LVM: dysk fizyczny, grupa wolumenów, wolumen logiczny

Poziom: podstawy

Dotyczy: RHEL/Rocky 9–10 (LVM2), Debian/Ubuntu i SLES identycznie (ten sam pakiet lvm2)

Czas: ~22 min

LVM dodaje warstwę pośrednią między fizycznym dyskiem a systemem plików: zamiast partycji o sztywnym rozmiarze przypiętej na stałe do jednego dysku, dostajesz wolumen logiczny, który da się powiększyć, przenieść na inny dysk albo rozciągnąć na kilka naraz — bez wyłączania systemu. Ta strona wprowadza cały słownik, którego reszta rozdziału o dyskach używa bez tłumaczenia: PV, VG, LV, ekstent.

Skróty użyte na tej stronie (9)

Skrót	Co znaczy
GB	Gigabyte: miliard bajtów, jednostka pojemności pamięci i plików, tu najczęściej rozmiar modelu albo pamięci karty graficznej
LV	Logical Volume: wolumen logiczny w LVM, na nim leży system plików
LVM	Logical Volume Manager: warstwa w Linuksie składająca dyski w elastyczne wolumeny
PV	Physical Volume: dysk lub partycja oddana pod LVM; w Kubernetesie PersistentVolume, trwały wolumen
RHEL	Red Hat Enterprise Linux: komercyjna dystrybucja Linuksa
SLES	SUSE Linux Enterprise Server: płatna dystrybucja Linuksa firmy SUSE, konkurent Rocky/RHEL
UUID	Universally Unique Identifier: długi, praktycznie niepowtarzalny identyfikator
VG	Volume Group: grupa wolumenów w LVM, pula złożona z dysków
XFS	system plików stworzony przez Silicon Graphics, domyślny w Rocky Linux i RHEL

Fiszka

TYLKO ODCZYT

Wypisanie wolumenów fizycznych: pvs

```
pvs
```

Wypisujemy skróconą listę wszystkich dysków i partycji oznaczonych jako wolumeny fizyczne LVM, razem z tym, do której grupy wolumenów należą i ile w nich jest wolnego miejsca.

Przykład:

```
PV         VG    Fmt  Attr PSize   PFree
/dev/sda2  vgOS  lvm2 a--   39.00g     0
/dev/sdb1  vgDB  lvm2 a--  100.00g  20.00g
```

LVM = Logical Volume Manager: warstwa w Linuksie składająca dyski w elastyczne wolumeny

TYLKO ODCZYT

Wyświetlenie pełnych szczegółów PV: pvdisplay

```
pvdisplay /dev/sdb1
```

Wyświetlamy ten sam wolumen fizyczny, ale ze wszystkimi polami naraz: rozmiarem ekstentu, ich liczbą całkowitą i wolną, identyfikatorem UUID.

Przykład:

```
  --- Physical volume ---
  PV Name               /dev/sdb1
  VG Name               vgDB
  PV Size               100.00 GiB / not usable 4.00 MiB
  Allocatable            yes
  PE Size               4.00 MiB
  Total PE              25599
  Free PE               5119
  Allocated PE          20480
  PV UUID               a1b2c3d4-...
```

PV = Physical Volume: dysk lub partycja oddana pod LVM; w Kubernetesie PersistentVolume, trwały wolumen · **UUID** = Universally Unique Identifier: długi, praktycznie niepowtarzalny identyfikator

◫◫ ZMIENIA SYSTEM

Oznaczenie dysku jako PV: pvcreate

```
pvcreate /dev/sdX
```

> (gdzie `/dev/sdX` to nazwa urządzenia blokowego, np. z wyniku `lsblk`)

Zapisujemy na urządzeniu nagłówek LVM (metadane), dzięki czemu reszta narzędzi LVM zaczyna je rozpoznawać jako materiał, z którego można budować grupy wolumenów. Działa na całym dysku (`/dev/sdX`) albo na pojedynczej partycji (`/dev/sdX1`) — obie formy są poprawne, wybór zależy od tego, czy dysk ma mieć w ogóle tabelę partycji.

Przykład:

```
  Physical volume "/dev/sdc" successfully created.
```

◫◫ **PV** = Physical Volume: dysk lub partycja oddana pod LVM; w Kubernetesie PersistentVolume, trwały wolumen · **LVM** = Logical Volume Manager: warstwa w Linuksie składająca dyski w elastyczne wolumeny · **pvcreate** = physical volume create: oddaj dysk pod LVM

◫◫ TYLKO ODCZYT

Wypisanie grup wolumenów: vgs

```
vgs
```

Wypisujemy jedną linię na grupę wolumenów: łączny rozmiar, ile PV ją tworzy, ile LV z niej korzysta, ile zostało wolnej przestrzeni.

Przykład:

```
VG    #PV #LV #SN Attr   VSize   VFree
vgOS    1   2   0 wz--n-  39.00g      0
vgDB    1   1   0 wz--n- 100.00g  20.00g
```

◫◫ **PV** = Physical Volume: dysk lub partycja oddana pod LVM; w Kubernetesie PersistentVolume, trwały wolumen · **LV** = Logical Volume: wolumen logiczny w LVM, na nim leży system plików · **vgs** = volume groups: pokaż listę grup wolumenów LVM

□□ TYLKO ODCZYT

Wyświetlenie pełnych szczegółów VG: vgdisplay

```
vgdisplay vgDB
```

Przykład:

```
  --- Volume group ---
  VG Name               vgDB
  Format                lvm2
  VG Access             read/write
  VG Size               100.00 GiB
  PE Size               4.00 MiB
  Total PE              25599
  Alloc PE / Size       20480 / 80.00 GiB
  Free  PE / Size       5119 / 20.00 GiB
  VG UUID               f00dbeef-...
```

□□ **VG** = Volume Group: grupa wolumenów w LVM, pula złożona z dysków

□□ ZMIENIA SYSTEM

Utworzenie grupy wolumenów: vgcreate

```
vgcreate vgDane /dev/sdc
```

Bierzemy jeden lub więcej wolumenów fizycznych (musiały wcześniej przejść przez `pvcreate`) i łączymy je w jedną wspólną pulę miejsca o podanej nazwie. Nazwę warto nadawać czytelną i informacyjną (`vgOS`, `vgDB`, `vgDane`) zamiast domyślnej, losowej — ułatwia to później skrypty i czytanie `lsblk`/`df -h` na wielu serwerach naraz.

Przykład:

```
  Volume group "vgDane" successfully created
```

□□ **vgcreate** = volume group create: utwórz grupę wolumenów

□□ TYLKO ODCZYT

Dołożenie kolejnego dysku do VG: vgextend

```
vgextend vgOS /dev/sdd
```

Dokładamy nowy wolumen fizyczny do już istniejącej grupy, powiększając jej łączną pojemność bez ruszania danych, które już na niej leżą. To pierwszy krok w scenariuszu „zabrakło miejsca, dokładam dysk" — pełną procedurę, razem z powiększeniem samego wolumenu logicznego i systemu plików, opisuje kolejna strona.

Przykład:

```
  Volume group "vgOS" successfully extended
```

VG = Volume Group: grupa wolumenów w LVM, pula złożona z dysków

TYLKO ODCZYT

Wypisanie wolumenów logicznych: lvs

```
lvs
```

Przykład:

```
LV      VG    Attr       LSize   Pool Origin Data%  Devices
lvOS  vgOS  -wi-ao----  20.00g                             /dev/sda2(0)
lvVAR   vgOS  -wi-ao----  19.00g                               /dev/sda2(5120)
lvDane  vgDB  -wi-ao----  80.00g                               /dev/sdb1(0)
```

lvs = logical volumes: pokaż listę wolumenów logicznych LVM

TYLKO ODCZYT

Wyświetlenie LV wraz z fizycznym rozmieszczeniem: -o +devices

```
lvs -o +devices
```

To samo, ale z dodaną kolumną `Devices`, pokazującą dokładnie, na których wolumenach fizycznych i pod jakim przesunięciem (w ekstentach) leżą dane danego LV. Kluczowe pytanie przed każdym powiększaniem czy przenoszeniem: jeśli LV korzysta z więcej niż jednego wpisu w tej kolumnie, jest rozciągnięty na kilka dysków naraz.

Przykład:

```
LV      VG    Attr       LSize   Devices
lvDane  vgDB  -wi-ao----  80.00g /dev/sdb1(0)
lvOracle   vgDB  -wi-ao---- 150.00g /dev/sdb1(20480),/dev/sdc1(0)
```

`lvOracle` zajmuje ekstenty z dwóch osobnych dysków (`sdb1` i `sdc1`) — dokładnie ten przypadek, przed którym ostrzega kolejna strona przy powiększaniu systemu plików.

LV = Logical Volume: wolumen logiczny w LVM, na nim leży system plików · **lvs** = logical volumes: pokaż listę wolumenów logicznych LVM

TYLKO ODCZYT

Wyświetlenie pełnych szczegółów LV: lvdisplay

```
lvdisplay /dev/vgDB/lvDane
```

Przykład:

```
  --- Logical volume ---
  LV Path                /dev/vgDB/lvDane
  LV Name                lvDane
  VG Name                vgDB
  LV Size                80.00 GiB
  Current LE             20480
  Segments               1
  Block device           253:2
```

LV = Logical Volume: wolumen logiczny w LVM, na nim leży system plików

ZMIENIA SYSTEM

Utworzenie wolumenu logicznego o konkretnym rozmiarze: -L

```
lvcreate -L 10G -n lvDane vgDane
```

Wydzielamy z wolnej przestrzeni grupy `vgDane` dokładnie 10 GB i tworzymy z nich nowe urządzenie blokowe dostępne pod `/dev/vgDane/lvDane` (i równolegle pod `/dev/mapper/vgDane-lvDane`). Na tym etapie LV jest surowym urządzeniem blokowym — bez systemu plików, więc zanim go zamontujemy, potrzebny jest jeszcze `mkfs`.

Przykład:

```
  Logical volume "lvDane" created.
```

GB = Gigabyte: miliard bajtów, jednostka pojemności pamięci i plików, tu najczęściej rozmiar modelu albo pamięci karty graficznej · **LV** = Logical Volume: wolumen logiczny w LVM, na nim leży system plików · **lvcreate** = logical volume create: utwórz wolumen logiczny

ZMIENIA SYSTEM

Wykorzystanie całej wolnej przestrzeni: -l 100%FREE

```
lvcreate -l 100%FREE -n lvDane vgDane
```

Zamiast liczby gigabajtów (`-L`), flaga `-l` przyjmuje liczbę ekstentów albo procent — `100%FREE` znaczy „weź całą pozostałą wolną przestrzeń tej grupy wolumenów". Wygodne przy pierwszym LV w świeżo utworzonej VG, gdy z góry wiadomo, że nic więcej nie będzie z niej korzystać.

Przykład:

```
  Logical volume "lvDane" created.
```

LV = Logical Volume: wolumen logiczny w LVM, na nim leży system plików · **VG** = Volume Group: grupa wolumenów w LVM, pula złożona z dysków · **lvcreate** = logical volume create: utwórz wolumen logiczny

OSTROŻNIE · NIEODWRACALNE

Usunięcie wolumenu logicznego: lvremove

```
lvremove /dev/vgDane/lvDane
```

Kasujemy wolumen logiczny, a zajmowane przez niego ekstenty wracają do puli wolnej przestrzeni grupy. Operacja pyta o potwierdzenie, jeśli LV jest zamontowany albo aktywny — najpierw odmontowujemy go (`umount`), inaczej dostajemy odmowę.

Przykład:

```
Do you really want to remove active logical volume vgDane/lvDane? [y/n]: y
  Logical volume "lvDane" successfully removed
```

LV = Logical Volume: wolumen logiczny w LVM, na nim leży system plików

TYLKO ODCZYT

Odłączenie dysku od grupy: vgreduce

```
vgreduce vgDane /dev/sdd
```

Operacja odwrotna do `vgextend`: usuwa wskazany wolumen fizyczny z grupy, o ile żaden LV nie ma na nim aktualnie żadnych ekstentów. Jeśli dane tam leżą, trzeba je najpierw przenieść na inny PV tej samej grupy (`pvmove`), zanim `vgreduce` się powiedzie.

Przykład:

```
  Removed "/dev/sdd" from volume group "vgDane"
```

LV = Logical Volume: wolumen logiczny w LVM, na nim leży system plików · **PV** = Physical Volume: dysk lub partycja oddana pod LVM; w Kubernetesie PersistentVolume, trwały wolumen

OSTROŻNIE · NIEODWRACALNE

Zdjęcie oznaczenia PV: pvremove

```
pvremove /dev/sdd
```

Ostatni krok odwrotny do `pvcreate` — usuwa nagłówek LVM z urządzenia, po którym system przestaje je traktować jako wolumen fizyczny. Wymaga, żeby dysk nie należał już do żadnej grupy wolumenów (czyli żeby `vgreduce` wykonał się wcześniej).

Przykład:

```
  Labels on physical volume "/dev/sdd" successfully wiped.
```

PV = Physical Volume: dysk lub partycja oddana pod LVM; w Kubernetesie PersistentVolume, trwały wolumen · **LVM** = Logical Volume Manager: warstwa w Linuksie składająca dyski w elastyczne wolumeny

TYLKO ODCZYT

Wyświetlenie wszystkich warstw w jednym drzewie: lsblk

```
lsblk
```

Widzimy od razu wszystkie trzy poziomy LVM w jednym drzewie: dysk, partycję oznaczoną jako PV i wolumeny logiczne zbudowane z niej wewnątrz VG.

Przykład:

```
NAME              MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sdb                 8:16   0  100G  0 disk
└─sdb1              8:17   0  100G  0 part
  ├─vgDB-lvOracle   253:2    0   80G  0 lvm  /var/lib/mysql
  └─vgDB-lvOracle    253:3    0   15G  0 lvm  /var/lib/app
```

Wcięcie pod `sdb1` i typ `lvm` w ostatniej kolumnie to sygnał: to, co widać niżej, nie jest już partycją, tylko wolumenem logicznym zbudowanym na jej bazie.

LVM = Logical Volume Manager: warstwa w Linuksie składająca dyski w elastyczne wolumeny · **PV** = Physical Volume: dysk lub partycja oddana pod LVM; w Kubernetesie PersistentVolume, trwały wolumen · **VG** = Volume Group: grupa wolumenów w LVM, pula złożona z dysków · **lsblk** = list block devices: pokaż dyski i partycje

TYLKO ODCZYT

Wyświetlenie surowej warstwy device-mapper: dmsetup ls

```
dmsetup ls
```

LVM sam w sobie nie zapisuje danych — deleguje tę pracę jądrowemu mechanizmowi *device-mapper*, a `/dev/mapper/vgDB-lvOracle` to tak naprawdę tylko czytelna nazwa dla urządzenia `dm-2`. `dmsetup ls` pokazuje tę warstwę bez pośrednictwa narzędzi LVM — przydatne, gdy coś nie zgadza się między tym, co mówi `lvs`, a tym, co faktycznie widać w `/dev`.

Przykład:

```
vgDB-lvOracle (253:2)
vgDB-lvOracle (253:3)
vgOS-lvOS (253:0)
vgVAR-lvVAR (253:1)
```

LVM = Logical Volume Manager: warstwa w Linuksie składająca dyski w elastyczne wolumeny

TYLKO ODCZYT

Wszystkie polecenia

> **Pojęcie: rodzina narzędzi LVM2.** Wszystkie polecenia LVM trzymają się jednego wzorca nazewnictwa: pierwsza litera mówi, na jakim poziomie działają (`p` = physical volume, `v` = volume group, `l` = logical volume), a końcówka mówi co robią (`s` = short list, `display` = pełne szczegóły, `create` / `remove` / `extend` / `reduce` = tworzenie i modyfikacja). Znajomość tego wzorca pozwala się domyślić nazwy polecenia, nawet jeśli nigdy się go nie widziało.

Poziom	Lista skrócona	Pełne szczegóły	Tworzenie	Usuwanie	Rozszerzanie/ zmniejszanie
Physical volume (PV)	`pvs`	`pvdisplay`	`pvcreate`	`pvremove`	—
Volume group (VG)	`vgs`	`vgdisplay`	`vgcreate`	`vgremove`	`vgextend` / `vgreduce`
Logical volume (LV)	`lvs`	`lvdisplay`	`lvcreate`	`lvremove`	`lvextend` / `lvreduce`

> **Uważaj**
>
> - `lvremove` i `vgreduce` nie proszą o dodatkowe potwierdzenie poza prostym `[y/n]` — na produkcyjnym serwerze zawsze warto sprawdzić `lvs -o +devices`, zanim cokolwiek zostanie usunięte, żeby nie trafić w LV, który ktoś inny właśnie aktywnie wykorzystuje.
> - Świeżo utworzony LV (`lvcreate`) to samo surowe urządzenie blokowe — bez `mkfs` nie da się go zamontować.
> - `pvremove` odmówi działania, dopóki PV należy do jakiejś grupy wolumenów — kolejność jest zawsze `lvremove` → `vgreduce` → `pvremove`, nigdy odwrotnie.

LVM = Logical Volume Manager: warstwa w Linuksie składająca dyski w elastyczne wolumeny · **PV** = Physical Volume: dysk lub partycja oddana pod LVM; w Kubernetesie PersistentVolume, trwały wolumen · **VG** = Volume Group: grupa wolumenów w LVM, pula złożona z dysków · **LV** = Logical Volume: wolumen logiczny w LVM, na nim leży system plików

Z życia wzięte

Z ŻYCIA WZIĘTE

> Środa rano. Ola prosi o więcej miejsca pod bazę aplikacji zespołu na pg01, zanim wdrożenie ruszy po południu. Tomek tłumaczy jej na tablicy, że dysk fizyczny, partycja, PV, VG i LV to pięć osobnych warstw, i że dokładając nowy dysk do vgPostgres nie trzeba ruszać niczego innego na tym serwerze.

Po co to istnieje

Klasyczna partycja (`/dev/sda1`) jest przypięta na stałe do jednego, konkretnego miejsca na jednym, konkretnym dysku — jej rozmiar ustala się raz, w momencie tworzenia tabeli partycji, i powiększenie jej wymaga wolnego miejsca leżącego bezpośrednio za nią na tym samym nośniku. Jeśli tego miejsca zabraknie — a dysk, na którym leży partycja, jest już pełny, mimo że w serwerowni stoi drugi, prawie pusty dysk — jedynym wyjściem przy klasycznym partycjonowaniu jest migracja całej zawartości na większy dysk albo dołożenie zupełnie osobnego punktu montowania. Oba rozwiązania wymagają przestoju albo trwałej zmiany struktury katalogów.

LVM (*Logical Volume Manager*) rozwiązuje ten problem, wstawiając między fizyczny dysk a system plików dodatkową warstwę abstrakcji, przez którą przechodzą wszystkie trzy pojęcia z Fiszki:

> **Pojęcie: wolumen fizyczny (PV, physical volume).** Dysk albo partycja oznaczona poleceniem `pvcreate` jako surowy materiał, z którego LVM może budować przestrzeń. Sam w sobie PV nie jest jeszcze niczym, co dałoby się zamontować — to tylko zgłoszenie do LVM: „to urządzenie jest teraz Twoje, licz je do puli".

> **Pojęcie: grupa wolumenów (VG, volume group).** Wspólna pula miejsca złożona z jednego albo wielu PV, utworzona poleceniem `vgcreate`. Grupa nie „pamięta" już, z którego konkretnie dysku pochodzi dany kawałek wolnej przestrzeni — to jest właśnie ten poziom, na którym znika ograniczenie „partycja żyje na jednym dysku".

> **Pojęcie: wolumen logiczny (LV, logical volume).** Właściwy, gotowy do formatowania i montowania odpowiednik partycji — urządzenie blokowe wycięte z wolnej przestrzeni grupy poleceniem `lvcreate`. LV może rosnąć (a przy niektórych systemach plików i maleć), może zajmować ekstenty rozproszone po kilku dyskach fizycznych naraz, i może zostać żywcem przeniesiony między dyskami (`pvmove`) bez odmontowywania i bez przestoju usługi, która z niego korzysta.

Innymi słowy: partycja wiąże Cię z geometrią jednego dysku. LVM wiąże Cię tylko z pojemnością grupy wolumenów — a do tej grupy zawsze można dołożyć kolejny fizyczny dysk.

Analogia

Wyobraź sobie **system magazynowy z paletami towaru**.

Palety towaru (**wolumeny fizyczne**) trafiają na wjazd do hali i zostają oznakowane kodem kreskowym systemu magazynowego — to jest dokładnie to, co robi `pvcreate`: nie zmienia samego towaru, tylko zgłasza go do systemu ewidencji. Paleta może stać w dowolnym miejscu hali, system i tak wie, że należy do zasobów magazynu.

Cała hala razem ze wszystkimi paletami, które w niej stoją, to **grupa wolumenów** — jedna wspólna pula towaru, z której magazynier wydaje zamówienia. Kiedy przychodzi zlecenie na 80 sztuk danego produktu, magazynier nie zastanawia się, z której konkretnie palety je wziąć — bierze z dowolnej, czasem z dwóch naraz, jeśli jedna nie starcza. To jest właśnie **wolumen logiczny**: konkretne zamówienie klienta, wydane z puli hali, zupełnie niezależnie od tego, na której palecie towar fizycznie leżał.

Gdy hala się zapełnia, magazyn dobudowuje drugą halę obok i włącza ją do tej samej puli (`vgextend`) — zamówienia klientów (LV) mogą teraz sięgać po towar z obu hal naraz, bez informowania klienta, że cokolwiek się zmieniło. To dokładnie to, co dzieje się przy powiększaniu przestrzeni dyskowej na kolejnej stronie: dokładasz dysk do grupy, a wolumen logiczny rośnie, nie przejmując się tym, że część jego pojemności leży teraz fizycznie gdzie indziej.

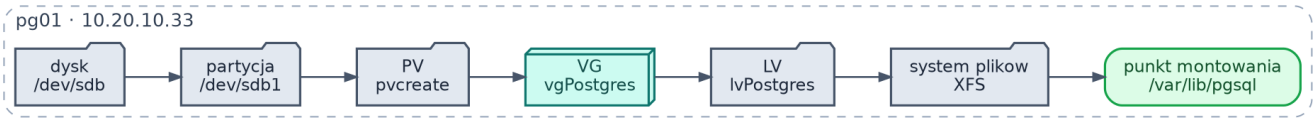
Analogia ma granicę: fizyczna paleta w magazynie jest niepodzielna — albo cała trafia do zamówienia, albo wcale. LVM dzieli każdy PV na dużo mniejsze, jednakowe porcje, więc jedno „zamówienie” (LV) może wykorzystać tylko fragment jednej palety i fragment drugiej. Ta jednostka podziału ma swoją nazwę — ekstent — i tłumaczy ją następna sekcja.

Jak to działa

Rozwiń: jak to działa pod spodem

> **Pojęcie: ekstent (PE i LE).** LVM nie operuje na całych dyskach ani na pojedynczych bajtach — dzieli każdy wolumen fizyczny na jednakowe, ponumerowane porcje o stałym rozmiarze (domyślnie 4 MiB), zwane ekstentami fizycznymi (*Physical Extent*, PE). Wolumen logiczny z kolei składa się z ekstentów logicznych (*Logical Extent*, LE) — a każdy LE jest po prostu mapowaniem na konkretny PE, na konkretnym PV. `lvcreate -L 10G` w praktyce znaczy „znajdź 2560 wolnych ekstentów po 4 MiB (10 GB ÷ 4 MiB) gdziekolwiek w tej grupie wolumenów i zmapuj je jako ten LV" — LVM nie musi ich

znaleźć obok siebie ani na jednym dysku, może rozrzucić je swobodnie po dostępnej przestrzeni. Dokładnie dlatego `lvs -o +devices` bywa pokazuje więcej niż jedno urządzenie w kolumnie `Devices`.



Pięć warstw między fizycznym dyskiem a punktem montowania w konwencji vg/lv Zielonej. (kliknięcie otwiera pełny rozmiar)

Pod spodem tego wszystkiego leży jądrowy mechanizm *device-mapper*: LVM to w gruncie rzeczy zestaw narzędzi w przestrzeni użytkownika, które generują dla jądra tabelę mapowań („blok logiczny N tego LV znajduje się pod blokiem M na tym PV") i przekazują ją do device-mappera, a ten tworzy wirtualne urządzenie blokowe realizujące tę tabelę. `/dev/mapper/vgDB-lvOracle` to czytelna nazwa nadana takiemu wirtualnemu urządzeniu — fizycznie to dowiązanie do `/dev/dm-2` albo podobnego. `dmsetup ls` z Fiszki pokazuje właśnie tę surową listę, bez żadnej kosmetyki ze strony LVM.

Pojęcie: cienkie provisionowanie (thin provisioning). Standardowy LV rezerwuje realne ekstenty w chwili tworzenia — 10 GB LV to 10 GB odjęte od `VFree` grupy, natychmiast i na stałe, niezależnie od tego, ile faktycznie danych tam trafi. Cienkie provisionowanie odwraca tę zasadę: tworzysz najpierw pulę cienką (*thin pool*), a potem wolumeny cienkie (*thin LV*), którym możesz zadeklarować rozmiar większy niż fizycznie dostępne miejsce w puli — ekstenty przydzielają się dopiero w miarę faktycznego zapisu danych, nie w momencie utworzenia wolumenu. To wygodne przy dyskach maszyn wirtualnych, które rzadko wykorzystują całą zadeklarowaną pojemność od razu, ale niesie realne ryzyko: jeśli suma rzeczywiście zapisanych danych we wszystkich cienkich wolumenach przekroczy fizyczną pojemność puli, zapis się nie powiedzie, mimo że każdy pojedynczy `lvs` z osobna pokazuje mnóstwo „wolnego” miejsca. Ta strona wprowadza tylko samo pojęcie — pełna konfiguracja puli cienkiej to temat osobny.

Krok po kroku

1. Oznacz nowy dysk jako wolumen fizyczny.

```
pvcreate /dev/sdc
```

```
  Physical volume "/dev/sdc" successfully created.
```

2. Utwórz z niego grupę wolumenów.

```
vgcreate vgDane /dev/sdc
```

```
  Volume group "vgDane" successfully created
```

3. Wytnij z niej wolumen logiczny o potrzebnym rozmiarze.

```
lvcreate -L 50G -n lvAplikacja vgDane
```

```
  Logical volume "lvAplikacja" created.
```

4. Dopiero teraz nałóż system plików — LV samo w sobie jest surowym urządzeniem.

```
mkfs.xfs /dev/vgDane/lvAplikacja
```

```
meta-data=/dev/vgDane/lvAplikacja isize=512    agcount=4, agsize=3276800 blks
data     =                       bsize=4096   blocks=13107200, imaxpct=25
```

5. Zamontuj i sprawdź całość od góry do dołu.

```
mkdir -p /opt/aplikacja
mount /dev/vgDane/lvAplikacja /opt/aplikacja
lsblk
df -h /opt/aplikacja
```

```
NAME               MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
sdc                  8:32   0  50G  0 disk
└─vgAplikacja-lvAplikacja 253:4   0  50G  0 lvm  /opt/aplikacja

Filesystem                   Size  Used Avail Use% Mounted on
/dev/mapper/vgAplikacja-lvAplikacja 50G   33M   50G   1% /opt/aplikacja
```

Pięć poleceń, trzy warstwy (PV → VG → LV) i gotowy system plików, który w każdej chwili można powiększyć dokładając kolejny dysk do `vgDane` — bez odmontowywania, co pokazuje następna strona.

Częste błędy

Próba zamontowania świeżo utworzonego LV bez `mkfs`.

```
mount /dev/vgDane/lvDane /mnt
```

```
mount: /mnt: wrong fs type, bad option, bad superblock on /dev/mapper/vgDane-lvDane, missing
codepage or helper program, or other error.
```

`lvcreate` tworzy wyłącznie surowe urządzenie blokowe wypełnione zerami albo poprzednią zawartością dysku — bez struktury systemu plików nie ma czego montować. Naprawa: `mkfs.xfs` (albo `mkfs.ext4`) przed pierwszym `mount`.

`pvremove` na wolumenie, który wciąż należy do grupy.

```
pvremove /dev/sdd
```

```
  Can't pvremove physical volume "/dev/sdd" with VG "vgDane" still there
```

LVM celowo blokuje tę operację, żeby nie dało się przypadkiem wyrwać dysku, na którym leżą aktywne dane. Właściwa kolejność to najpierw `vgreduce vgDane /dev/sdd` (o ile dysk nie trzyma żadnych ekstentów żadnego LV), dopiero potem `pvremove`.

`vgreduce` na dysku, który wciąż trzyma dane LV.

```
vgreduce vgDane /dev/sdd
```

```
  Physical volume "/dev/sdd" still in use
```

Sam fakt, że dysk należy do grupy, nie znaczy, że jest pusty — jeśli którykolwiek LV ma tam swoje ekstenty (sprawdź `pvdisplay -m /dev/sdd` albo `lvs -o +devices`), trzeba je najpierw przenieść na inny PV tej samej grupy poleceniem `pvmove /dev/sdd`, dopiero potem `vgreduce` się powiedzie.

Założenie, że `lvcreate -l 100%FREE` w grupie z kilkoma dyskami da LV na jednym, przewidywalnym dysku.

```
lvcreate -l 100%FREE -n lvOracle vgDane
lvs -o +devices vgDane/lvOracle
```

```
LV      VG     Attr       LSize   Devices
lvOracle  vgDane -wi-ao---- 150.00g /dev/sdc(0),/dev/sdd(0)
```

Efekt jest poprawny — LV faktycznie zajął całą wolną przestrzeń — ale rozciągnięty jest na dwóch dyskach naraz, o czym łatwo zapomnieć. Konsekwencje tego wychodzą na jaw dopiero przy próbie usunięcia jednego z dysków albo przy powiększaniu systemu plików wskazującym konkretne urządzenie — dokładny temat następnej strony.

□□ Ćwiczenia

> Wszystko poniżej wykonujesz w `/tmp`, na plikach obrazów podłączonych jako urządzenia pętli zwrotnej (`losetup`) — LVM widzi je identycznie jak prawdziwe dyski, ale żaden realny dysk systemu nie jest ruszany. Operacje LVM (`pvcreate`, `vgcreate`, `lvcreate`, `mount`) wymagają uprawnień administratora — reszta poleceń (`pvs`, `vgs`, `lvs`, `lsblk`) działa bez nich.

Ćwiczenie 1 — od zera do zamontowanego LV na dwóch „dyskach"

```
cd /tmp
mkdir -p lvm-cwiczenie && cd lvm-cwiczenie

# Dwa pliki udające dwa osobne dyski po 200 MB
truncate -s 200M dysk1.img
truncate -s 200M dysk2.img

sudo losetup -fP dysk1.img
sudo losetup -fP dysk2.img
losetup -a | grep lvm-cwiczenie
```

Zapisz sobie nazwy urządzeń pętli (np. `/dev/loop10`, `/dev/loop11` — u Ciebie numery mogą być inne), a następnie zbuduj z nich VG i LV:

```
LOOP1=/dev/loopXX   # podmień na swoje z poprzedniego kroku
LOOP2=/dev/loopYY

sudo pvcreate "$LOOP1" "$LOOP2"
sudo vgcreate vgCwiczenie "$LOOP1" "$LOOP2"
sudo lvcreate -l 100%FREE -n lvCwiczenie vgCwiczenie
sudo mkfs.xfs /dev/vgCwiczenie/lvCwiczenie
```

```
sudo mkdir -p /mnt/lvm-cwiczenie
sudo mount /dev/vgCwiczenie/lvCwiczenie /mnt/lvm-cwiczenie
lsblk | grep -A3 loop
df -h /mnt/lvm-cwiczenie
```

Jaki rozmiar pokazuje `df -h` i dlaczego jest on nieco mniejszy niż suma 400 MB obu plików?

Odpowiedź

`df -h` pokaże trochę poniżej 400 MB — różnica to metadane LVM (nagłówek na każdym PV) oraz narzut samego systemu plików XFS (superbloki, tablice i-węzłów). `lsblk` pokaże strukturę: dwa urządzenia `loop`, każde z warstwą `vgCwiczenie-lvCwiczenie` pod spodem — ten sam LV wyświetlony dwa razy, raz pod każdym PV, z którego faktycznie korzysta, bo `lvcreate -l 100%FREE` na dwóch dyskach rozciągnął LV na oba naraz.

Sprzątanie po ćwiczeniu:

```
sudo umount /mnt/lvm-cwiczenie
sudo vgremove -f vgCwiczenie
sudo pvremove "$LOOP1" "$LOOP2"
sudo losetup -d "$LOOP1"
sudo losetup -d "$LOOP2"
rm -f dysk1.img dysk2.img
```

Ćwiczenie 2 — odłączenie jednego dysku od grupy

Korzystając ze skonfigurowanego wcześniej `vgCwiczenie` (przed sprzątaniem z Ćwiczenia 1, albo odtwórz je od nowa z jednym LV mniejszym niż jeden dysk, np. `lvcreate -L 100M`), spróbuj:

```
sudo vgreduce vgCwiczenie "$LOOP2"
```

Sprawdź, czy się powiodło. Jeśli nie — co pokazuje `lvs -o +devices` i jaki jest właściwy następny krok?

Odpowiedź

Jeśli LV zajmuje ekstenty na obu urządzeniach pętli (tak jak przy `100%FREE` w Ćwiczeniu 1), `vgreduce` odmówi komunikatem `Physical volume "..." still in use`. `lvs -o +devices` pokaże, które PV trzyma dane. Rozwiązanie to `pvmove "$LOOP2"` — przeniesienie wszystkich ekstentów z tego urządzenia na pozostałe wolne miejsce w grupie — a dopiero potem `vgreduce`. Jeśli natomiast LV mieści się w całości na jednym urządzeniu (np. mały LV utworzony przez `-L 100M` zaraz po `vgcreate`, zanim cokolwiek trafiło na drugi dysk), `vgreduce` na drugim, nietkniętym urządzeniu zadziała od razu, bez `pvmove`.

Pytania kontrolne

1. Czym różni się wolumen fizyczny od zwykłej, niezmienionej partycji?

Odpowiedź

Zwykła partycja nie ma żadnych dodatkowych metadanych poza tabelą partycji dysku — jest po prostu zakresem bloków. Wolumen fizyczny to ta sama partycja (albo cały dysk) po wykonaniu `pvcreate`, które zapisuje na niej nagłówek LVM, zgłaszając ją do wspólnej puli, z której później buduje się grupy wolumenów. Różnica jest więc czysto administracyjna — LVM „wie" o PV, nie wie o zwykłej, nieoznaczonej partycji.

2. Masz grupę wolumenów złożoną z trzech dysków. Czy jeden wolumen logiczny może wykorzystywać przestrzeń tylko z dwóch z nich?

Odpowiedź

Tak. LVM przydziela LV ekstenty z dowolnego wolnego miejsca w grupie, niezależnie od tego, z ilu fizycznie PV korzysta. LV może w całości mieścić się na jednym dysku, rozciągać się na dwóch z trzech, albo obejmować wszystkie trzy naraz — sprawdzisz to zawsze przez `lvs -o +devices`.

3. Co dokładnie robi `lvcreate -L 10G`, jeśli spojrzeć na poziom ekstentów?

Odpowiedź

Przelicza 10 GB na liczbę ekstentów przy domyślnym rozmiarze 4 MiB (10 GB ÷ 4 MiB ≈ 2560 ekstentów logicznych) i mapuje tyle wolnych ekstentów fizycznych z grupy wolumenów na nowo tworzony wolumen logiczny — bez wymogu, żeby leżały obok siebie ani na jednym konkretnym PV.

4. Dlaczego kolejność `lvremove` → `vgreduce` → `pvremove` jest sztywna, a nie dowolna?

Odpowiedź

Każdy krok wymaga, żeby poprzedni już się wykonał: `vgreduce` odmówi usunięcia PV z grupy, jeśli jakiś LV wciąż trzyma tam swoje ekstenty (stąd najpierw `lvremove` albo `pvmove` danych gdzie indziej), a `pvremove` odmówi zdjęcia nagłówka LVM z urządzenia, które wciąż formalnie należy do jakiejś grupy wolumenów (stąd najpierw `vgreduce`). To zabezpieczenie przed przypadkowym oderwaniem dysku, na którym wciąż leżą aktywne dane.

Powiązane: Co zajmuje miejsce: df, du, lsblk, lsscsi · Powiększanie przestrzeni dyskowej bez restartu · Instalacja Rocky Linux z podziałem na VG i LV

Źródła: `man 8 lvm` · `man 8 pvcreate` · `man 8 vgcreate` · `man 8 lvcreate` · `man 8 dmsetup` · `man 7 lvmthin`

Revision #16
Created 2026-09-17 21:43:03 UTC by Tomasz Konieczny
Updated 2026-10-03 19:08:44 UTC by Tomasz Konieczny