

Laboratorium

Zarządzanie dyskami w linuxie. Partycjonowanie. Raid programowy. LVM. Naprawa.

Programy do zarządzania partycjami

- zapoznanie z programami do zarządzania partycjami
- fdisk
- cfdisk
- parted <http://www.gnu.org/software/parted/manual/parted.html>
- sfdisk
- Gparted
- QtParted
- wyświetlenie informacji o partycjach

```
# fdisk -l
```

- założenie nowej partycji przy pomocy programu fdisk
- opcje fdisk:

```
# fdisk /dev/sda
n (to create new partition)
p (for primary partition)
1 (for partition number)
default for First cylinder
t (change partition's system id)
1 (for partition number)
L (to view hex codes)
87 (for NTFS partition)
w (to write changes)
```

```
# fdisk /dev/sda
Command (m for help):      / wybieramy opcję n /
Command action
   e   extended              / partycja rozszerzona /
   p   primary partition (1-4) / partycja podstawowa /
/ program zapyta nas o numer partycji, pierwszy i rozmiar partycji /
/ rozmiar możemy podać w cylindrach, KiB, MiB lub GiB /
/ wyjście z programu po wybraniu q - bez zapisania zmian /
/ wyjście z programu po wybraniu w - z zachowaniem zmian /
```

- opcje parted

Command	Description
<code>check minor-num</code>	Perform a simple check of the file system
<code>cp from to</code>	Copy file system from one partition to another; <i>from</i> and <i>to</i> are the minor numbers of the partitions
<code>help</code>	Display list of available commands
<code>mktable label</code>	Create a disk label for the partition table
<code>mkfs minor-num file-system-type</code>	Create a file system of type <i>file-system-type</i>
<code>mkpart part-type fs-type start-mb end-mb</code>	Make a partition without creating a new file system
<code>mkpartfs part-type fs-type start-mb end-mb</code>	Make a partition and create the specified file system
<code>move minor-num start-mb end-mb</code>	Move the partition
<code>name minor-num name</code>	Name the partition for Mac and PC98 disklabels only
<code>print</code>	Display the partition table
<code>quit</code>	Quit parted
<code>rescue start-mb end-mb</code>	Rescue a lost partition from <i>start-mb</i> to <i>end-mb</i>
<code>resize minor-num start-mb end-mb</code>	Resize the partition from <i>start-mb</i> to <i>end-mb</i>
<code>rm minor-num</code>	Remove the partition
<code>select device</code>	Select a different device to configure
<code>set minor-num flag state</code>	Set the flag on a partition; <i>state</i> is either on or off
<code>toggle [NUMBER [FLAG]]</code>	Toggle the state of <i>FLAG</i> on partition <i>NUMBER</i>
<code>unit UNIT</code>	Set the default unit to <i>UNIT</i>

- korzystanie z tablicy partycji zapisanej do pliku.
- wykonaj zrzut tablicy partycji dysku do pliku

```
# sfdisk -d /dev/sda > sda.out
```

- przywróć zapisaną tablicę partycji na drugim dysku

```
# sfdisk /dev/sda < sda.out
```

Tworzenie systemów plików

```
/sbin/mkfs.ext2
/sbin/mkfs.ext3
/sbin/mkfs.reiserfs
/sbin/mkfs.msdos
/sbin/mkfs.vfat
/sbin/mkfs.xfs
```

- aby utworzyć system plików wywołujemy odpowiedni program z nazwą urządzenia jako parametrem

```
# /sbin/mkfs.ext3 /dev/sda1
```

- tworzenie obszaru wymiany

```
# mkswap /dev/sda5
```

- podmontowanie partycji i aktywacja swapu

```
# mount /dev/sda1 /katalog_docelowy
# /sbin/swapon /dev/sda5
```

- poznanie typu systemu plików

```
# file -s /dev/sda1
/dev/sda1: SGI XFS filesystem data (blkisz 4096, inosz 256, v2 dirs)
```

- lista urządzeń z utworzonymi do tej pory systemami plików

```
# blkid
/dev/md/0: UUID="4f6fc9cc-9a3a-42bd-9e47-50ecfbeb090b" TYPE="xfs"
/dev/hda1: UUID="6d7abd95-9731-4406-b154-78ee66fc6c7f" TYPE="ext2"
/dev/sda: UUID="4f6fc9cc-9a3a-42bd-9e47-50ecfbeb090b" TYPE="xfs"
/dev/sdb: UUID="4f6fc9cc-9a3a-42bd-9e47-50ecfbeb090b" TYPE="xfs")

# mount
/dev/hda1 on /boot type ext2 (rw, sync)
/dev/md/0 on /vservers type xfs (rw, noatime, usrquota)
```

RAID programowy

W systemie Linux istnieje możliwość tworzenia na dyskach programowych macierzy RAID poziomów 0, 1, 4, 5, 6, 10. Służy do tego celu usługa mdadm.

Lista i opis dostępnych rodzajów macierzy dla mdadm, w nawiasach podano nazwy parametrów programu:

- ⑩ RAID 0 (raid0, 0, stripe) - striping czyli połączenie dwóch dysków (partycji) z przeplotem danych, zwiększa się wydajność w porównaniu z pojedynczym dyskiem, obniża odporność na awarie dysków - awaria jednego dysku to utrata wszystkich danych.
- ⑩ RAID 1 (raid1, 1, mirror) - kopie lustrzane, dyski są w dwóch jednakowych kopiach, w przypadku awarii jednego drugi przejmuje rolę pierwszego. Wydajność tak jak pojedynczy dysk, duże bezpieczeństwo, wadą jest duża strata pojemności (n/2 - n-liczba dysków w macierzy)
- ⑩ RAID 4 (raid4, 4) - dane są rozpraszane na kolejnych dyskach a na ostatnim zapisywane są dane parzystości, zwiększone bezpieczeństwo danych przy zachowaniu dużej pojemności (n-1). Wymaga przynajmniej trzech dysków, wydajność ograniczona przez dysk parzystości
- ⑩ RAID 5 (raid5, 5) - rozpraszane są zarówno dane jak i informacje o parzystości na wszystkich dyskach, dzięki czemu wydajność jest wyższa niż w RAID 4; pojemność n-1, wymaga przynajmniej trzech dysków.
- ⑩ RAID 6 (raid6, 6) - jest to rzadko stosowana, rozbudowana macierz typu 5. Jediną różnicą jest dwukrotne zapisanie sum kontrolnych. Dzięki temu macierz może bez utraty danych przetrwać awarię dwóch dysków. Wymaga minimum czterech dysków, jej pojemność to n-2.
- ⑩ Tryb liniowy (linear) - czyli połączenie dwóch dysków w jeden w ten sposób że koniec pierwszego jest początkiem drugiego, nie zapewnia absolutnie żadnego bezpieczeństwa a wręcz obniża odporność na awarie dysków.

- Instalacja:

```
# apt-get install mdadm
```

- Tworzenie macierzy RAID

```
mdadm -C {$dev_RAID} --level={$rodzaj} --raid-devices={$ilość_urzadzen} {$surzadzenia}
```

-C, --create - utwórz nową macierz.

-l, --level - ustaw poziom RAID np: linear, raid0, 0, stripe, raid1, 1, mirror, raid4, 4, raid5, 5, raid6, 6; -n, --raid-devices - liczba aktywnych urządzeń (dysków) w macierzy

-x, --spare-devices - liczba zapasowych (eXtra) urządzeń w tworzonej macierzy. Zapasowe dyski można dodawać i usuwać także później.

-v --verbose - tryb "gadatliwy"

--auto=yes - automatyczne tworzenie urządzeń w /dev/ przez mdadm

- Przykłady tworzenia macierzy różnego typu:

- RAID0 na dwóch partycjach - /dev/sda1 i /dev/sdb1 jako /dev/md0

```
# mdadm -C -v /dev/md0 --level=0 -n 2 /dev/sda1 /dev/sdb1
```

- RAID1 na dwóch partycjach - /dev/sdc1 i /dev/sdd1 jako /dev/md1

```
# mdadm -C -v /dev/md1 --level=1 -n 2 /dev/sdc1 /dev/sdd1
```

- RAID5 na 4 partycjach w tym jedna jako zapasowa (hot spare), jeśli nie podasz ile ma być zapasowych

partycji domyślnie 1 zostanie zarezerwowana na zapasową

```
# mdadm -C -v /dev/md2 --level=5 -n 4 --spare-devices=1 \
/dev/sda3 /dev/sdb3 /dev/sdc3 /dev/sdd3
```

– Konfiguracja

Po utworzeniu macierzy postępujemy z nią dalej jak z partycją, czyli zakładamy system plików i odwołujemy się do niej np: jako /dev/md0 np.:

```
# mkfs.xfs /dev/md0
```

– Teraz możemy dokonać odpowiednich wpisów w pliku /etc/fstab.

– Aby macierz była automatycznie składana przy starcie systemu musimy dodać odpowiednie wpisy do pliku /etc/mdadm.conf. Na początek dodajemy wiersz, w którym wymieniamy listę urządzeń z których budowane są macierze (można używać wyrażeń regularnych):

```
DEVICE /dev/sd[abcd] [123]
```

– Następnie dodajemy definicje macierzy, możemy to zrobić automatem:

```
# mdadm --detail --scan >> /etc/mdadm.conf:
```

lub samodzielnie, poprzez dodanie następujących wierszy:

```
ARRAY /dev/md0 devices=/dev/sda1,/dev/sdb1
```

```
ARRAY /dev/md1 devices=/dev/sdc1,/dev/sdd1
```

```
ARRAY /dev/md2 devices=/dev/sda3,/dev/sdb3,/dev/sdc3,/dev/sdd3
```

– Macierze (inne niż rootfs) są składane przez rc-skrypt /etc/rc.d/rc.sysinit, na podstawie powyższych wpisów konfiguracyjnych, zatem po restarcie maszyny będziemy już z nich korzystać. Jeśli mamy macierz z głównym systemem plików, to musimy jeszcze przygotować initrd i bootloader.

– Przy ręcznym składaniu macierzy przydane może być polecenie skanujące urządzenia blokowe w poszukiwaniu istniejących macierzy:

```
# mdadm --examine --scan -v
```

– Testowanie (diagnostyka)

– skrócone informacje o macierzy:

```
# mdadm --query /dev/md0
```

– dokładne dane macierzy i jej stan

```
# mdadm --detail /dev/md0
```

```
# cat /proc/mdstat
```

LVM

– LVM (Logical Volume Management) to system zaawansowanego zarządzania przestrzenią dyskową, który jest o wiele bardziej elastyczny, niż klasyczne partycje dyskowe. To wiąże się z bardziej złożoną konstrukcją, która składa się z następujących struktur:

– PV (physical volumes) - fizyczne woluminy: są bezpośrednio związane z partycjami dyskowymi (np. /dev/hda1, /dev/sdb3).

– VG (volume groups) - grupy woluminów: składają się z co najmniej jednego PV, ich wielkość to suma objętości wszystkich PV należących do danej grupy. Uzyskane miejsce dyskowe może być dowolnie dysponowane pomiędzy kolejne LV.

– LV (logical volumes) - woluminy logiczne: są obszarami użytecznymi dla systemu (podobnie jak partycje dyskowe). LV są obszarami wydzielonymi z VG, zatem suma wielkości woluminów nie może zatem przekraczać objętości VG, do którego należą.

– Planowanie wolumenów

Mamy dwa dyski twarde po 500GB (/dev/sdb i /dev/sdc), których powierzchnię chcemy połączyć i rozdysponować

– /dev/sdb: cały dysk dla woluminów

– /dev/sdc: cały dysk dla woluminów

– Instalujemy pakiet lvm2

```
# apt-get install lvm2
```

– Budowanie woluminów

Ładujemy moduł device mappera:

```
# modprobe dm-mod (może nie być konieczne)
```

wskazujemy urządzenia Physical Volumes:

```
# pvcreate /dev/sdb /dev/sdc
```

tworzymy Volume Group o nazwie "vgsys" z wskazanych powyżej PV:

```
# vgcreate vgsys /dev/sdb /dev/sdc
```

w powstałej VG tworzymy woluminy o podanych pojemnościach (-L) i dowolnych nazwach (-n):

```
# lvcreate -L rozmiar -n nazwa vgsys
```

– Deaktywowanie wszystkie woluminów

```
# vgchange -a n
```

- Aktywowanie wszystkich woluminów

```
# vgchange -a y
```

- Narzędzia diagnostyczne

Skrócone informacje o każdym z rodzaju komponentów (PV/VG/LV) wyświetlimy za pomocą programów `pvs`, `vgs`, `lvs`. Więcej informacji uzyskamy za pomocą programów: `pvdisplay`, `vgdisplay`, `lvdisplay`.

- Powiększanie woluminu

Potęga LVM-a: - możliwość powiększania woluminu, gdy dochodzimy do wniosku, że przeznaczonego miejsca jest za mało.

```
# lvextend -l 100%VG /dev/vgsys/nazwa1
# xfs_growfs /nazwa1
```

Diagnostyka i naprawa

- Uszkodzenia fizyczne (sprawdzamy programem `badblocks` z pakietu `e2fsprogs`)

```
# badblocks -s /dev/sda
```

Program wypisze listę uszkodzonych bloków

- Naprawa systemu plików

```
# fsck.ext2
```

```
# fsck.reiserfs
```

```
# fsck.vfat
```

```
# fsck.jfs
```

```
# xfs_repair
```