Fileserver mit LVM und NIS-Anbindung zur Rechteprüfung

Die Anforderungen

- Ich möchte einen Fileserver erstellen, der einen vorhandenen NIS-Server für die User-Berechtigungen verwendet.

  Hinweis: Zur Installation eines NIS-Servers siehe auch

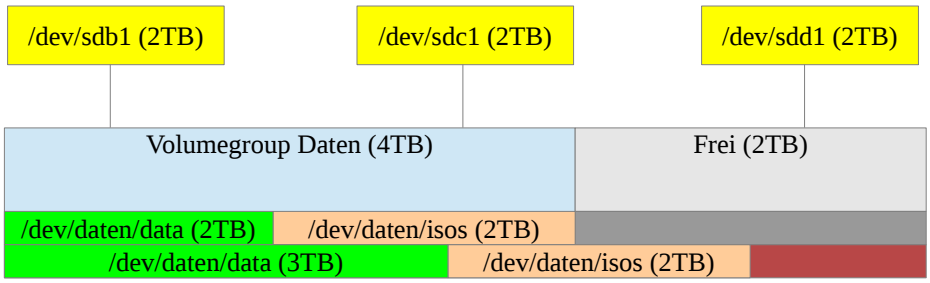
  http://mn-portal.at/index.php?qwid=40

- Weiters sollen neu erstellte Dateien/Verzeichnissen in einem definierten Gruppenordner automatisch immer der Gruppe zugeordnet werden, der der Ordner gehört, die Eigentums-Rechte sollen also immer USER.GUPPE lauten, egal von wem Datei/Verzeichnis erstellt wird.
- User die einer Gruppe nicht angehören dürfen keinen Zugriff auf ein Gruppen-Verzeichnis erhalten.
- Die Login`s (User und Gruppen) sollen zentral über NIS verwaltet werden.
- Um öffentlichen Zugriff auf öffentliche Dateien zu erlauben, möchte ich einen Webserver anbieten, der den Inhalt eines Datenverzeichnisses anzeigt.
- Damit mein Fileserver auch in Zukunft genug Platz bietet, wird der Datenpool mit LVM organisiert, wir können den verfügbaren Platz also jeder Zeit erweitern. Dieses Howto deckt also auch die Grundlagen von LVM ab.
- Zum Einsatz kommt Debian Jessie (derzeit noch Testing), die erfolgreiche Installation auf einem Server setze ich voraus.

Windows-Clients werden mit Winscp angebunden um Daten hochladen zu können, ich verzichte bewusst auf Samba.

In meinem Server stecken 3 Platten, eine für das System, die anderen beiden (je 2TB) werde ich für die Daten verwenden. Ich setze also 3 Platten voraus.

Eine kurze Information wie LVM funktioniert:



LVM fasst die ***physikalischen*** Partitionen ***/dev/sdb1*** und ***/dev/sdc1*** zu einer ***physikalischen*** Volumegroup (Daten) von 4TB Gesamtgrösse zussammen. Diese Volumegroup kann nun wiederum in einzelne ***logische*** Platten mit beliebiger Grösse zerlegt werden (hier z.B. *data* mit 2TB und *isos* mit 2TB), wobei das Gesamtvolumen der logischen Platten natürlich jenes der Volumegroup nicht überschreiten kann.

Vergrössern der Volumegroup "Daten"

Fügten wir nun noch die Platte ***/dev/sdd1*** (2TB) der Volumegroup „Daten“ hinzu, würde die Gesamt Kapazität von „Daten“ logischer Weise auf 6TB wachsen. Der so zur Verfügung stehende neue, grössere Gesamtplatz würde es wiederum erlauben die logischen Platten zu vergrössern (z.B. data auf 3TB).

Die Grösse der Platten spielt übrigens keine Rolle, es können Platten der unterschiedlichsten Grössen zu einem Volume zusammengefasst werden. Wir könnten also auch probelmlos eine Platte mit 500GB dem Volume hinzufügen.

Bitte den Vorgang nicht mit RAID verwechseln, LVM und RAID sind vollkommen unterschiedliche Techniken. LVM`s lassen sich auch als RAID zusammenfassen.

Legen wir also los.

Installation Fileserver

Nachdem Jessie installiert und lauffähig ist erstellen wir uns zunächst mit (2 x 2TB) Platten ein „Logical Volume“. LVM erlaubt es bestehende LV`s jeder Zeit um weiteren Speicherplatz zu erweitern, man muss also nicht schon am Anfang wissen, wie viel Platz man auf einem Server braucht und kann diesen nach Bedarf erweitern oder auch verringern.

Installation von lvm

```
apt-get install lvm2
```

Nach der Installation sehen wir uns unser Platten an:

```
fdisk -l
```

sollte ca. folgendes ausgeben:

```
…
Device Boot Start End Blocks Id System
/dev/sda1 * 1 18 144553+ 83 Linux
/dev/sda2 19 2450 19535040 83 Linux
/dev/sda4 2451 2610 1285200 82 Linux swap / Solaris

Disk /dev/sdb: 2000 GB, 2000592672016 bytes
...
Disk /dev/sdb doesn't contain a valid partition table

Disk /dev/sdc: 2000 GB, 2000592672016 bytes
…..
Disk /dev/sdc doesn't contain a valid partition table
```

Um unsere Platten */dev/sdb* und */dev/sdc* nun zu einem Volume zusammenzufassen müssen wir zunächst Partitionen erstellen und ihnen einen LVM-Lable mitgeben, der die Partition als zukünftigen Bestandteil eines gemeinsamen Volumes deklariert.

Ich kürze die Eingaben etwas ab, fdisk sollte eigentlich bekannt sein, eventuell ist die Änderung der System id (die Eingabe t bei fdisk) etwas neues.

```
fdisk /dev/sdb

Command (m for help): <-- n
Command action
e extended
p primary partition (1-4)
<-- p
Partition number (1-4): <-- 1
First cylinder (1-xxxxx, default 1): <-- <ENTER>
Using default value 1
.
.
Command (m for help): <-- t
Selected partition 1
Hex code (type L to list codes): <-- 8e (Linux LVM)

Command (m for help): <-- w
The partition table has been altered!
```

Gleiches machen wir mit */dev/sdc*

Ein *fdisk -l* müsste uns nun unsere Platten */dev/sdb1* und */dev/sdc1* als Linux LVM anzeigen.

Die Partitionen sind nun erstellt!

Als nächstes bereiten wir unsere Partitionen nun für eine physikalische Volumegroup vor.

```
pvcreate /dev/sdb1 /dev/sdc1
```

sollte als Ausgabe folgendes anzeigen:

```
Physical volume "/dev/sdb1" successfully created
Physical volume "/dev/sdc1" successfully created
```

Um diesen Vorgang rückgängig zu machen lässt sich mit pvremove /dev/sdb1 /dev/sdc1 alles wieder auf NULL stellen.

pvdisplay zeigt uns unser neuen **physikalischen Volumes** an und gibt die Grösse und Zuordnung zu den Volumes (hier noch none, wir sind ja noch nicht so weit) bekannt.

Nun erstellen wir unsere erste **Volumegroup** "***daten***" indem wir die physikalischen Volumes zusammenführen.

```
vgcreate daten /dev/sdb1 /dev/sdc1
```

vgdisplay:

```
  --- Volume group ---
  VG Name               daten
  System ID
  Format                lvm2
  Metadata Areas        2
  Metadata Sequence No  5
  VG Access             read/write
  VG Status             resizable
  MAX LV                0
  Cur LV                2
  Open LV               2
  Max PV                0
  Cur PV                2
  Act PV                2
  VG Size               4,00 TiB
  PE Size               4,00 MiB
  Total PE              1048574
  Alloc PE / Size       1016832 / 3,88 TiB
  Free  PE / Size       31742 / 123,99 GiB
  VG UUID               b5IZB2-TYQT-wHlx-0YTD-7Iwf-PYwO-1Fhp3X
```

Mit vgrename kann man diese Volumegroup auch umbenennen (z.B.vgrename daten data).

In dieser Volumegroup kann man nun logische Volumes mit einer festgelegten Grösse erstellen), mit:

```
lvcreate --name share --size 3TB daten
lvcreate --name isos --size 900G daten
```

wird ein LV mit dem Namen **share** in der Grösse 3TB und ein LV mit namen **isos** in der Grösse 900GB in der Volumegroup **daten** erstellt.

lvdisplay listet alle logischen Volumes auf.

```
lvdisplay
  --- Logical volume ---
  LV Path                /dev/daten/share
  LV Name                share
  VG Name                daten
  LV UUID                h7Slnm-Pvrf-PnhN-EMNW-L2TS-IVtE-KDUW7L
  LV Write Access        read/write
  LV Creation host, time fileserver, 2015-02-05 11:01:42 +0100
  LV Status              available
  # open                 1
  LV Size                3,00 TiB
  Current LE             786432
  Segments               2
  Allocation             inherit
  Read ahead sectors     auto
  - currently set to     256
  Block device           254:2

  --- Logical volume ---
  LV Path                /dev/daten/isos
  LV Name                isos
  VG Name                daten
  LV UUID                IT51kH-9Tl9-GNMT-3ov3-ig3F-UHG4-tycNn8
  LV Write Access        read/write
  LV Creation host, time fileserver, 2015-02-05 11:04:18 +0100
  LV Status              available
  # open                 1
  LV Size                900,00 GiB
  Current LE             230400
  Segments               1
  Allocation             inherit
  Read ahead sectors     auto
  - currently set to     256
  Block device           254:3
```

Wie man recht schön an der Ausgabe sehen kann, wurden durch LVM neue Devices erstellt /dev/daten/share und /dev/daten/isos, auf die man nun im System zugreifen kann.

Was noch fehlt sind Filesysteme auf unseren neuen Devices. Dabei gilt, dass man jedem Device eigene, unterschiedliche Filesysteme zuweisen kann, man ist also nicht an ein bestimmtes oder gemeinsames Filesystem gebunden. Ich nutze ext3.

```
mkfs.ext3 /dev/daten/share
mkfs.ext3 /dev/daten/isos
```

Nun sind die Devices bereit ins System eingebunden zu werden. Ansichten gibt es viele, wie man die Einbindung korrekt macht, ich binde die Devices über die rc.local ein, natürlich ist ein Eintrag in die fstab genauso möglich

```
mkdir -p /export/daten
mkdir -p /export/isos
```

in der /etc/rc.local folgende Einträge vor dem exit 0 einfügen:

```
mount /dev/daten/share /export/daten
mount /dev/daten/isos /export/isos
```

Nach dem Einbinden (mounten) kann man sich mit df -h ansehen, was nun in unserem System an Platz zur Verfügung steht.

```
/dev/mapper/daten-share             3,0T     72M  2,9T    1% /export/daten
/dev/mapper/daten-isos              886G     72M  841G    1% /export/isos
```

Vergrössern/Verkleinern eines Logical Volumes

LVM ist umfangreich dokumentiert, daher verzichte ich hier auf eine genaue Beschreibung, wie man die Grösse eines Volumes verändert. Nur soviel:

LVM stellt die Befehle lvextend und lvreduce zum Vergrössern/Verkleinern zur Verfügung, allerdings dürfen Volums nicht ins System eingebunden sein um Grössenveränderungen vorzunehmen. Man muss also zunächst ein umount durchführen und dann die Veränderungen vornehmen.

Ich vergrössere als kleines Beispiel share um 50G.

```
umount /export/share
df -h
.
.
# (share sollte nicht mehr ausgegeben werden)
.
.
lvextend -L 3050G /dev/daten/share
.
.
#Nun noch das ext3 Dateisystem checken und vergrössern
e2fsk -f /dev/daten/share
.
.
resize2fs /dev/daten/share
.
.
mount /dev/daten/share /export/daten
df -h
```

Festplatte hinzufügen

Zuerst fdisk wie oben für */dev/sdb* und */dev/sdc* beschrieben nun für */dev/sdd* durchführen.

```
pvcreate /dev/sdd1
vgextend daten /dev/sdd1
vgdisplay
```

/dev/sdd1 sollte nun dem Volume daten hinzugefügt worden sein vgdisplay sollte und nun eine neue Grösse für daten anzeigen und wir können den Platz unseren logischen Volumes zuweisen.

Eine umfangreiche Anleitung zu LVM findet man z.B. hier:

https://www.howtoforge.de/anleitung/lvm-anleitung-fur-anfanger

Apache installieren und das Verzeichnis /export/daten listen

```
apt-get -y install apache2
```

installiert den Indianer auf unserem System. Unter Jessie zeigt der DocumentRoot-Pfad auf das Verzeichnis /var/www/html, das ist also sozusagen das Startverzeichnis des Apache.

Um dieses Verzeichnis zu verbiegen braucht es wenig.

```
cd /var/www
mv html/ htmlold
ln -s /export/daten html
/etc/init.d/apache2 restart
```

Das war es schon, der Apache zeigt nun den Inhalt von */export/daten* an (http://SERVER_IP).
Das Schöne daran ist, dass der Apache die Recht erkennt und die Anzeige für Verzeichnisse unterdrückt, zu denen er keinen Zugriff hat.

Bsp.

Wir erstellen (als root) ein Verzeichnis */export/daten/test* und weisen dem Verzeichnis die Rechte 770 zu. Nun haben nur root und die Gruppe root zugriff auf dieses Verzeichnis.
Ruft man aber in einem Browser den Apache auf, wird das Verzeichnis test NICHT gelistet.
Versucht man über die URL doch Zugriff zu bekommen, wird ein Forbidden ausgegeben.

Apache-Login über die angelegten User

Wer den Apache dazu bringen möchte nur User, die auch am System angelegt sind zuzulassen kann das wie folgt erreichen:

Wir installieren mod-authz-external auf unserem Apache:

```
apt-get -y install libapache2-mod-authnz-external pwauth
```

Im Verzeichnis /etc/apache2/conf.enabled erstellen wir eine Datei auth_pam.conf mit folgendem Inhalt:

```
<Directory /var/www/html>
#    SSLRequireSSL
    AuthType Basic
    AuthName "PAM Authentication"
    AuthBasicProvider external
    AuthExternal pwauth
    require valid-user
</Directory>
```

Mit:

```
a2enmod authnz_external
```

Wird das Modul aktiviert, was aber bei Jessie schon durch die Installation geschehen ist.

Danach starten wir den Apache neu.

Ab nun werden beim Aufruf der Webseite Logindaten abgefragt, wer nicht im System registriert ist, kann sich nicht anmelden.

NIS-Anbindung

Unter

http://mn-portal.at/index.php?qwid=40

zeige ich, wie man einen NIS-Server installiert, User und Gruppen anlegt und eine Anbindung als Client erfolgt. Hier sein schlicht die Anbindung als Client nochmal wiederholt. Ich setze voraus, dass nach erfolgreicher Anbindung die Gruppen open und closed als Beispiel zur Verfügung stehen.

Die NIS-Domain ist bekannt (hier nis.world).

Wir konfigurieren den Fileserver (Debian) nun als NIS-Client

```
apt-get -y install nis
```

installiert die notwendigen Pakete

Wir editieren folgende Dateien:

Datei ***/etc/yp.conf*** einfügen

```
ypserver IP_DES_NIS_SERVERS
```

Datei ***/etc/nsswitch.conf***

```
passwd: files nis
shadow: files nis
group: files nis
hosts: files dns nis
```

Datei ***/etc/defaultdomain***

```
nis.world
```

Datei ***/etc/default/nis*** wir kontrollieren/setzen

```
NISSERVER=false
NISCLIENT=true
YPPWDDIR=/etc
```

Nach einem Reboot sollte nun ein Login mit Userdaten des NIS-Servers funktionieren.