID-Verbundes mit mdadm um ein Hard- oder Software-RAID. Begründen Sie Ihre Entscheidung.*

- Software-RAID
- da wir keinen Hardware-Controller vorliegen haben
- Dadurch über das BS die Aufgabe des RAID-Controllers und hat somit eine Leistungseinbuße

Nach dem erfolgreichen Installieren des Paketes fahren Sie die Maschine herunter und öffnen Sie erneut die Massenspeicher-Einstellungen. Fügen Sie nun alle drei erstellten RAID-Festplatten nacheinander der Maschine hinzu und starten Sie die VM neu. Prüfen Sie selbstständig, ob die Platten in der Maschine verfügbar sind. Merken Sie sich auch hier die systeminternen Bezeichnungen für die Platten.

```
$ lsblk
$ sudo blkid
```

Nachdem wir die Platten verfügbar am System vorliegen haben und **mdadm** installiert ist, können wir aus den Platten auch schon einen RAID-Verbund erstellen. Dazu nutzen wir folgende Befehlssyntax:

```
$ mdadm [Modus] <RAID-Verbund> [Optionen] <RAID-Komponenten>
```

Für die Modi können wir aus Erstellen, Zusammensetzen, Monitoren oder das Behandeln einer Bestimmten Komponenten aus dem Verbund wählen. Für Raid-Verbund geben Sie dem Verbund einen Namen nach dem Schema **/dev/<Name>**. Optionen, welche beim Erstellen angegeben werden müssen sind das RAID-Level und die die Anzahl der einzuschließenden Geräte. Nach dieser Anzahl müssen auch die einzubindenden Geräte mit Pfad angegeben werden. Beispielhaft kann das Kommando wir folgt aussehen:

```
$ mdadm -C /dev/md0 --level=raid0 --raid-devices=2 /dev/sdx /dev/sdy
```

Erstellen Sie aus den drei hinzugefügten Festplatten einen RAID5-Verbund mit dem Namen **/dev/md0***. Prüfen Sie Ihren Erfolg mit den bekannten Befehlen und dem Kommando:*

```
$ mdadm --detail --scan
$ sudo mdadm -C -v /dev/md0 --level=raid5 --raid-devices=3 /dev/sdc /dev/sdd /dev/sde
$ lsblk
$ sudo blkid
```

Nachdem wir ein RAID erstellt haben, müssen wir denn Konfiguration im System manuell hinterlegen. *Nutzen Sie dazu den gerade verwendeten Befehl und* ***HÄNGEN*** *Sie dessen Ausgabe an die Datei* **/etc/mdadm/mdadm.conf** ***AN****. Anschließend öffnen Sie die Datei mit einem beliebigen Texteditor und Superuser-Rechten und schauen sie sich diese an.*

```
$ sudo mdadm --detail --scan | sudo tee -a /etc/mdadm/mdadm.conf
$ sudo nano /etc/mdadm/mdadm.conf
```

In der Datei finden Sie nun die Ausgabe am Ende wieder mit dem Aufbau:

```
ARRAY /dev/md0 metadata=1.2 name=praktikus-VB:0 UUID=[UUID-Wert]
```

Löschen Sie bitte aus dieser Zeichenkette die kompletten Angaben zum Namen und Metadaten (rot). Diese werden nicht benötigt und werden nach dem Entfernen auf den Standardwert gesetzt. Danach müssen wir noch die Initial-Ramdisk updaten, damit die Einstellungen beim Neustart übernommen werden mit dem Befehl:

```
$ sudo update-initramfs -u
```

Frage: *Welchen Gesamtnutzspeicherplatz erwarten Sie auf dem RAID-Verbund? Überlegen Sie sich einen Rechen-/Lösungsweg.*

- $(n-1)*m \mid n \sim$ *Anzahl Platten*, $m \sim$ *Speicherplatz pro Platte*
- ca. 200MB

Legen Sie nun auf dem erstellten RAID-Verbund **/dev/md0** *eine neue Partition an, die sich über den gesamten Platz erstreckt und formatieren Sie diese mit einem ext4 Dateisystem.*

```
$ sudo gdisk /dev/md0
$ n
$ Nummer:          1
$ Startsektor:     2048
$ Endsektor:       401374
$ Partitionskennung:      0x86 Linux
$ w
$ Y
$ mkfs.ext4 /dev/md0p1
```

Starten Sie dann das System neu und prüfen Sie, ob das RAID direkt beim Starten verfügbar ist.

```
$ reboot
$ lsblk
$ sudo blkid
$ Ja das RAID ist direkt beim Hochfahren der Maschine unter /dev/md0 verfügbar
$ prüfbar mit: ls -l /dev | grep md0
```

Bisher haben wir das System allerdings noch nicht eingebunden und können damit auch noch keine Daten darauf schreiben. Hängen Sie den RAID-Verbund in /mnt/raid0 ein. Erstellen Sie den Einhängepunkt selbstständig. Prüfen Sie Ihren Erfolg mit den Ihnen bekannten Befehlen.

```
$ Variante 1:      mount /dev/md0p1 /dev/raid0 -o X-mount.mkdir
$ Variante 2:
        ○   mkdir /mnt/raid0
        ○   mount /dev/md0p1 /mnt/raid0
$ lsblk
$ mount | grep md0
```

Damit haben wir das RAID eingehängt. Damit das in Zukunft automatisch beim Startup erledigt wird, müssen wir die **fstab** bearbeiten. Diese Datei regelt, die vorhandene Geräte in das System eingebunden werden. *Öffnen Sie die Datei* **/etc/fstab** *in einem Texteditor mit Superuser-Rechten*.

```
$ sudo nano /etc/fstab
```

Dazu fügen Sie am Ende der Datei die folgende Zeile ein (Die Angaben sind jeweils durch einen Tabulator getrennt):

```
/dev/md0p1      /mnt/raid0     ext4    defaults       0       0
```

Speichern Sie Ihre Einstellungen und überprüfen Sie nun wieder Ihren Erfolg, indem Sie die Maschine neu starten und prüfen, ob das RAID bereits eingehängt ist.

```
$ reboot
$ lsblk
```