Verschlüsselung von Dateisystemen- Einführung

• eingesetzte Programme und Technologien	✔ 25 Min
• Beispiel 1: Verschlüsseln von /home	✔ 10 Min
• Pause	✔ 10 Min
• Beispiel 2: LVM2 und Cryptsetup	✔ 20 Min
• Fehlerquellen	✔ 10 Min
• Literatur – Wo erfahre ich mehr?	✔ 5 Min
• Zeit für Fragen	
	= 70 Min

Eingesetzte Programme und Technologien

Der Kernel Devicemapper

mit Kernel 2.6 eingeführtes Feature, welches beliebige Geräte auf beliebige Gerätedateien umlenkt. Die neuen Gerätedateien befinden sich unter /dev/mapper

Eingesetzte Programme und Technologien

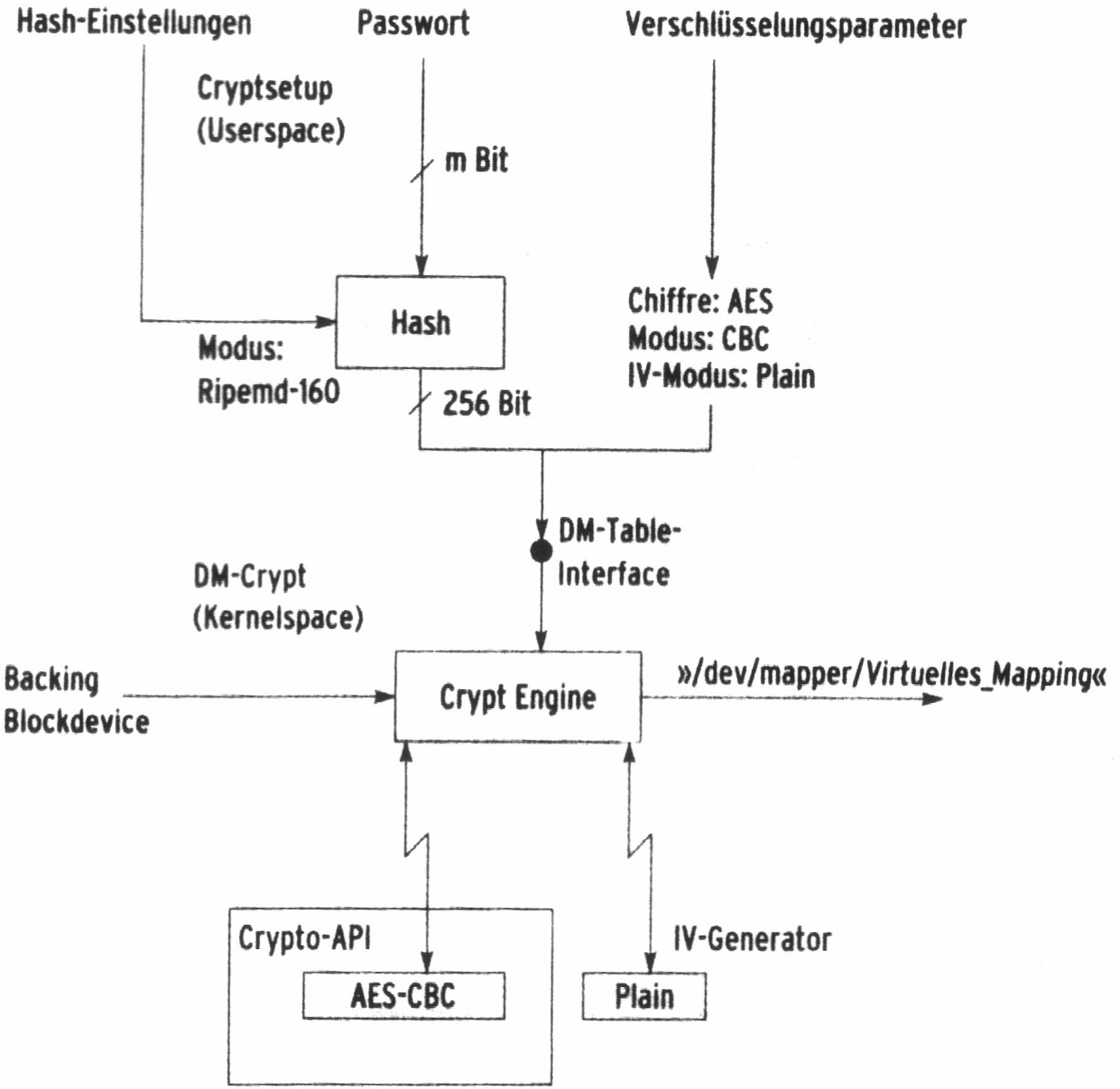
Cryptsetup

Verschlüsselungssoftware, setzt auf DM des Kernels auf
Schlüsselgenerierung und Verschlüsselungsverfahren beeinflußbar
dh. beliebig lange Passwörter wählbar, die Daten werden mit 256-Bit-Schlüssel verschlüsselt

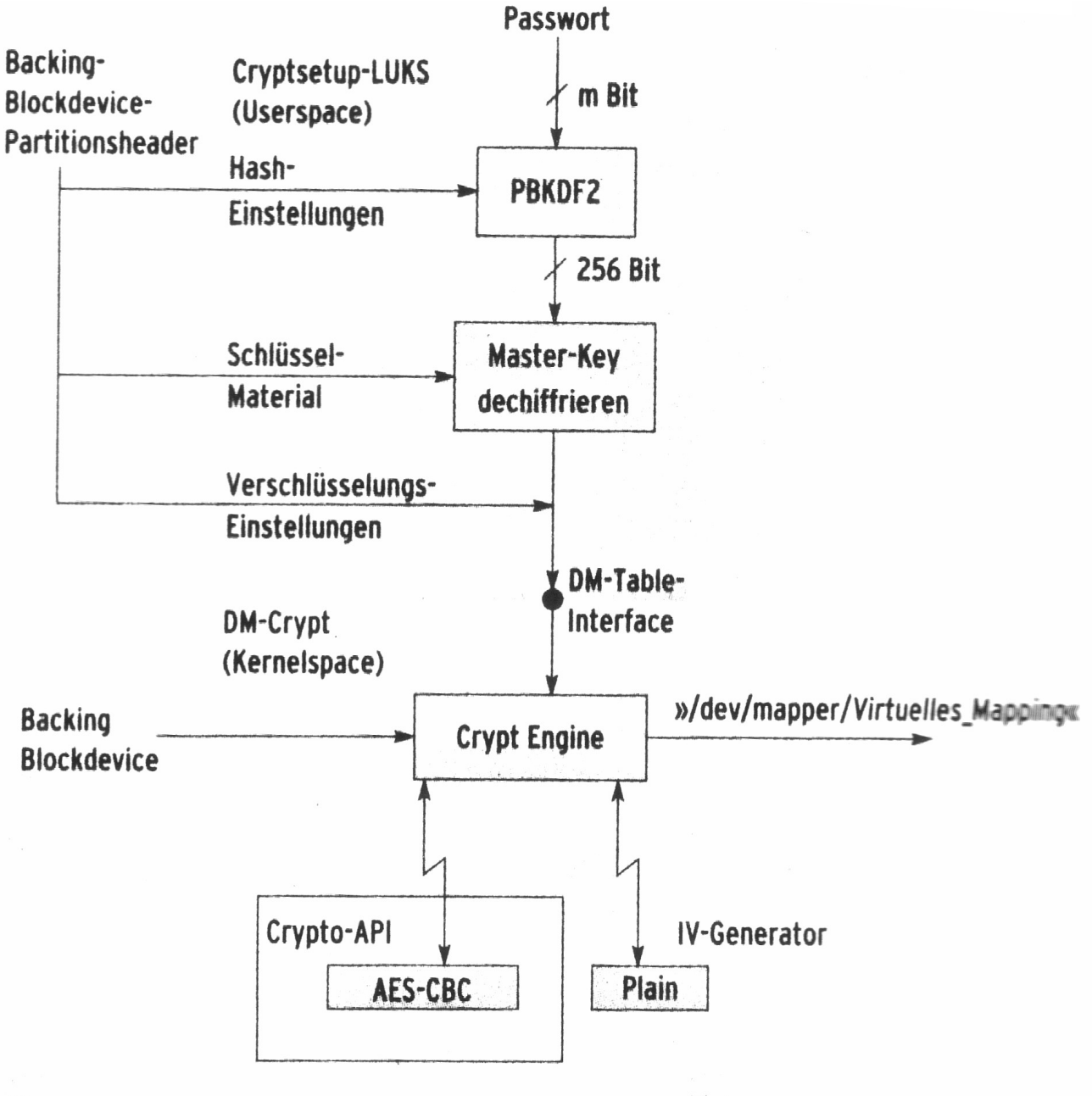
Nachteil: trennt Informationen wie die Daten verschlüsselt sind von den Informationen selbst, dh. die Parameter stehen unverschlüsselt in Skripten und Konfigurationsdateien.

--> Sind diese Informationen weg sind auch die Daten verloren

Eingesetzte Programme und Technologien



Eingesetzte Programme und Technologien



Eingesetzte Programme und Technologien

Cryptsetup-Luks:
Linux Unified Key Setup
Managementtool

setzt Masterkey ein, der Hash und damit das Passwort kann beliebig geändert werden, mehrere Passwörter möglich

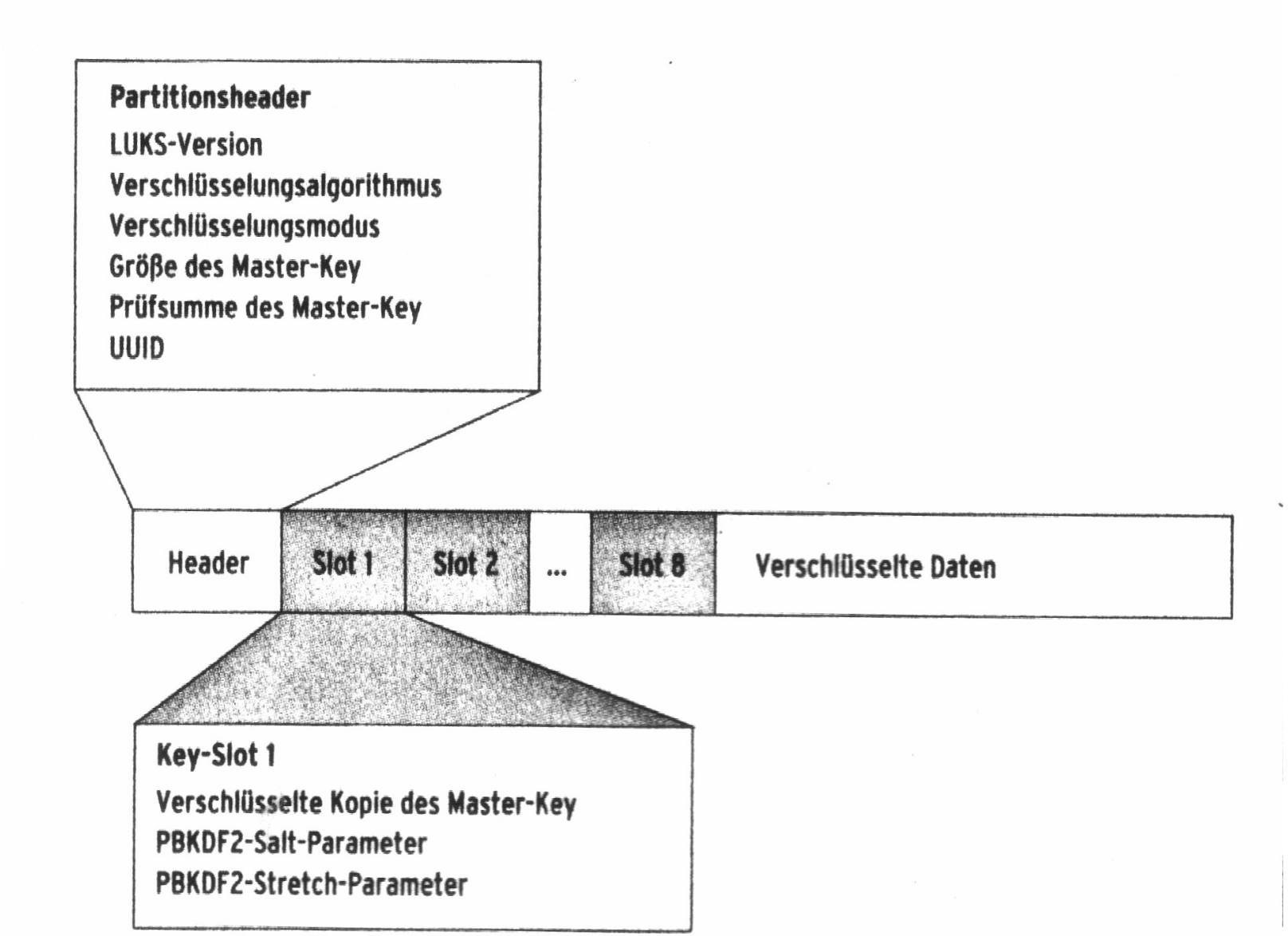
Wird erreicht durch zusätzliche Passwortmanagementschicht die die Partitionsdaten schützt

--> Spart viel Zeit und Nerven bei Änderung des Passwortes
--> auch für andere Betriebssysteme einsetzbar (Theorie?)

Eingesetzte Programme und Technologien

definiert Header für DM-Crypt-Partitionen, in dem alle Informationen für die Schlüsselableitung, sowie Algorithmus und Modus enthalten sind. Der Header ist Teil der verschlüsselten Partition, somit sind diese Informationen immer vorhanden.

Eingesetzte Programme und Technologien



Eingesetzte Programme und Technologien

Cryptsetup-Luks setzt
statt „Ripemd160“ „PBKDF2“ zur Hashbildung ein
(Password Based Key Derive Function, Version 2)

--> antiforensische Informationsspeicherung,
ermöglicht Salting und stretching, AF-Splitter

stretching: absichtlich rechenintensive Funktion zur Berechnung eines Hashwertes (Wörterbuchangriffe)
salting: zufällige Zeichenkette hinter jedem Passwort, diese Zeichenkette wird Klartext im Partitionsheader gespeichert.

Eingesetzte Programme und Technologien

Watermarkingangriffe:
--> gleicher Klartext führt zu gleichem Schlüssel

ECB :Electronic Code Book
und CBC Cipher Block Chaining
LRW-AES: Liskov, Rivest, Wagner Advanced Encryption Standard
ESSIV: Encryptet Salt-Sector Initial Vector

--> cryptsetup -c aes-cbc-essiv:sha256 -y -s 256 luksFormat
ab Kernel 2.6.10 (ohne ESSIV ab 2.6.4)

Eingesetzte Programme und Technologien

LVM- Logical Volume Manager

--> setzt auf Devicemapper des Kernels auf (LVM 2)
--> ermöglicht dynamisches Anpassen der Partitionen

LVM führt zwischen Daten und Ihrem Speicherort eine Abstraktionsschicht ein. Die laufenden Applikationen bekommen davon nichts mit, also keine Programmänderung nötig

Eingesetzte Programme und Technologien

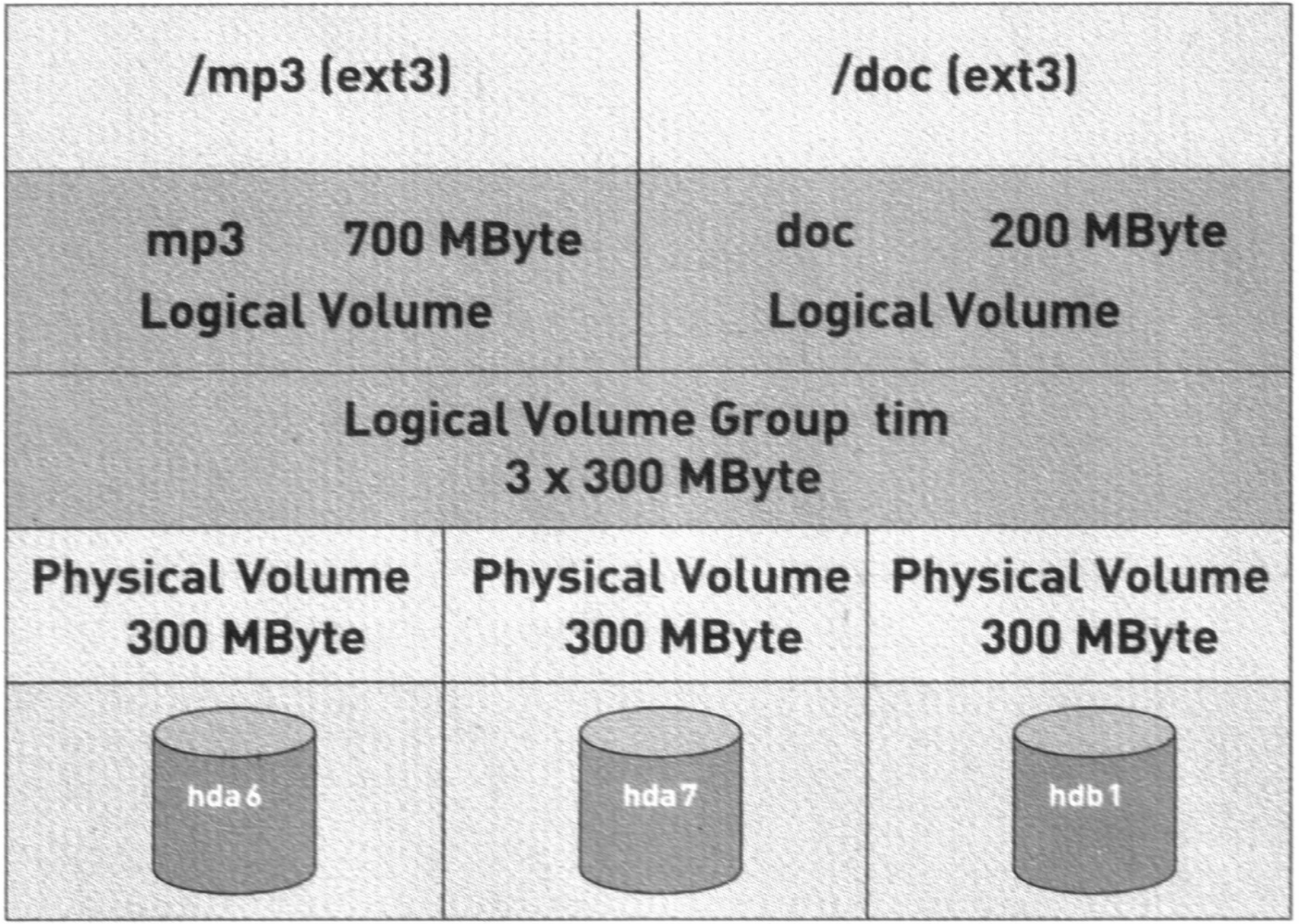
Begriffe:

PV- Physical Volume: Die Festplatte(n) oder Partition(en)

VG- Volume Group: Die „virtuelle Festplatte“

LV- Logical Volume: Die „virtuelle Partition“

Eingesetzte Programme und Technologien



Eingesetzte Programme und Technologien

pvcreate	Erzeugen eines Physical Volumes
vgcreate	Erzeugen einer Volume Group
lvcreate	Erzeugen eines Logical Volumes
pvdisplay	Eigenschaften einer Volume Group anzeigen
lvdisplay	Eigenschaften eines Logical Volumes anzeigen
vgextend	Physical Volumes einer Volume Group hinzufügen
lvextend	Logical Volume vergrößern
lvreduce	Logical Volume verkleinern
pvremove	Physical Volumes löschen
vgremove	Volume Group löschen
lvremove	Logical Volumes löschen

Eingesetzte Programme und Technologien

kleinste mögliche Speichereinheit im LVM:

PE - Physical Extend im PV
zwischen 4 MB (Voreinstellung) und 1GB
32 MB ist ein guter, in der Praxis erprobter Wert

LE- Logical Extend im LV

1 PE = 1 LE

Eingesetzte Programme und Technologien

Besonderheit:

Durch mehrere PV ist eine Art Striping möglich,
ähnlich einem raid0 erhöht es die Performance
Standard ist jedoch lineares Schreiben

Schnappschuß- Funktion möglich, das LVM „friert“ ein

--> LVM Howto liefert nähere Beschreibungen

Eingesetzte Programme und Technologien

Kernel:

Code maturity level options ---> <*> Prompt for development and/or incomplete code/drivers
General setup ---> <*> Support for hot-pluggable devices
Device Drivers > *Multi-device support (RAID and LVM).
<M> Device mapper support
<M> Crypt target support
Cryptographic options ---> <M> AES cipher algorithms

Eingesetzte Programme und Technologien

Cryptsetup, LVM2 und der Devicemapper:

Normalerweise erkennt LVM keine Blockdevices (wie Dm-Crypt) für Gebrauch als Physical Volumes !

/etc/lvm/lvm.conf ändern:

types = ["device-mapper", 16]

Beispiel 1: Verschlüsseln von /home

```
cryptsetup -c aes-cbc-essiv:sha256 -y -s 256 luksFormat /dev/hda3
cryptsetup luksOpen /dev/hda3 crypthome
mkfs.ext3 /dev/mapper/crypthome
mount /dev/mapper/crypthome /home2

umount /home2
cryptsetup luksClose crypthome

cryptsetup luksAddKey /dev/hda3
cryptsetup luksDelKey /dev/hda3 0
```

Beispiel 1: Verschlüsseln von /home

Mit einer angepassten initrd wird beim booten der Schlüssel abgefragt und das System startet normal weiter.

/etc/fstab:

# <file system>	<mount point>	<type>	<options>	<dump>	<pass>
/dev/mapper/crypthome	/home2	ext3	defaults	0	2

Alternativ kann auch das rc.local Skript angepasst werden

Pause

Beispiel 2: LVM2 und Cryptsetup

- eingesetzte Programme und Technologien
- Beispiel 1: Verschlüsseln von /home
- Pause
- Beispiel 2: LVM2 und Cryptsetup
- Fehlerquellen
- Literatur – Wo erfahre ich mehr?
- Zeit für Fragen

Beispiel 2: LVM2 und Cryptsetup

Vorgehensweise:

-- minimales Debiansystem installieren, mit cryptsetup und LVM Packeten

-- neuen angepassten Kernel erstellen

-- Verschlüsselung und LVM anlegen

-- laufendes System umkopieren

-- Bootloader anpassen

-- spezielle initrd erstellen, Neustart

Beispiel 2: Debiansystem installieren

- Normale Installation von Sarge, besser Etch auf **eine** Partition
- spätere Partition für /boot und Crypt kann jetzt schon angelegt werden
  Beispiel:

/dev/hda	30 GB	
/dev/hda1	50 MB	
/dev/hda2	2 GB	<---- bsp. hier installieren
/dev/hda3	der Rest	

- bei Sarge: Cryptsetup per Hand bauen oder aus Etch holen

Beispiel 2: angepassten Kernel erstellen

- Er muss ohne Initrd booten können
- LVM und unterstützung für Initrd muss fest im Kernel sein:

  <*>Device Mapper
  <*>Crypttarget Support
  <*>AES
- später benutzter Verschlüsselungsalgorithmus muss fest im Kernel sein

Beispielkonfigurationen und Skripte dazu auf meiner Homepage

Beispiel 2: Verschlüsselung und LVM anlegen

/etc/lvm/lvm.conf anpassen!! ->> types = ["device-mapper", 16]

```
cryptsetup -c aes-cbc-essiv:sha256 -y -s256 luksFormat /dev/hda3
cryptsetup luksOpen /dev/hda3 hda3
pvcreate /dev/mapper/hda3       #wenn hier Fehlermeldung: lvm.conf falsch!
vgcreate crypt /dev/mapper/hda3
lvcreate -L2500 -nroot crypt
lvcreate -L500 -nhome crypt
mkfs.ext3 /dev/crypt/root
mkfs.ext3 /dev/crypt/home
mkfs.ext2 /dev/hda1             # späterer /boot
```

Beispiel 2: laufendes System umkopieren

```
mkdir /target
mount /dev/crypt/root /target
mkdir /target/boot
mkdir /target/home
mount /dev/hda5 /target/boot
mount /dev/crypt/home /target/home
init s
cp -av / /target/
rm -fr /target/tmp/*
rm -fr /target/proc/*
rm -fr /target/sys/*
rm  /target/etc/mtab
nano /target/etc/fstab
```

Beispiel 2: Bootloader anpassen

```
chroot /target
umount -a
umount /proc
mount /proc
mount -a
grub
>root (hd0,0)        #/boot in /dev/hda1
>setup (hd0)
>quit
exit
```

Beispiel 2: spezielle initrd erstellen, Neustart

initrd-Skript benutzen:

http://tuxmobil.de/samsung_x20_linux_lvm2create_initrd.html

Die initrd erzeugt beim Starten eine Ramdisk, in der ein Minimalsystem die Partition entschlüsselt und ein chroot ins richtige System macht.

Es müssen nur die Parameter für die Bootpartition angepasst werden, also sehr einfach.

CRYPT="/dev/hda1"
CRYPTMOUNTPOINT="crypt"

Fehlerquellen- LVM

LVM1: PE (Physical Extend Size) steht auf 4 MB, damit sind nur ca. 250 GB Daten adressierbar. Bei LVM2 sollte die PE größer gesetzt werden, wenn mehr als 250 GB adressiert werden sollen, sonst ist es sehr langsam.

Swap nicht auf LVM
Technisch kein Problem, ist aber sehr langsam

Booten von LVM geht nicht, wenn eine initrd verwendet wird!
Deshalb /boot nicht aufs logische Volume

Fehlerquellen- LVM

Das richtige Dateisystem wählen

Online vergrößern und verkleinern des Dateisystems geht nur mit ext3 (und Kernelpatch)
Reiserfs ist nur nur offline verkleinerbar
XFS und JFS lassen sich nicht verkleinern, weder online noch offline.

nicht jedes Backuptool kommt mit LVM klar, vorher testen!
Mondo und Mkcdrec können mit LVM umgehen

Fehlerquellen- Initrd

Die initrd könnte verändert werden, so dass das Passwort zum Entschlüsseln der Partition ausgelesen werden kann

Abhilfe: md5sum der initrd erstellen, auf der Verschlüsselten Platte ablegen und beim Systemstart automatisch vergleichen

Fehlerquellen- Mensch

Nach Systemstart ist einzig das Betriebssystem für die Sicherheit der Daten zuständig!

Sichere Passwörter wählen
Computer nicht im laufendem Betrieb alleine lassen
Sicherheitsupdates regelmäßig einspielen
....

Literatur- Wo erfahre ich mehr?

Fragen und Anregungen: Lutz Willek lutz.willek@belug.de

Dieses Dokument: http://www.belug.de/~lutz/

Device-Mapper: http://sources.redhat.com/dm

LVM2: http://sourceware.org/lvm2

LVM2 HOWTO: http://deb.riseup.net/storage/lvm2/

Cryptsetup-Luks: http://www.saout.de/misc/dm-crypt/

Linux-Magazin 8/2005, S28: „Geheime Niederschrift“

Linux-User 7/2006, S.82: „Volumenkontrolle“

Ihre Fragen?

- eingesetzte Programme und Technologien
- Beispiel 1: Verschlüsseln von /home
- Beispiel 2: LVM2 und Cryptsetup
- Fehlerquellen

In eigener Sache

Vorträge und Workshops der Belug e.V. sind kostenlos
Wir bieten Hilfestellung bei vielen Problemen
Wir setzen uns Aktiv für Open Source Software ein

Bitte unterstützen Sie uns:

Spenden Sie
werden Sie Mitglied

Werden Sie Aktiv !

Dankeschön

Lutz Willek

BeLUG - Berliner Linux User Group e.V.
Lehrter Str. 53
D-10557 Berlin (Tiergarten-Mitte)
http://www.belug.de

Spenden:

Postbank Leipzig
BLZ: 860 100 90
Kto.-Nr.: 607 117 901

Intern

Die folgenden Dokumente sind nur für den privaten Gebrauch zur Weiterbildung bestimmt und dürfen nicht weitergegeben werden.