

Student-notities

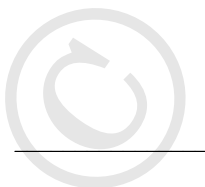
Linux/UNIX systeembeheer en -concepten

Voorbeeld-hoofdstuk

06. Logische disks



Nijmegen



Copyright © AT Computing 2005
Versie: 5d

Student-notities



at computing
The Linux/UNIXperts

Logische disks

Onderwerpen

- Overzicht logische en fysieke disks
- Partitionering
- RAID technologie
- Logical Volume Management (LVM)
- Software RAID
 - geïntegreerd in LVM
 - separate implementatie
- Overzicht processen en logische disks
 - buffer cache
 - Virtual Filesystem Switch (VFS)
 - drie toegangspaden
- Toepassingen

Figuur 1.

Student-notities

De bolletjes, ruitjes en driehoekjes in het plaatje op de sheet geven aan dat er device-namen zijn (in de `/dev-directory`) op verschillende niveau's. We hebben eerder gezien dat device-namen worden gebruikt om device drivers aan te spreken.

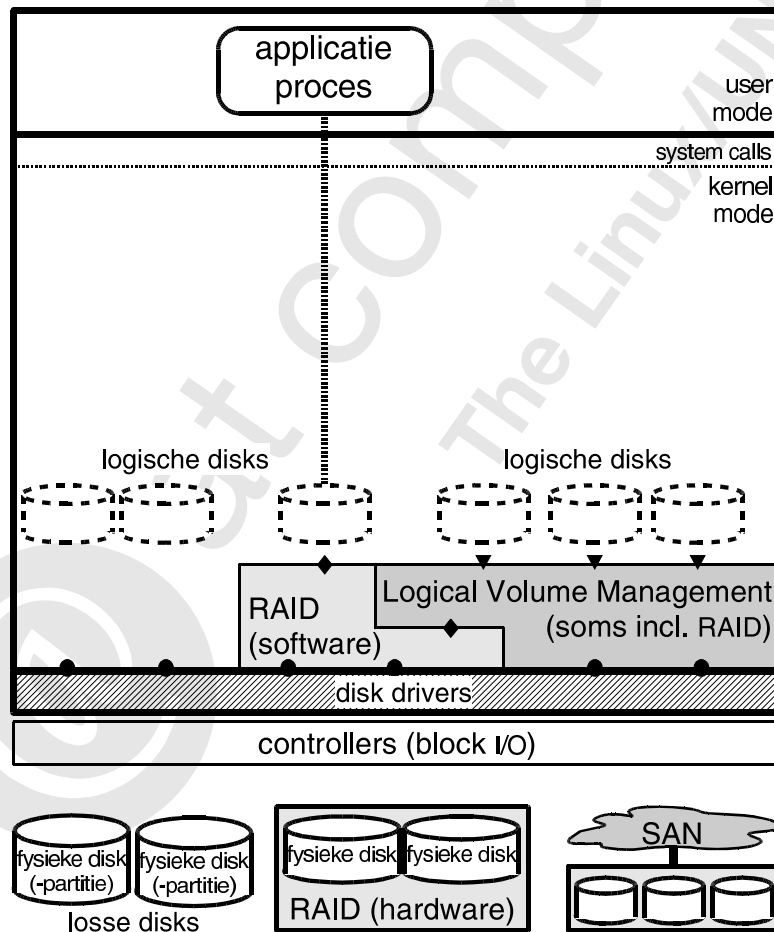
In de praktijk worden RAID- en LVM-lagen in de kernel ook geïmplementeerd als device drivers die via een device-naam aanspreekbaar zijn. Hierarchisch zitten deze drivers op hetzelfde niveau als een echte disk driver. De RAID- en LVM-drivers zijn echter pseudo drivers: ze sturen niet direct hardware aan. De onderkant van deze drivers maakt weer gebruik van de bovenkant van een echte disk device driver; de RAID- of LVM-driver klimt als het ware op de rug van één of meer disk drivers en is zodoende in staat om extra functionaliteit toe te voegen. Je moet dus bijv. een LVM-driver vertellen welke `/dev-entries` hij mag gebruiken om de onderliggende disk(-driver) aan te spreken, of welke `/dev-entries` om de onderliggende RAID-driver aan te spreken; in het laatste geval krijgen we een stapeltje van drie drivers op elkaar.

RAID-levels kunnen hardware- of software-matig worden geïmplementeerd.

Overzicht logische disks

Logische disk

- gedraagt zich als disk naar bovenliggende lagen ("container" voor filesystem, swap, database, ...)
- wordt samengesteld uit (delen van) fysieke disks



Figuur 2.

Student-notities

Een disk driver kan (via /dev-entries) worden aangesproken op het niveau van:

- Disk als geheel.
Een disk kan als geheel (d.w.z. zonder verdere opsplitsing in partities) worden aangesproken. Op deze manier kan de disk als geheel worden aangeboden aan bijvoorbeeld de LVM-laag die allerlei stukjes van zo'n disk mengt met stukjes van andere disks en hieruit weer nieuwe logische disks samenstelt.
Ook als een disk gepartitioneerd wordt gebruikt, zou je bijvoorbeeld een backup kunnen maken van de disk als geheel; zodoende hoeft je niet van iedere partitie apart een backup te maken.
- Disk-partities.
De disk driver "kent" een disk tot op partitie-niveau. Iedere partitie heeft een bepaalde start-sector op disk en bestaat uit een aantal sectoren. De disk-driver beheert de partities op een fysieke disk. De hogere software-lagen in de kernel melden bij ieder verzoek aan de disk-driver om welke partitie het gaat en om welke sector *binnen* die partitie; voor de de hogere lagen begint iedere partitie met sector 0 en lijkt zodoende een disk op zichzelf. Bij ieder verzoek zorgt de disk-driver voor omrekening van sector n op partitie x naar het absolute sectornummer op de fysieke disk ($n + \text{start-sector } x = \text{absolute sector}$).
Iedere gedefinieerde partitie kan worden aangesproken via een eigen device-naam onder de /dev-directory. Het minor nummer levert (na omrekening) de index in de partitietabel.

In plaats van de term *partities* worden fysieke stukken disk ook wel *slices*, *subvolumes*, *sections* of *divisions* genoemd.

Als partitionering wordt ondersteund, heeft de systeembeheerder de vrije keuze bij het aanbrengen van partitiegrenzen op disk. Zodra een disk actief wordt, leest de disk driver de partitie-tabel van die disk en bewaard deze in geheugen.

De volgende commando's zijn beschikbaar per UNIX-variant voor het manipuleren met disk-partities:

<i>Variant</i>	<i>Partitionering</i>
AIX	niet gebruikt (onderverdeling via LVM)
HP-UX	niet gebruikt (onderverdeling via LVM)
Linux	fdisk, parted
Solaris	format fmthard
Tru64	disklabel -e

Partitionering

Fysieke disk opdelen in partities

- disk driver houdt partitie-tabel bij
 - bevat startpositie en grootte van iedere partitie
 - kan worden opgevraagd of gewijzigd met (niet-standaard) commando
- disk driver kan via device-naam worden aangesproken op
 - disk-niveau
 - partitie-niveau

Vaak wordt partitionering overgeslagen

☞ volledige disk wordt aangeboden aan LVM

Figuur 3.

Student-notities

De term *RAID* is de afkorting van Redundant Arrays of Independent Disks, of Redundant Arrays of Inexpensive Disks (de geleerden zijn het hierover niet geheel eens).

De RAID-technologie is onderverdeeld in zeven niveau's. Het gaat hierbij niet om hiërarchische niveau's, maar om een aantal subtechnologieën die ieder afzonderlijk of in combinatie gebruikt kunnen worden. De meeste niveau's, zoals mirroring, zijn bedoeld om de betrouwbaarheid te verhogen. Mn. striping is bedoeld om de toegangssnelheid te vergroten, in de kale vorm zelfs ten koste van de betrouwbaarheid. Met striping en mirroring zijn ook de populairste RAID-levels genoemd.

RAID-0 - Striping

Striping is een techniek waarbij de fysieke blokken van twee of meer disks *in elkaar gevlochten* worden, met als doel een betere spreiding van I/O-opdrachten over meerdere disks (performance-winst).

Als een systeem is uitgerust met meerdere disks, blijkt vaak in de praktijk dat slechts één disk druk bezocht wordt, terwijl andere disks niet of nauwelijks verzoeken krijgen. Dit betekent dat er wachtrijen van verzoeken ontstaan voor de drukke disk, waardoor de responstijden van applicatie-processen te wensen overlaten. Een betere spreiding van de verzoeken zou dan tot een betere performance kunnen leiden. Een handmatige herindeling van veel-gebruikte files over verschillende filesystemen zou dan een mogelijkheid zijn, maar stel dat een gehele database slechts in één file ligt opgeslagen.... Met behulp van striping zou ook dan een goede spreiding kunnen worden bereikt.

Bij striping wordt iedere fysieke disk in zgn. *stripes* verdeeld; dit zijn moten van een bepaalde grootte die de bouwstenen vormen waaruit een logische disk wordt opgebouwd. Een logische disk bestaat uit stripes die alternerend over de fysieke disks zijn genummerd.

Stel dat een logische disk bestaat uit vier fysieke disks en dat de stripe-grootte is gedefinieerd als 32 Kbyte, dan komt de eerste 32K-stripe van de logische disk op de eerste fysieke disk terecht, de tweede 32K-stripe op de tweede fysieke disk, de derde stripe op de derde fysieke disk, de vierde stripe op de vierde fysieke disk, de vijfde stripe weer op de eerste fysieke disk, etc.

Als de logische disk later beschreven wordt met datablokken, worden deze blokken automatisch verspreid over alle fysieke disks waaruit deze logische disk bestaat. Een goede balans wordt met name bereikt als meerdere processen gelijktijdig willekeurige blokken gaan benaderen.

Het grote voordeel van striping is performance-winst. Het nadeel daarentegen is de grotere kwetsbaarheid: als één van de fysieke disks, waaruit een logische disk is opgebouwd rotte plekken vertoont of zelfs geheel crasht, wordt de gehele logische disk onbruikbaar. Om dit nadeel op te heffen, wordt striping vaak gecombineerd met mirroring.

RAID technologie

Redundant Arrays of Independent Disks

- meerdere fysieke disks vormen één of meer logische disks
- doelstelling
 - hoge betrouwbaarheid
 - hoge performance
- verzameling van 7 onafhankelijke technieken (*levels*); slechts enkele daarvan worden frequent gebruikt
- implementatie
 - hardware-matig

voorbeeld: RAID-cabinet met vele fysieke disks wordt aan SCSI-controller gepresenteerd als één of enkele logische disks

- software-matig

voorbeeld: vele losse fysieke disks worden in hogere kernel-lagen gecombineerd tot één of enkele logische disks

Figuur 4.

Student-notities

RAID-0 (striping) staat beschreven op de vorige pagina van deze notities.

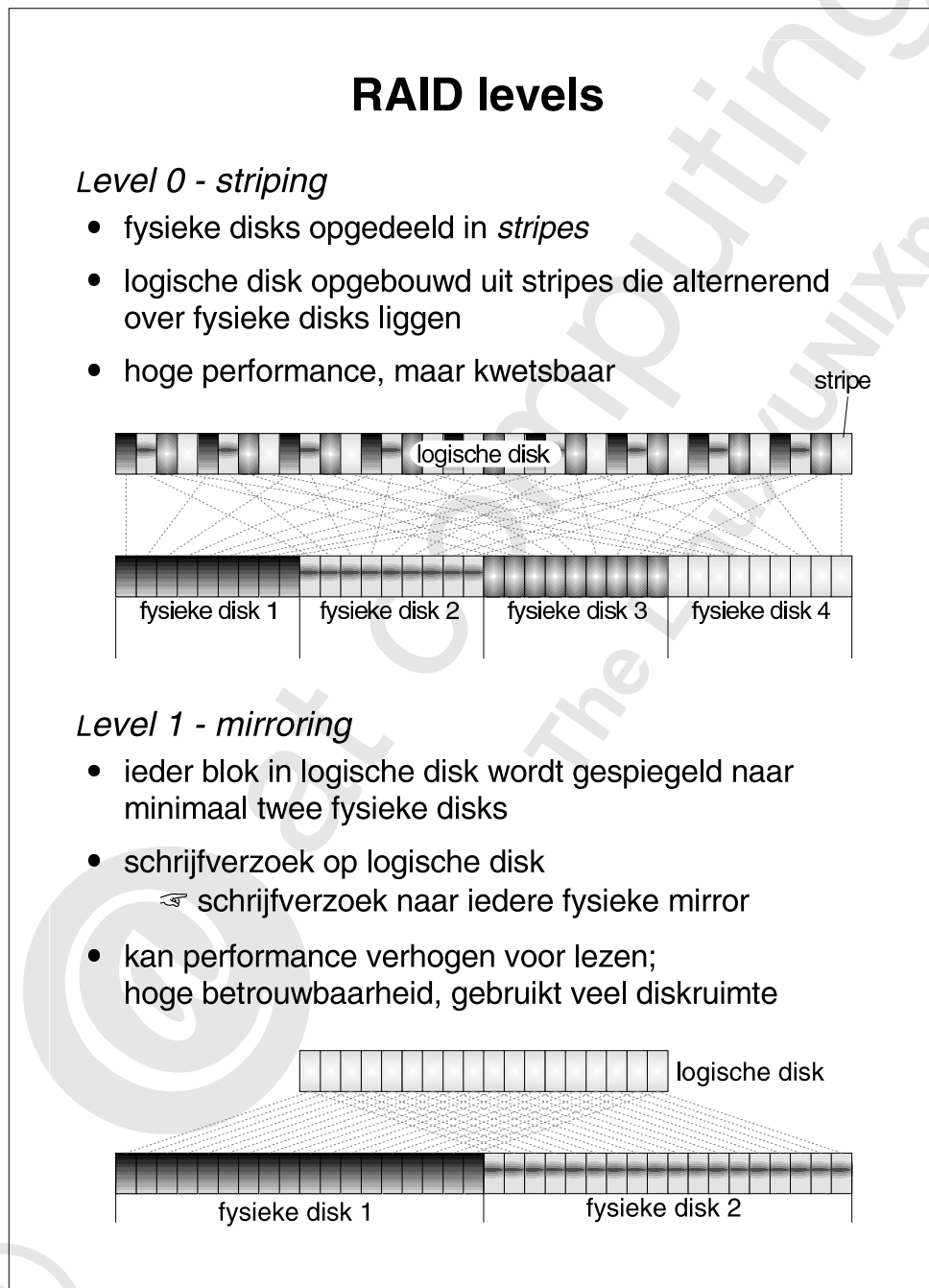
RAID-1 - Mirroring

Mirroring is een techniek waarbij ieder datablok van een logische disk wordt afgebeeld op twee of meer fysieke disks, met als primair doel de betrouwbaarheid te vergroten en het risico van dataverlies bij disk-crashes te verminderen.

Als een blok naar een logische disk met mirroring wordt *geschreven*, wordt naar iedere onderliggende fysieke disk een schrijfverzoek gestuurd (meestal parallel). Het schrijfverzoek naar de logische disk is pas afgerond, als *alle* onderliggende fysieke disks het schrijfverzoek hebben klaargemeld. De duur van een schrijfverzoek voor een proces wordt dus bepaald door de langzaamste (of de drukste) fysieke disk.

Als een blok van een logische disk wordt *gelezen*, wordt het leesverzoek op slechts één van de onderliggende fysieke disks gestart. Dit kan altijd een specifieke disk zijn, maar er kan ook per leesverzoek gekeken worden welk van de onderliggende disks op dat moment het minst druk is (de kortste wachtrij heeft). In het laatste geval kan er met mirroring performance-winst worden geboekt als het lezen betreft.

Het grote voordeel van mirroring is de betrouwbaarheid. Het nadeel is dat er minimaal twee keer zoveel diskruimte benodigd is.



Figuur 5.

Student-notities

Bij RAID-5 wordt uit de corresponderende bits van een aantal data-stripes een pariteit-stripe berekend (via een zgn. "exclusive OR"). Zo'n pariteit-stripe wordt weggeschreven op een andere fysieke disk dan de disks waarop de bijbehorende data-stripes liggen.

Een voorbeeld:

```

1 1 0 1 0 0 0 1 1 1 0 1 0 0 ..... bits op stripe 0
1 0 0 1 1 0 1 0 1 0 0 0 0 1 ..... bits op stripe 1
0 0 0 1 0 1 1 0 0 0 0 1 0 0 ..... bits op stripe 2
----- exclusive OR: een even aantal gezette bits
0 1 0 1 1 1 0 1 0 1 0 0 0 1 ..... bits op pariteit stripe

```

Als stripe 0, 1 of 2 verminkt raakt, kan deze weer worden gereconstrueerd met behulp van de overgebleven stripes en de pariteit-stripe.

Stel dat stripe 2 kapot gaat:

```

1 1 0 1 0 0 0 1 1 1 0 1 0 0 ..... bits op stripe 0
1 0 0 1 1 0 1 0 1 0 0 0 0 1 ..... bits op stripe 1
0 1 0 1 1 1 0 1 0 1 0 0 0 1 ..... bits op pariteit stripe
----- exclusive OR: een even aantal gezette bits
0 0 0 1 0 1 1 0 0 0 0 1 0 0 ..... nieuwe bits voor stripe 2

```

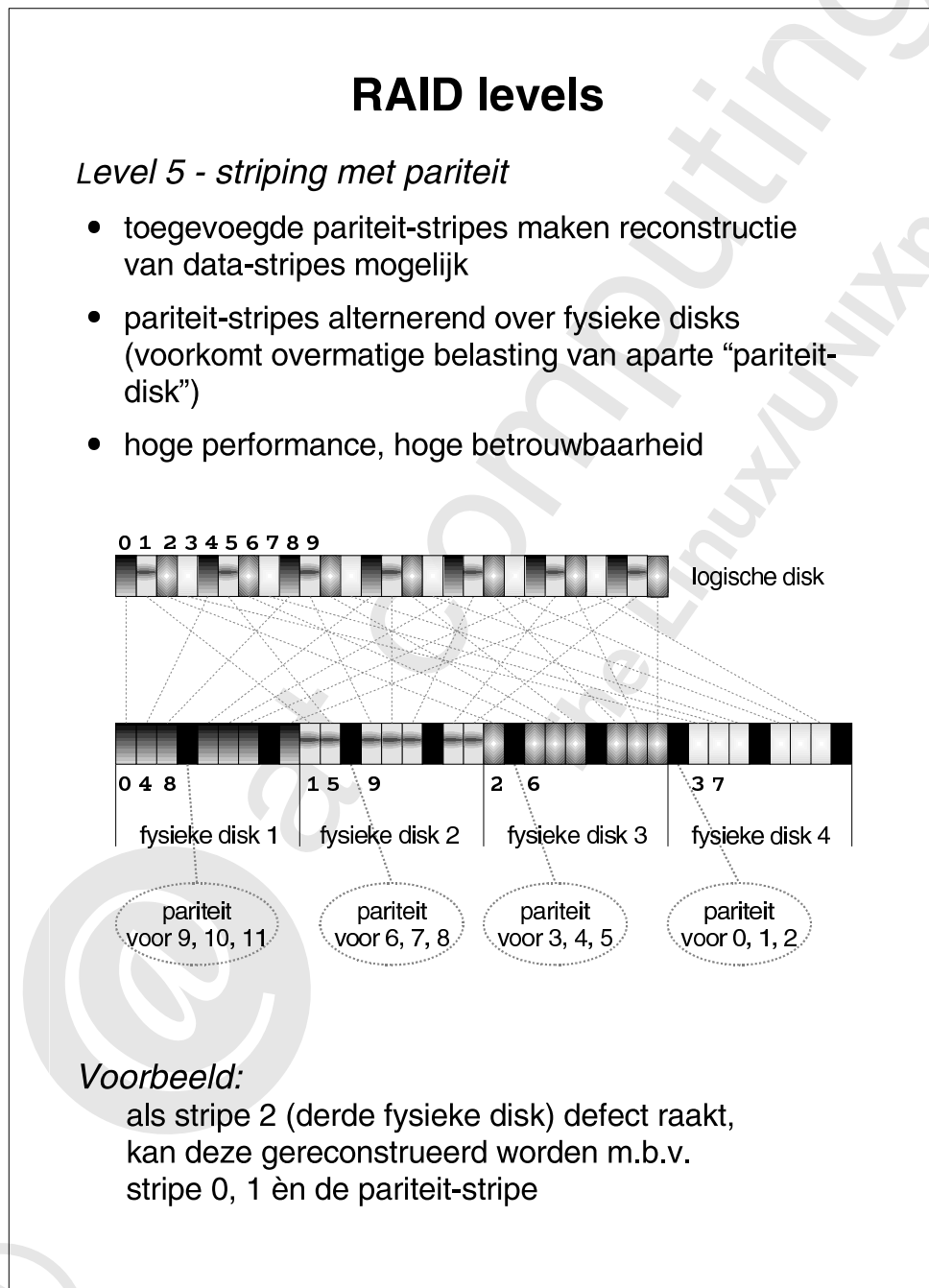
Op dezelfde wijze als waarop één data-stripe kan worden gereconstrueerd, kunnen *alle* data-stripes van een specifieke fysieke disk gereconstrueerd worden als de gehele disk defect raakt.

In het plaatje op de sheet is ook zichtbaar dat de pariteit-stripes alternerend over de fysieke disks zijn verspreid, zodat ook wat dat betreft een gelijkmatige disk-belasting ontstaat¹.

Als één van de data-stripes verandert (bij een schrijfactie), moet de bijbehorende pariteit-stripe opnieuw berekend worden. Zowel de gewijzigde data-stripe alsook de pariteit-stripe moeten dan worden teruggeschreven naar disk. Dat betekent dus dat schrijfacties bij RAID-5 kostbaar zijn (qua performance); in praktijk probeert men het schrijven van de pariteit-stripe zo lang mogelijk uit te stellen d.m.v. caching-technieken. Bij lezen van data wordt de pariteit-stripe uiteraard niet gewijzigd, dus kost dat geen extra performance.

Het voortdurend opnieuw berekenen van de pariteit kost veel processortijd. Dit is een reden dat er weinig software-matige RAID-5 implementaties zijn (pariteit-berekening zou de centrale processor overmatig belasten).

1. Dit is het grote verschil tussen RAID-5 en RAID-3/RAID-4. RAID levels 3 en 4 gebruiken ook striping in combinatie met pariteit, maar alle pariteit-informatie wordt op een aparte daarvoor gereserveerde fysieke disk geschreven (deze disk is meestal de performance-bottleneck). Bij RAID level 3 wordt de pariteit berekend op byte-basis, terwijl dit bij RAID-4 op blok-basis gebeurt.



Figuur 6.

Student-notities

In de praktijk worden de volgende RAID-levels vaak gecombineerd:

RAID 0+1:

Een gestripete logische disk wordt gemirrored naar een andere gestripete logische disk. Deze combinatie levert zowel hoge performance (striping) alsook hoge betrouwbaarheid (mirroring).

RAID 1+0:

Twee of meer gemirrored logische disks worden weer gecombineerd tot één gestripete logische disk. Deze combinatie (ook wel RAID-10 genoemd) levert dezelfde performance en betrouwbaarheid als RAID 0+1, zij het dat reconstructie na een disk-fout wat eenvoudiger is.

Hardware-implementaties van RAID vind je in speciale disk-controllers of in RAID-cabinets. Een RAID-cabinet bestaat uit een grote verzameling fysieke disks (een *disk array*), die zich naar het systeem voordoet als één (of enkele) grote disk-drive(s). Een RAID-cabinet kan op een standaard SCSI-bus worden aangesloten, of vereist een speciale (meegeleverde) host-adapter.

Een disk-array heeft een behoorlijke lading intelligentie en RAM-geheugen aan boord voor performance-optimalisatie (extra processoren, extra cache-geheugen).

Hardware-implementaties van RAID-levels hebben het voordeel dat de centrale processor van het UNIX-systeem niet wordt belast met allerlei reken- en regelwerk. Ook kunnen bepaalde technieken efficiënter worden geïmplementeerd, omdat de logica dichtbij de fysieke disk zit (deze weet bijvoorbeeld de exacte positie van de disk-kop). Anderzijds hangt er vaak een stevig prijskaartje aan.

Bij RAID-1 en RAID-5 is het mogelijk om op voorhand al één of meer "spare-disks" aan te bieden. Deze disks worden niet gebruikt totdat één van de gebruikte disks kapot gaat; de "spare" gaat de kapotte dan automatisch vervangen.

Als de hardware het toelaat, kunnen kapotte disks on-the-fly worden vervangen door nieuwe disks (*hot-swapping*). Zo'n nieuwe disk kan dan weer de functie van spare-disk gaan vervullen, of wordt onmiddellijk weer geladen met de oorspronkelijke data om als actieve disk mee te gaan draaien. Dit alles gebeurt terwijl het UNIX-systeem ondertussen doordraait.... Maar onderschat de belasting van het opnieuw vullen van een vervangen disk niet.



RAID aandachtspunten

Extra benodigde diskruimte per RAID-level

RAID-0: geen extra diskruimte (geen redundantie)

RAID-1: tweevoudige, drievoudige, diskruimte

RAID-5: één fysieke disk in totaal

RAID-1 en RAID-5

- op voorhand spare-disk aanbieden
- on-the-fly kapotte disk vervangen (hot-swap)

Eisen aan disks per mirrorset/stripeset

- aangeboden disk(-partitie)s van gelijke grootte
- in geval van disk-partities:
alle partities op verschillende disks
- bij voorkeur alle disks van hetzelfde type/merk
(met name dezelfde snelheid)
- bij voorkeur alle disks aangesloten op
verschillende controllers

Figuur 7.

Student-notities

Er zijn grofweg vier LVM-implementaties² te vinden bij de vijf UNIX-varianten:

- AIX,
- HP-UX (native) en Linux,
- Tru64, en
- Veritas VM (Solaris en HP-UX).

HP-UX:

Ondersteunt zowel een eigen versie alsook Veritas VM (VxVM).

Bij deze implementaties worden soms verschillende termen gebruikt voor bepaalde entiteiten:

<i>Variant</i>	<i>Groep disks</i>	<i>Logische disk</i>	<i>Eenheid van allocatie</i>
AIX	Volume Group	Logical Volume	Partition (verwarrende term!)
HP-UX	Volume Group	Logical Volume	Extent
Linux	Volume Group	Logical Volume	Extent
Solaris	zie Veritas VM	zie Veritas VM	zie Veritas VM
Tru64 LSM	Disk Group	Volume	Extent
Tru64 AdvFS	Domain	Fileset	Extent
Veritas VM	Disk Group	Volume	Extent

Een *Volume Group (VG)* kan te allen tijde worden uitgebreid met nieuwe Physical Volumes (PV's). Daarmee wordt de pool met vrije diskbrokjes aangevuld. Deze brokjes kunnen weer toegewezen worden aan nieuwe of bestaande Logical Volumes.

Een *Logical Volume (LV)* is uitbreidbaar of krimpbaar als de "content" (bijv. het filesysteem) dit ook ondersteunt.

De grootte van een *Physical Extent (PE)*³ is een eigenschap van de Volume Group, dus geldt voor alle Physical Volumes binnen zo'n Volume Group.

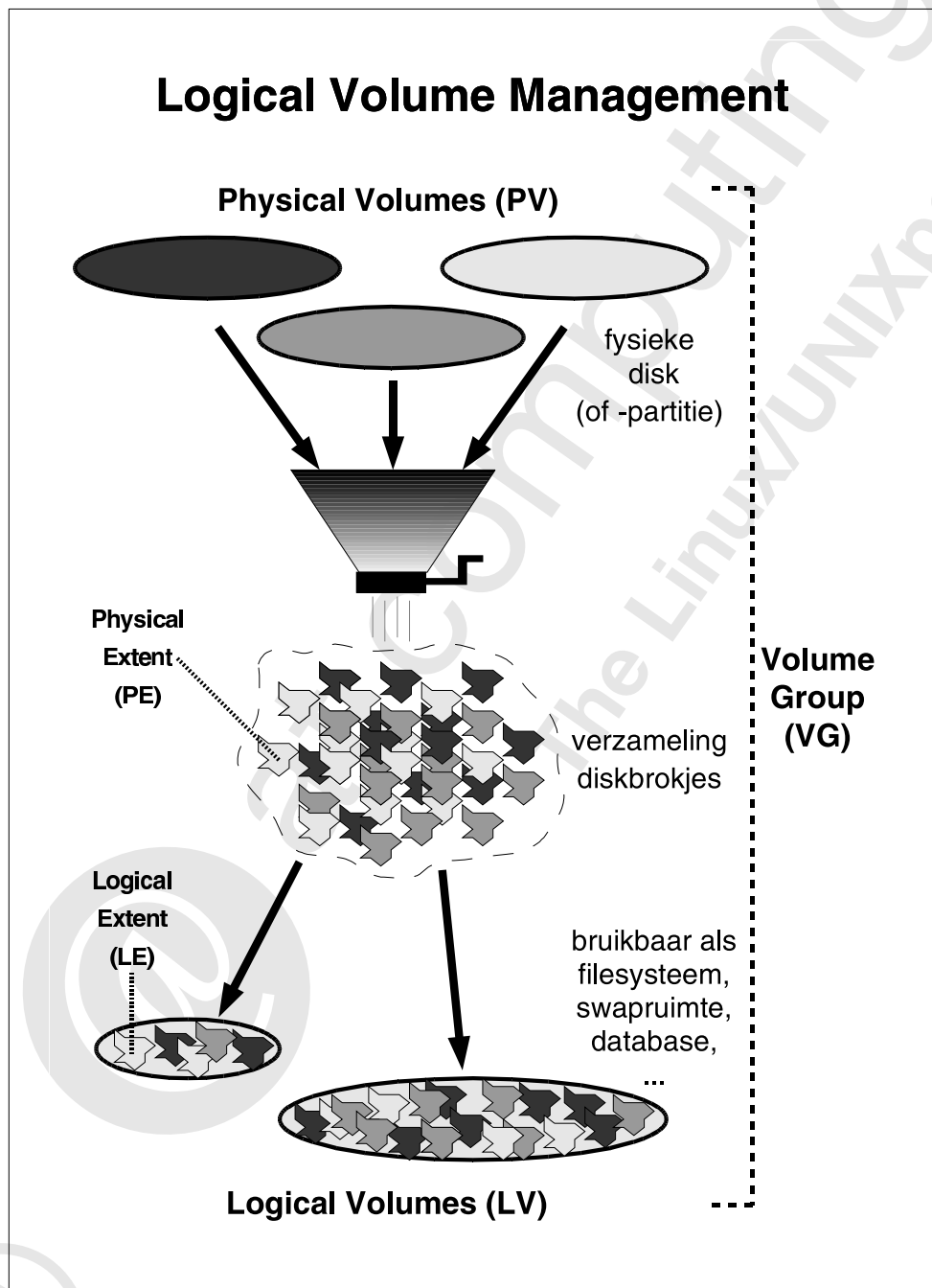
Een Logical Volume wordt opgebouwd uit *Logical Extents (LE's)*; een LE heeft dezelfde grootte als PE. Een LE kan een een-op-een relatie hebben met een PE, maar bij mirroring refereert één LE naar twee of meer PE's (meerdere fysieke kopieën van die LE).

2. *Tru64:*

Hier wordt de term Logical Storage Management (LSM) voor het separate LVM-product gebruikt. Bovendien is een beperkte variant van LVM geïntegreerd in het AdvFS-filesysteem.

3. *AIX:*

Physical Partition (PP)



Figuur 8.

Student-notities

Overzicht van de LVM-commando's per UNIX-variant.

Physical Volumes

<i>Variant</i>	<i>Prepareer PV</i>	<i>Wijzig PV</i>	<i>Verwijder PV</i>	<i>Toon PV</i>
AIX	mkdev	chpv	reducevg	lspv
HP-UX	pvcreate	pvchange	vgreduce	pvdisplay
Linux	pvcreate	pvchange	vgreduce	pvdisplay
Solaris	zie Veritas VM	zie Veritas VM	zie Veritas VM	zie Veritas VM
Tru64 LSM	voldiskadd		voldg rmdisk	volprint -dl
Veritas	vxdiskadd		vxdg rmdisk	vxprint -dl

Volume Groups

<i>Variant</i>	<i>Maak VG</i>	<i>Verwijder VG</i>	<i>Toon VG</i>
AIX	mkvg		lsvg -l vg
HP-UX	vgcreate	vgremove	vgdisplay -v vg
Linux	vgcreate	vgremove	vgdisplay -v
Solaris	zie Veritas VM	zie Veritas VM	zie Veritas VM
Tru64 LSM	voldg init		volprint -l -g vg
Veritas VM	vxdg init		vxprint -l -g vg

Logical Volumes

<i>Variant</i>	<i>Maak LV</i>	<i>Wijzig LV</i>	<i>Verwijder LV</i>	<i>Toon LV</i>
AIX	mklv	chlv	rmlv	lslv
HP-UX	lvcreate	lvchange	lvremove	lvdisplay
Linux	lvcreate	lvchange	lvremove	lvdisplay
Solaris	zie Veritas VM	zie Veritas VM	zie Veritas VM	zie Veritas VM
Tru64 LSM	volassist make	voledit set	voledit ...	volprint -vl
Veritas	vxassist make	vxedit set	vxedit rm	vxprint -vl

Logical Volume Management

Virtualisatie van diskruimte

- + reduceren van onbenutte diskruimte (pool van vrije extents per volume group)
- + filesystemen en databases kunnen groter worden dan enkele fysieke disk
- + logische disks kunnen dynamisch worden uitgebreid (of ingekrompen)
- + striping en/of mirroring vaak geïntegreerd
- fragmentatie ("wat ligt waar?")

Opzetten van LVM (basis)

1. maak fysieke disks of disk-partities geschikt
 - ↳ physical volumes
2. creëer volume group bestaande uit physical volumes
 - ↳ pool van physical extents
3. creëer logical volume(s) binnen volume group

Figuur 9.

Student-notities

Ingebouwde RAID-mogelijkheden van LVM:

<i>Variant</i>	<i>Creëer striped LV</i>	<i>Creëer mirrored LV</i>	<i>Mirror toevoegen</i>
AIX	<code>mklv -u 3 -S 64K</code>	<code>mklv -c 2</code>	<code>mklvcopy lv 2</code>
HP-UX	<code>lvcreate -i 3 -I 64</code>	<code>lvcreate -m 1</code>	<code>lvextend -m 1</code>
Linux	<code>lvcreate -i 3 -I 64</code>		
Solaris	zie Veritas VM	zie Veritas VM	zie Veritas VM
Tru64 LSM	<code>volassist make vol 100mb layout=stripe</code>	<code>volassist make vol 100mb mirror=true</code>	
Veritas VM	<code>vxassist make vol 100mb layout=raid5</code>	<code>vxassist mirror</code>	

Bij LVM-implementaties worden fysieke disks eerst ondergebracht in een Volume Group; uit die voorraad disk-broekjes kunnen later Logical Volumes worden opgebouwd of uitgebreid.

Bij *software RAID*-implementaties worden meerdere fysieke disks “rechtstreeks” gebundeld tot een logische disk, gebruik makend van een bepaald RAID-level.

Voorbeelden van dergelijke implementaties zijn:

Linux Multiple Device

De Multiple Device (md) driver onder Linux biedt mogelijkheden voor software RAID. Naast RAID-0, RAID-1 en RAID-5 wordt ook RAID-4 ondersteund en nog tal van andere varianten op het RAID-thema.

RAID-devices moeten eerst gespecificeerd worden in de file `/etc/raidtab` (zie “`man raidtab`”); daarna zorgt het commando `mkraid` dat deze devices worden geconfigureerd.

Het commando `raidstart` activeert één of meer RAID-devices. Het commando `raidstop` schakelt een RAID-device vervolgens weer uit.

Bij RAID-1 en RAID-5 kunnen “*spare disks*” worden gedefinieerd ter vervanging van disks die later kapot gaan.

Solaris Volume Manager

De Solaris Volume Manager biedt mogelijkheden om meerdere fysieke disks te combineren tot een *volume* (logische disk).

Voordat je volumes kunt maken, moet met commando `metadb` een “state database” (met eventuele replica’s) worden gemaakt op een aparte fysieke disk. Hierin wordt alle administratie vastgelegd.

Met het commando `metainit` kan een volume worden gemaakt bestaande uit één of meer fysieke disks. Met vlaggen wordt aangegeven of de fysieke disks via striping (RAID-0 of RAID-5) of mirroring (RAID-1) moeten worden gecombineerd.

LVM en/of software RAID

Ondersteuning software RAID

- (deels) geïntegreerd in LVM en/of
- in aparte kernel-laag

Overzicht per UNIX-variant

<i>Variant</i>	<i>RAID-0 Striping</i>	<i>RAID-1 Mirroring</i>	<i>RAID-5 Striping+parity</i>
AIX	LVM	LVM	
HP-UX	LVM (native) VxVM	LVM (native) VxVM	VxVM
Linux	LVM md	LVM md	md
Solaris	VxVM volmgr	VxVM volmgr	VxVM volmgr
Tru64	LSM AdvFS (per file)	LSM	LSM

Figuur 10.

Student-notities

Vergroten of verkleinen van logical volumes:

<i>Variant</i>	<i>Vergroot LV</i>	<i>Verklein LV</i>
AIX	extendlv	reducelv
HP-UX	lvextend	lvreduce
Linux	lvextend	lvreduce
Solaris	zie Veritas VM	zie Veritas VM
Tru64 LSM	volassist growto	volassist shrinkto
Veritas VM	vxassist growto	vxassist shrinkto

Voorbeeld LVM onder AIX

Toelichting bij het voorbeeld op de sheet:

- Commando `lsvg` (zonder vlaggen) toont welke volume groups er zijn: volume group `rootvg` wordt altijd gecreëerd tijdens installatie. Commando "`lsvg -p`" toont welke physical volumes onderdeel zijn van de `rootvg` volume group.
- Commando "`lsdev -C -c disk`" geeft een overzicht van de fysieke disks. De disks `hdisk1` en `hdisk2` zijn zojuist gemonteerd.
- Met commando `mkvg` wordt een nieuwe volume group gemaakt, bestaande uit de fysieke disks `hdisk1` en `hdisk2`. De volume group krijgt de naam `atvg` (vlag `-y`) en een partitie-grootte van 8 Mbytes (vlag `-s`) in plaats van de default grootte van 4 Mbytes.
- Commando `varyonvg` zorgt dat de volume group wordt geactiveerd en beschikbaar is voor gebruik.
- Met commando `mklv` wordt een logical volume gemaakt. Vlag `-u` geeft het aantal disk-units aan waarover het LV verspreid moet worden, vlag `-c` geeft het aantal kopieën aan van iedere logical partition (mirroring) en `-S` geeft de grootte aan per stripe (striping).
 - atdevel0: LV van 1 Gbytes, te alloceren op een willekeurige PV.
 - atdevel1: LV van 2 Gbytes met enkelvoudige mirroring over beide PV's (totaal 4 Gbytes).
 - atdevel2: LV van 1600 Mbytes met striping (stripe 16 Kb) verspreid over beide PV's.
- Commando `lsvg` geeft een overzicht van de PV's (vlag `-p`) en de LV's (vlag `-l`) in de `atvg` volume group. Commando `lsvg` met de vlag `-M` geeft een volledige lijst met de afbeelding van logical partitions op physical partitions binnen een volume group. De output van dit commando (sluit aan bij het voorbeeld) vindt u achteraan in dit hoofdstuk.

De logical volumes kunnen nu ingericht worden....

Logical Volume Management

Voorbeeld (AIX)*:

```
# lsvg
rootvg

# lsvg -p rootvg
PV_NAME    PV STATE      TOTAL PPs   FREE PPs
hdisk0     active        537         284

# lsdev -C -c disk
hdisk0 Available 04-C0-00-0,0 SCSI ..
hdisk1 Available 04-C0-00-1,0 SCSI ..
hdisk2 Available 04-C0-00-2,0 SCSI ..

# mkvg -y atvg -s 8 hdisk1 hdisk2
# varyonvg atvg
# mklv -y atdevel0          atvg 128
# mklv -y atdevel1 -u 2 -c 2  atvg 256
# mklv -y atdevel2 -u 2 -S 16k atvg 200

# lsvg -p atvg
PV_NAME    PV STATE      TOTAL PPs   FREE PPs
hdisk1     active        537         53
hdisk2     active        537        181

# lsvg -l atvg
LV_NAME    TYPE          LPs   PPs   PVs
atdevel0   jfs           128   128   1
atdevel1   jfs           256   512   2
atdevel2   jfs           200   200   2

# ls -l /dev/atdevel*
brw-rw---- root system 37,0 .. /dev/atdevel0
brw-rw---- root system 37,1 .. /dev/atdevel1
brw-rw---- root system 37,2 .. /dev/atdevel2
```

Figuur 11.

Student-notities

In het resterende deel van dit hoofdstuk worden de lagen behandeld die zich in de kernel *boven* de disk drivers (logische disks) bevinden, zoals de buffer cache en het filesysteem (VFS).

Verder komt het indelen en dimensioneren van swapruimte ter sprake als praktische toepassing van logische disks.



at computing
The Linux/UNIXperts