ZFS Grundlagen am Beispiel Proxmox VE(Stand Januar 2025)

Der Verfasser des Artikels, Christian-Peter Zengel, hat zum Zeitpunkt des Artikels ca **15 Jahre Erfahrung mit ZFS und Proxmox. Er betreibt aktuell ca 150 Systeme mit Proxmox und ZFS**

Das Einsatzgebiet geht von Standaloneinstallation bis zu ca 10 Hosts im Cluster. **Es ist keine Ceph Expertise vorhanden!**

> Dieser Dokumentation basiert auf diesem online Kurse von cloudistboese.de
>
> ## 21. + 23.01.2025 (13.00 Uhr bis 17.00 Uhr) - ZFS Grundlagen

ZFS ist die perfekte Grundlage für kleine und mittlere Systeme um gegen Ausfälle, Dummheit anderer und Erpressungstrojanern ideal aufgestellt zu sein.
In diesem Kurs erhalten sie Grundkenntnisse für Systeme wie Proxmox, TrueNAS oder ähnliche Systeme.

Der Fokus liegt auf der Technologie und der permanenten Sicherheit im Umgang mit Daten, Linux und dem Terminal.

Themen:

- Begrifflichkeiten und Überblick
- Basisfunktionen von Proxmox und TrueNAS per GUI
- Erklärung und Beispiele Snapshots und Rollbacks
- Praxisbeispiele Replikation
- Prüfung der Replikationen
- Weitere Features die das Leben massiv erleichtern

Nach diesem Kurs bist Du bei Ausfällen und Datenverlust vor dem Schlimmsten geschützt und in Minuten wieder produktiv!

> https://de.wikipedia.org/wiki/ZFS_(Dateisystem)
>
> https://www.proxmox.com/de/
>
> https://cloudistboese.de

So erstellt Proxmox VE seinen ZFS Raid

> zpool create -f -o cachefile=none -o ashift=12 rpool mirror disk1 disk2
>
> Destruktives Anlegen (-f), kein automatisches Importieren beim Start (-o cachefile=none)
> -o ashift=12 ist für 4k Festplatten, ashit=9 für 512e Disks, wobei man mit 12 nichts verkehrt macht
> rpool ist der Name vom künftigen Pool, mirror der Raidlevel und die Disks wurden zweifelsfrei über Ihre ID definiert
>
> Optionen für Raidlevel wären noch: ohne als Stripe, raidz, raidz2 oder raidz3, also Raid Level 5-7

Dokument 142
Bearbeitet

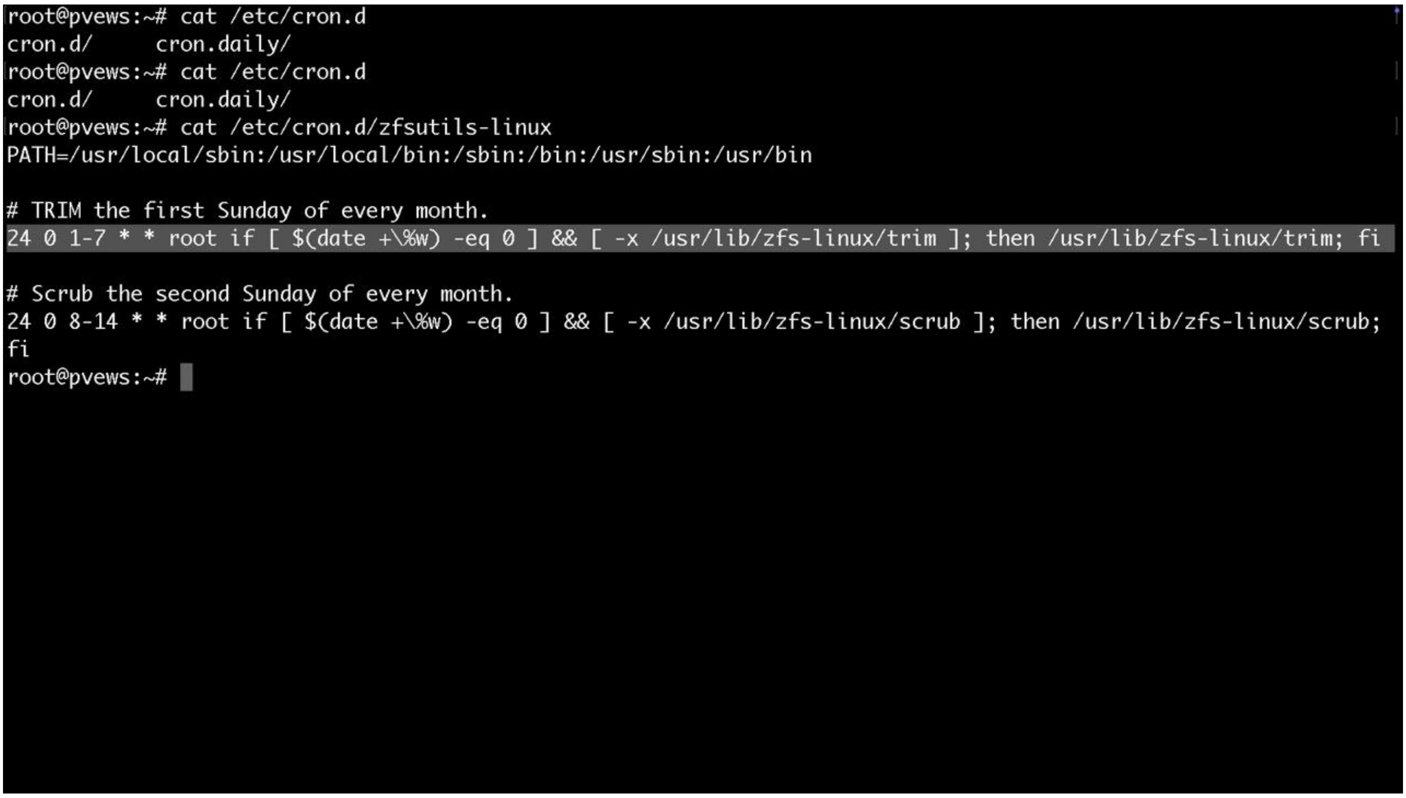
```
zpool create #erstellt raid
-f force #overwrite
-o cachefile=none #das teil merkt sich die letzte importsituation
-o ashift=12 #es gab früher 512e Disk, inzwischen ist 4k Standard
rpool mirror /dev/disk/by-id/ata-CT1000BX500SSD1_2348E88597D2-part3 /dev/disk/by-id/ata-CT1000BX500SSD1_2348E885985B-part3


ashift=9  =  512b
ashift=12 = 4096b

RaidZ(5) 4+4+4+R
Raid(1)  4+R

Virtuelles Volume war Standardmässig 8k
Virtuelle Volumes mit mindestens 16k, genauso gilt das auch für Dateien (Standard 128k)
```

Buchstaben: 501 · Auswahl: 6 - 1 · Worte: 79

> Die ersten Zpool Manöver und die endlose Aktionsliste sieht man mit
>
> zpool history

```
        NAME                                         STATE     READ WRITE CKSUM
        rpool                                        ONLINE       0     0     0
          mirror-0                                   ONLINE       0     0     0
            ata-CT1000BX500SSD1_2348E88597D2-part3   ONLINE       0     0     0
            ata-CT1000BX500SSD1_2348E885985B-part3   ONLINE       0     0     0

errors: No known data errors
root@pvews:~# zpool history
History for 'rpool':
2025-01-20.14:52:00 zpool create -f -o cachefile=none -o ashift=12 rpool mirror /dev/disk/by-id/ata-CT1000BX500SSD1_2348E88597D2-part3 /dev/disk/by-id/ata-CT1000BX500SSD1_2348E885985B-part3
2025-01-20.14:52:00 zfs create rpool/ROOT
2025-01-20.14:52:00 zfs create rpool/data
2025-01-20.14:52:00 zfs create rpool/ROOT/pve-1
2025-01-20.14:52:00 zfs set atime=on relatime=on rpool
2025-01-20.14:52:00 zfs set compression=on rpool
2025-01-20.14:52:00 zfs set sync=disabled rpool
2025-01-20.14:55:27 zfs set sync=standard rpool
2025-01-20.14:55:27 zfs set mountpoint=/ rpool/ROOT/pve-1
2025-01-20.14:55:27 zpool set bootfs=rpool/ROOT/pve-1 rpool
2025-01-20.14:55:27 zpool export rpool
2025-01-20.15:00:36 zpool import -N rpool
2025-01-20.15:44:29 zpool import -N rpool
2025-01-20.16:49:35 zpool import -N rpool
2025-01-20.16:50:42 zpool clear rpool
2025-01-21.10:16:45 zpool import -N rpool
2025-01-21.13:19:38 zpool offline rpool ata-CT1000BX500SSD1_2348E885985B-part3
2025-01-21.13:21:48 zpool online rpool ata-CT1000BX500SSD1_2348E885985B-part3
2025-01-21.13:22:30 zpool clear rpool
2025-01-21.13:28:54 zpool set autotrim=on rpool

root@pvews:~#
```

> Manuelles Trimmen der SSDs um gelöscht Blöcke schner überschreiben zu können findet man unter /etc/cron.d
> Ebenso wird nach diesem Zeitplan der sog. **zpool scrub** durchgeführt, der die Konsistenz der Redundanz prüft und ggf. korrigiert. Gefundene Fehler findet man dann mit **zpool status**



```
root@pvews:~# cat /etc/cron.d
cron.d/     cron.daily/
root@pvews:~# cat /etc/cron.d
cron.d/     cron.daily/
root@pvews:~# cat /etc/cron.d/zfsutils-linux
PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin

# TRIM the first Sunday of every month.
24 0 1-7 * * root if [ $(date +\%w) -eq 0 ] && [ -x /usr/lib/zfs-linux/trim ]; then /usr/lib/zfs-linux/trim; fi

# Scrub the second Sunday of every month.
24 0 8-14 * * root if [ $(date +\%w) -eq 0 ] && [ -x /usr/lib/zfs-linux/scrub ]; then /usr/lib/zfs-linux/scrub;
fi
root@pvews:~#
```

> Zum identifizieren der Platten bieten sich neben der PVE GUI noch folgende Befehle an. Es wird dringend empfohlen bereits genutzte Platten via PVE GUI zu wipen
>
> > dmesg -Tw (live)
> > lsblk
> > ls -althr /dev/disk/by-id

```
root@pvews:~# lsblk
NAME                                    MAJ:MIN RM   SIZE RO TYPE MOUNTPOINT
sda                                       8:0    0   2.7T  0 disk
├─sda1                                    8:1    0   2.7T  0 part
└─sda9                                    8:9    0     8M  0 part
sdb                                       8:16   0   2.7T  0 disk
├─sdb1                                    8:17   0   2.7T  0 part
└─sdb9                                    8:25   0     8M  0 part
sdc                                       8:32   0   2.7T  0 disk
├─sdc1                                    8:33   0   2.7T  0 part
└─sdc9                                    8:41   0     8M  0 part
sdd                                       8:48   0   2.7T  0 disk
├─sdd1                                    8:49   0   2.7T  0 part
└─sdd9                                    8:57   0     8M  0 part
sde                                       8:64   0   2.7T  0 disk
├─sde1                                    8:65   0   2.7T  0 part
└─sde9                                    8:73   0     8M  0 part
sdf                                       8:80   0   2.7T  0 disk
├─sdf1                                    8:81   0   2.7T  0 part
└─sdf9                                    8:89   0     8M  0 part
sdg                                       8:96   0   2.7T  0 disk
├─sdg1                                    8:97   0   2.7T  0 part
└─sdg9                                    8:105  0     8M  0 part
sdh                                       8:112  0 931.5G  0 disk
├─sdh1                                    8:113  0  1007K  0 part
├─sdh2                                    8:114  0     1G  0 part
└─sdh3                                    8:115  0 930.5G  0 part
sdi                                       8:128  0 931.5G  0 disk
```

Erstellen verschiedener Raidlevel und Ihrer Vorzüge

> Mit ls -althr /dev/disk/by-id haben wir folgende Festplatten anhand ihrer aufgedruckten WWN Kennung identifiziert. Es ist dringend empfohlen bei HBAs sich die Slots und die Seriennummern beim Einbau zu notieren und in die Dokumentation aufzunehmen

wwn-0x5000cca01a72ded4 wwn-0x5000cca01a7b1e2c wwn-0x5000cca01a83a61c wwn-0x5000cca01a832d24 wwn-0x5000cca01a83331c wwn-0x5000cca01a7495b8 wwn-0x5000cca01a8417fc

Stripe, also Raid0

zpool create -f test wwn-0x5000cca01a72ded4 wwn-0x5000cca01a7b1e2c wwn-0x5000cca01a83a61c wwn-0x5000cca01a832d24 wwn-0x5000cca01a83331c wwn-0x5000cca01a7495b8

```
test                    ONLINE
 wwn-0x5000cca01a72ded4  ONLINE
 wwn-0x5000cca01a7b1e2c  ONLINE
 wwn-0x5000cca01a83a61c  ONLINE
 wwn-0x5000cca01a832d24  ONLINE
 wwn-0x5000cca01a83331c  ONLINE
```

wwn-0x5000cca01a7495b8 ONLINE
wwn-0x5000cca01a8417fc ONLINE

Mirror, also Raid1

```
zpool create -f test mirror wwn-0x5000cca01a72ded4 wwn-0x5000cca01a7b1e2c
```

```
test
 mirror
   wwn-0x5000cca01a72ded4
   wwn-0x5000cca01a7b1e2c
```

Striped Mirror, also Raid10

```
zpool create -f test mirror wwn-0x5000cca01a72ded4 wwn-0x5000cca01a7b1e2c mirror
 wwn-0x5000cca01a83a61c  wwn-0x5000cca01a832d24  mirror wwn-0x5000cca01a83331c
 wwn-0x5000cca01a7495b8
```

```
test
 mirror
   wwn-0x5000cca01a72ded4
   wwn-0x5000cca01a7b1e2c
 mirror
   wwn-0x5000cca01a83a61c
   wwn-0x5000cca01a832d24
 mirror
   wwn-0x5000cca01a83331c
   wwn-0x5000cca01a7495b8
```

RaidZ, also Raid 5, Nettoplatz x-1

```
zpool create test raidz wwn-0x5000cca01a72ded4 wwn-0x5000cca01a7b1e2c  wwn-
0x5000cca01a83a61c wwn-0x5000cca01a832d24  wwn-0x5000cca01a83331c  wwn-
0x5000cca01a7495b8  wwn-0x5000cca01a8417fc
```

```
test                        ONLINE       0     0     0
 raidz1-0                   ONLINE       0     0     0
   wwn-0x5000cca01a72ded4  ONLINE       0     0     0
   wwn-0x5000cca01a7b1e2c  ONLINE       0     0     0
   wwn-0x5000cca01a83a61c  ONLINE       0     0     0
   wwn-0x5000cca01a832d24  ONLINE       0     0     0
   wwn-0x5000cca01a83331c  ONLINE       0     0     0
   wwn-0x5000cca01a7495b8  ONLINE       0     0     0
   wwn-0x5000cca01a8417fc  ONLINE       0     0     0
```

RaidZ2, also Raid 6, Nettoplatz x-2

```
zpool create test raidz2 wwn-0x5000cca01a72ded4 wwn-0x5000cca01a7b1e2c  wwn-0x5000cca01a83a61c wwn-0x5000cca01a832d24  wwn-0x5000cca01a83331c  wwn-0x5000cca01a7495b8  wwn-0x5000cca01a8417fc
```

```
 raidz2-0                  ONLINE       0     0     0
   wwn-0x5000cca01a72ded4  ONLINE       0     0     0
   wwn-0x5000cca01a7b1e2c  ONLINE       0     0     0
   wwn-0x5000cca01a83a61c  ONLINE       0     0     0
   wwn-0x5000cca01a832d24  ONLINE       0     0     0
   wwn-0x5000cca01a83331c  ONLINE       0     0     0
   wwn-0x5000cca01a7495b8  ONLINE       0     0     0
   wwn-0x5000cca01a8417fc  ONLINE       0     0     0
```

RaidZ-0, also Raid 5-0, Nettoplatz x-2, in diesem Beispiel

```
zpool create test raidz wwn-0x5000cca01a72ded4 wwn-0x5000cca01a7b1e2c  wwn-0x5000cca01a83a61c raidz wwn-0x5000cca01a832d24  wwn-0x5000cca01a83331c  wwn-0x5000cca01a7495b8
```

```
test                       ONLINE       0     0     0
 raidz1-0                  ONLINE       0     0     0
   wwn-0x5000cca01a72ded4  ONLINE       0     0     0
   wwn-0x5000cca01a7b1e2c  ONLINE       0     0     0
   wwn-0x5000cca01a83a61c  ONLINE       0     0     0
 raidz1-1                  ONLINE       0     0     0
   wwn-0x5000cca01a832d24  ONLINE       0     0     0
   wwn-0x5000cca01a83331c  ONLINE       0     0     0
   wwn-0x5000cca01a7495b8  ONLINE       0     0     0
   wwn-0x5000cca01a8417fc  ONLINE       0     0     0
```

> oder mit Spare, wenn Du davon nicht booten musst

```
zpool create test raidz wwn-0x5000cca01a72ded4 wwn-0x5000cca01a7b1e2c  wwn-0x5000cca01a83a61c raidz wwn-0x5000cca01a832d24  wwn-0x5000cca01a83331c  wwn-0x5000cca01a7495b8  spare wwn-0x5000cca01a8417fc
```

Erweiterung eines RaidZ zum RaidZ-0

```
test                          ONLINE       0     0     0
 raidz1-0                     ONLINE       0     0     0
   wwn-0x5000cca01a72ded4  ONLINE       0     0     0
   wwn-0x5000cca01a7b1e2c  ONLINE       0     0     0
   wwn-0x5000cca01a83a61c  ONLINE       0     0     0
```

> zpool add -n test raidz wwn-0x5000cca01a832d24 wwn-0x5000cca01a83331c wwn-0x5000cca01a7495b8

```
test
 raidz1-0
   wwn-0x5000cca01a72ded4
   wwn-0x5000cca01a7b1e2c
   wwn-0x5000cca01a83a61c
 raidz1
   wwn-0x5000cca01a832d24
   wwn-0x5000cca01a83331c
   wwn-0x5000cca01a7495b8
```

Erweiterung eines Raid durch Ersetzen mit größeren Platten

> Die Platten müssen die Gleiche Geometrie hanben, also 4k zu 4k oder 512e zu 512e
>
> Bei Raid 1 und 10 reicht es zwei Disks zu tauschen, bei RaidZx müssen alle Disks getauscht werden
>
> Replace Vorgänge können auch laufen ohne die Redundanz zu brechen, wenn weitere Slots frei sind
>
> Bei Erweiterung durch Austausch von großen und vor allem älteren Raid 10 prüfen ob in einem Mirror ggf. neue Platten getauscht wurden, diese sollten weiter im Betrieb bleiben, während ältere Paare getauscht werden können

rpool-hdd
 mirror-0
  2024-8T (vorher Disk aus 2014)
  2024-8T (vorher Disk aus 2014)
 mirror-1
  2014-2T
  2021-2T
 mirror-2
  2022-2T
  2014-2T

Caches und Logdevices

> Vorsichtig formuloiert, Lese- und Schreibcache
>
> First Level Cache, sog. ARC kommt aus dem RAM und bekommt idealerweise ca. 1GB für 1TB Nettodaten
>
> Second Level Cache wird als Cachedevice als Partition auf einer schnelleren Disk als der Pool hat bereitgestellt, z. B. am idealsten mit NVMe

```
zpool add test -n cache sdl
```

test
 raidz2-0
   wwn-0x5000cca01a72ded4
   wwn-0x5000cca01a7b1e2c
   wwn-0x5000cca01a83a61c
   wwn-0x5000cca01a832d24
   wwn-0x5000cca01a83331c
   wwn-0x5000cca01a7495b8
   wwn-0x5000cca01a8417fc
cache
 sdl

> Logdevice (Writecache muss gespiegelt werden)
> Damit der Log auch genutzt wird muss man noch mit zfs set sync den cache aktivieren

```
zpool add test -n log mirror sdl
```

```
zfs set sync=always test
```

test
 raidz2-0
   wwn-0x5000cca01a72ded4
   wwn-0x5000cca01a7b1e2c
   wwn-0x5000cca01a83a61c
   wwn-0x5000cca01a832d24
   wwn-0x5000cca01a83331c
   wwn-0x5000cca01a7495b8
   wwn-0x5000cca01a8417fc

logs
 mirror
   sdl
   sdk

> **Arc wird mit "arcstat 1" ausgelesen, er wird bei der Installation festgelegt oder später unter /etc/modprobe.d/zfs.conf geändert.**
> **update-initramfs -u macht das dann persistent und aktiviert die Änderungen beim nächsten reboot.**
> **Zur Laufzeit ändert man den Firstlevelcahce mit**
> **Cache- und Logdevices mit zpool iostat 1**

```
echo 2147483648 > /sys/module/zfs/parameters/zfs_arc_max
echo 1073741824 > /sys/module/zfs/parameters/zfs_arc_min
free -h && sync && echo 3 > /proc/sys/vm/drop_caches && free -h
```

```
arcstat 1
```

```
15:15:07     3     0      0     0    0      0    0      0    0  1.4G   33G    87G
15:15:08     8     0      0     0    0      0    0      0    0  1.4G   33G    87G
15:15:09     0     0      0     0    0      0    0      0    0  1.4G   33G    87G
15:15:10   157     0      0     0    0      0    0      0    0  1.4G   33G    87G
15:15:11    15     0      0     0    0      0    0      0    0  1.4G   33G    87G
15:15:12     5     0      0     0    0      0    0      0    0  1.4G   33G    87G
15:15:13    24     0      0     0    0      0    0      0    0  1.4G   33G    87G
15:15:14     0     0      0     0    0      0    0      0    0  1.4G   33G    87G
15:15:15   166     0      0     0    0      0    0      0    0  1.4G   33G    87G
15:15:16    15     0      0     0    0      0    0      0    0  1.4G   33G    87G
15:15:17     3     0      0     0    0      0    0      0    0  1.4G   33G    87G
15:15:18     0     0      0     0    0      0    0      0    0  1.4G   33G    87G
15:15:19     0     0      0     0    0      0    0      0    0  1.4G   33G    87G
15:15:20   152     0      0     0    0      0    0      0    0  1.4G   33G    87G
15:15:21    15     0      0     0    0      0    0      0    0  1.4G   33G    87G
15:15:22     3     0      0     0    0      0    0      0    0  1.4G   33G    87G
15:15:23    24     0      0     0    0      0    0      0    0  1.4G   33G    87G
15:15:24     2     0      0     0    0      0    0      0    0  1.4G   33G    87G
15:15:25    98     0      0     0    0      0    0      0    0  1.4G   33G    87G
15:15:26    10     0      0     0    0      0    0      0    0  1.4G   33G    87G
15:15:27     3     0      0     0    0      0    0      0    0  1.4G   33G    87G
15:15:28     2     0      0     0    0      0    0      0    0  1.4G   33G    87G
15:15:29     0     0      0     0    0      0    0      0    0  1.4G   33G    87G
15:15:30   167     0      0     0    0      0    0      0    0  1.4G   33G    87G
    time  read  miss  miss%  dmis  dm%  pmis  pm%  mmis  mm%  size     c  avail
15:15:31    15     0      0     0    0      0    0      0    0  1.4G   33G    87G
15:15:32     5     0      0     0    0      0    0      0    0  1.4G   33G    87G
^C
root@pvews:~# #arc cat 1GB pro Netto TB
root@pvews:~# 
```

Test der Leistung eines Pools

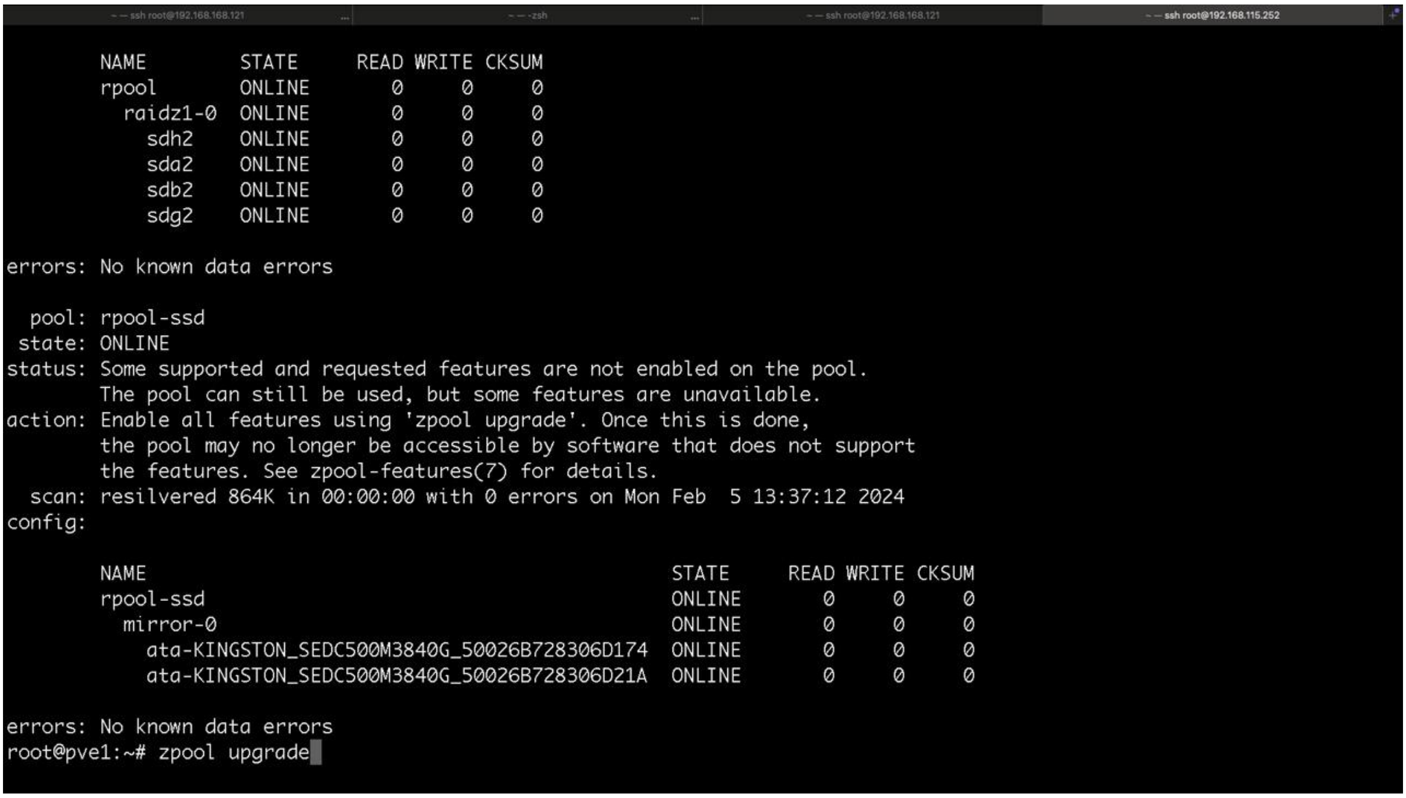
```
zfs create -o compression=off test/speed
cd /test/speed
dd if=/dev/zero of=dd.tmp bs=256k count=16384 status=progress
dd if=/dev/zero of=dd.tmp bs=2M count=16384 status=progress
```

```
root@pvews:~# zpool destroy test
root@pvews:~# zpool create -f test mirror wwn-0x5000cca01a72ded4 wwn-0x5000cca01a7b1e2c mirror  wwn-0x5000cca01a83a61c
 wwn-0x5000cca01a832d24  mirror wwn-0x5000cca01a83331c  wwn-0x5000cca01a7495b8
root@pvews:~# zpool add test  log mirror sdl sdk
root@pvews:~# zpool create -f test mirror wwn-0x5000cca01a72ded4 wwn-0x5000cca01a7b1e2c mirror  wwn-0x5000cca01a83a61c
 wwn-0x5000cca01a832d24  mirror wwn-0x5000cca01a83331c  wwn-0x5000cca01a7495b8^C
root@pvews:~# cd /test/speed
-bash: cd: /test/speed: No such file or directory
root@pvews:~# zfs list
NAME               USED  AVAIL     REFER  MOUNTPOINT
rpool             1.54G   898G      104K  /rpool
rpool/ROOT        1.54G   898G       96K  /rpool/ROOT
rpool/ROOT/pve-1  1.54G   898G     1.28G  /
rpool/data          96K   898G       96K  /rpool/data
test               164K  8.03T       24K  /test
root@pvews:~# zpool add test  log mirror sdl sdk^C
(failed reverse-i-search)`create tan': zpool ^Ceate test raidz2 wwn-0x5000cca01a72ded4 wwn-0x5000cca01a7b1e2c  wwn-0x50
00cca01a83a61c wwn-0x5000cca01a832d24  wwn-0x5000cca01a83331c  wwn-0x5000cca01a7495b8  wwn-0x5000cca01a8417fc
root@pvews:~# zfs create -o compression=off test/speed
cd /test/speed
#dd if=/dev/zero of=dd.tmp bs=256k count=16384 status=progress
dd if=/dev/zero of=dd.tmp bs=2M count=16384 status=progress
9151971328 bytes (9.2 GB, 8.5 GiB) copied, 15 s, 610 MB/s^C
4430+0 records in
4430+0 records out
9290383360 bytes (9.3 GB, 8.7 GiB) copied, 15.3659 s, 605 MB/s

root@pvews:/test/speed# zfs set sync=always test
root@pvews:/test/speed# dd if=/dev/zero of=dd.tmp bs=2M count=16384 status=progress
```

```
    wwn-0x5000cca01a832d24      -      -      0      0      0      0
  mirror-2                   583M  2.72T      0      0      0      0
    wwn-0x5000cca01a83331c      -      -      0      0      0      0
    wwn-0x5000cca01a7495b8      -      -      0      0      0      0
logs                            -      -      -      -      -      -
  mirror-3                   614M   443G      0    597      0   120M
    sdl                         -      -      0    298      0  60.1M
    sdk                         -      -      0    298      0  60.1M
--------------------------  -----  -----  -----  -----  -----  -----
                              capacity     operations     bandwidth
pool                        alloc   free   read  write   read  write
--------------------------  -----  -----  -----  -----  -----  -----
test                        1.69G  8.15T      0    597      0   120M
  mirror-0                   573M  2.72T      0      0      0      0
    wwn-0x5000cca01a72ded4      -      -      0      0      0      0
    wwn-0x5000cca01a7b1e2c      -      -      0      0      0      0
  mirror-1                   579M  2.72T      0      0      0      0
    wwn-0x5000cca01a83a61c      -      -      0      0      0      0
    wwn-0x5000cca01a832d24      -      -      0      0      0      0
  mirror-2                   583M  2.72T      0      0      0      0
    wwn-0x5000cca01a83331c      -      -      0      0      0      0
    wwn-0x5000cca01a7495b8      -      -      0      0      0      0
logs                            -      -      -      -      -      -
  mirror-3                   614M   443G      0    596      0   121M
    sdl                         -      -      0    296      0  59.8M
    sdk                         -      -      0    299      0  61.1M
--------------------------  -----  -----  -----  -----  -----  -----
^C
root@pvews:~# #perfekt für onboard nvme
```

Erweiterung der Funktionen des Pools

> zpool status meldet neue Funktionen im ZFS und kann leicht und schnell mit
>
> > zpool upgrade -a
>
> erledigt werden
>
> Jedoch würde ich in jedem Fall empfehlen zu prüfen ob meine Notfall ISO diese Funktionen bereits unterstützt!



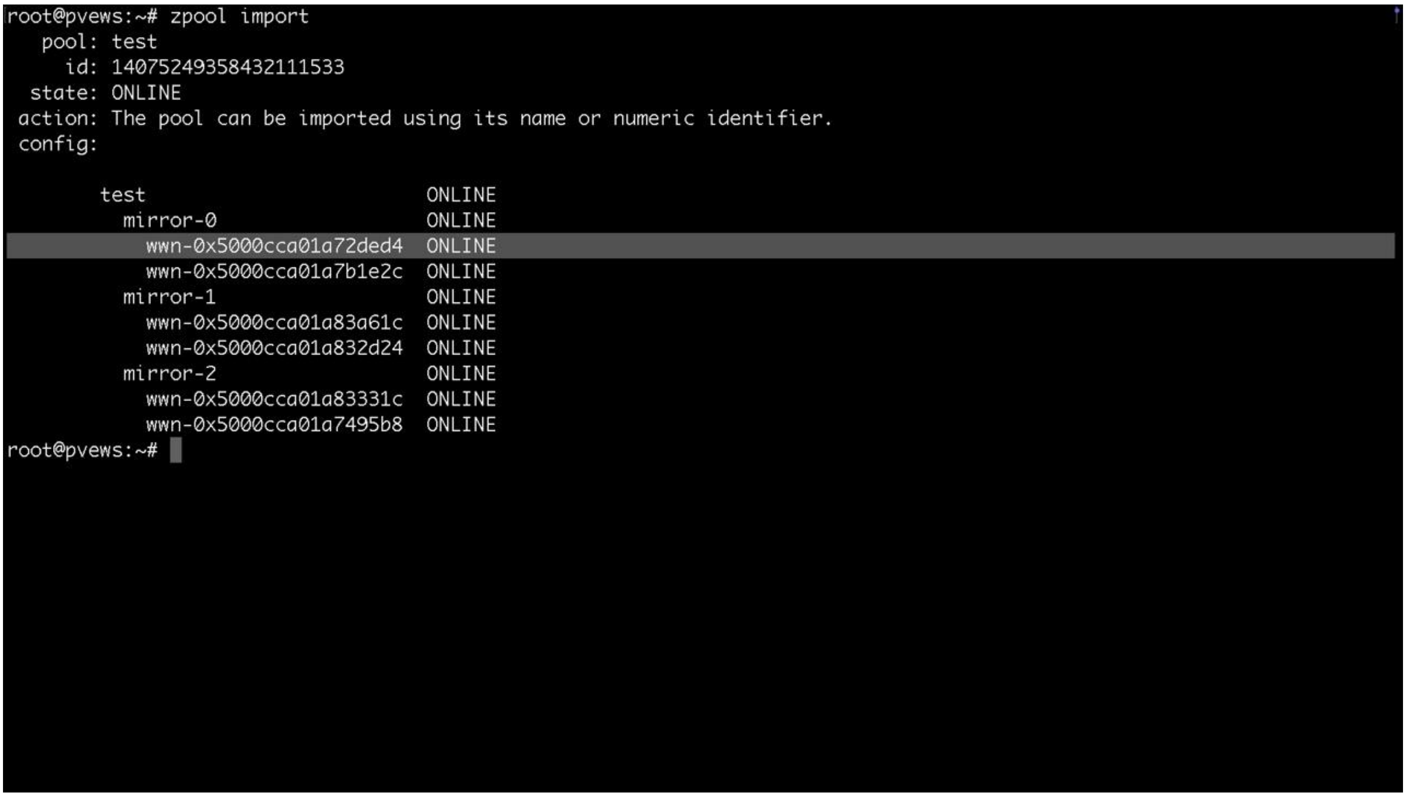
```
Some supported features are not enabled on the following pools. Once a
feature is enabled the pool may become incompatible with software
that does not support the feature. See zpool-features(7) for details.

Note that the pool 'compatibility' feature can be used to inhibit
feature upgrades.

POOL  FEATURE
---------------
backup
      zilsaxattr
      head_errlog
      blake3
      block_cloning
      vdev_zaps_v2
rpool
      zilsaxattr
      head_errlog
      blake3
      block_cloning
      vdev_zaps_v2
rpool-ssd
      zilsaxattr
      head_errlog
      blake3
      block_cloning
      vdev_zaps_v2

root@pve1:~#
```

> Für das Auslesen der Parameter von Pools, Volumes, Datasets und Snapshots gibt es den Parameter **get**

```
root@pvews:~# zpool get all rpool
NAME   PROPERTY                       VALUE                          SOURCE
rpool  size                           928G                           -
rpool  capacity                       0%                             -
rpool  altroot                        -                              default
rpool  health                         ONLINE                         -
rpool  guid                           15399376736129336856           -
rpool  version                        -                              default
rpool  bootfs                         rpool/ROOT/pve-1               local
rpool  delegation                     on                             default
rpool  autoreplace                    off                            default
rpool  cachefile                      -                              default
rpool  failmode                       wait                           default
rpool  listsnapshots                  off                            default
rpool  autoexpand                     off                            default
rpool  dedupratio                     1.00x                          -
rpool  free                           926G                           -
rpool  allocated                      1.55G                          -
rpool  readonly                       off                            -
rpool  ashift                         12                             local
rpool  comment                        -                              default
rpool  expandsize                     -                              -
rpool  freeing                        0                              -
rpool  fragmentation                  0%                             -
rpool  leaked                         0                              -
rpool  multihost                      off                            default
rpool  checkpoint                     -                              -
rpool  load_guid                      16411429224255305155           -
rpool  autotrim                       on                             local
rpool  compatibility                  off                            default
```

Bereitstellen und entfernen von Pools ohne löschen

> zpool import #zeigt was es zum importieren, also bereitstellen gibt
>
> zpool import poolname oder -a für alle importiert den Pool oder alle verfügbaren Pools
>
> Am elegantesten importiert man den Pool über
>
> **zpool import -d /dev/disk/by-id**
>
> damit man hinterher nicht sda, sdb, sondern die Bezeichner im Status sieht



```
root@pvews:~# zpool import
   pool: test
     id: 14075249358432111533
  state: ONLINE
 action: The pool can be imported using its name or numeric identifier.
 config:

        test                        ONLINE
          mirror-0                  ONLINE
            wwn-0x5000cca01a72ded4  ONLINE
            wwn-0x5000cca01a7b1e2c  ONLINE
          mirror-1                  ONLINE
            wwn-0x5000cca01a83a61c  ONLINE
            wwn-0x5000cca01a832d24  ONLINE
          mirror-2                  ONLINE
            wwn-0x5000cca01a83331c  ONLINE
            wwn-0x5000cca01a7495b8  ONLINE
root@pvews:~#
```

Nach der Bereitstellung würde man hier lediglich einen Mountpoint /test finden, bei TrueNAS /mnt/test. Dazu noch die Systemdatasets vom Proxmox selbst. Danach legen wir gleich mal ein Dataset namens dateisystem und ein Volume namens volume an.

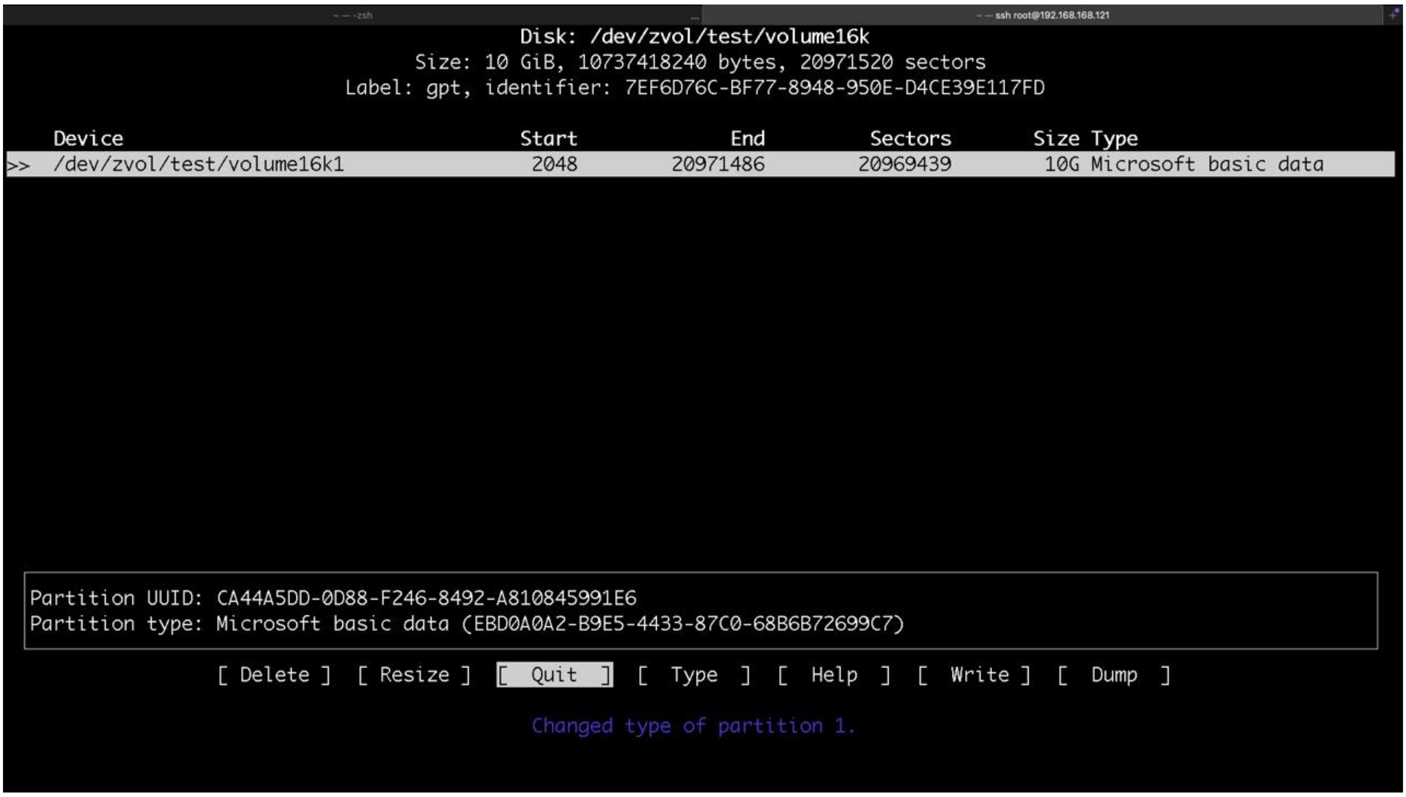
```
root@pvews:~# zfs list
NAME               USED  AVAIL     REFER  MOUNTPOINT
rpool             1.55G   898G      104K  /rpool
rpool/ROOT        1.54G   898G       96K  /rpool/ROOT
rpool/ROOT/pve-1  1.54G   898G     1.28G  /
rpool/data          96K   898G       96K  /rpool/data
test               213K  16.1T     37.5K  /test
root@pvews:~# #zfs
root@pvews:~# zfs create test/dateisystem
root@pvews:~# zfs create -V 100G test/volume
root@pvews:~#
```

> Datasets werden als Ordner gemountet und / oder /mnt
>
> Volumes findet man unter /dev/zd... und zusätzlich mit vernünftigen Namen unter /dev/zvol/tankname/....

```
root@pvews:~# zfs list
NAME               USED  AVAIL     REFER  MOUNTPOINT
rpool             1.55G   898G      104K  /rpool
rpool/ROOT        1.54G   898G       96K  /rpool/ROOT
rpool/ROOT/pve-1  1.54G   898G     1.28G  /
rpool/data          96K   898G       96K  /rpool/data
test               213K  16.1T     37.5K  /test
root@pvews:~# #zfs
root@pvews:~# zfs create test/dateisystem
root@pvews:~# zfs create -V 100G test/volume
root@pvews:~# zfs list
NAME               USED  AVAIL     REFER  MOUNTPOINT
rpool             1.55G   898G      104K  /rpool
rpool/ROOT        1.54G   898G       96K  /rpool/ROOT
rpool/ROOT/pve-1  1.54G   898G     1.28G  /
rpool/data          96K   898G       96K  /rpool/data
test               110G  16.0T     25.4K  /test
test/dateisystem    24K  16.0T       24K  /test/dateisystem
test/volume        110G  16.1T       12K  -
root@pvews:~#
```

> Datasets werden mit Linuxcontainern, Backupfiles, Vorlagen oder Serverdateien bespielt
>
> Volumes verhalten sich wie eingebaute Datenträger, jedoch virtuell. Sie stehen nach dem Import des Pools bereit und können in einer VM genutzt werden. Manipulationen an Volumes zur Vorbereitung oder Datenrettung können jedoch ebenso auf dem PVE Host vorgenommen werden

Diese Schritte übernimmt üblicherweise der Installer in der VM und sollen nur aufzeigen wie diese ZVOLs genutzt werden



```
test/volume16k    10.3G  16.1T     1.01G  -
root@pvews:~# dd if=/dev/zero of=/dev/zvol/test/volume16k bs=1M count=1024 status=progress
1024+0 records in
1024+0 records out
1073741824 bytes (1.1 GB, 1.0 GiB) copied, 14.9745 s, 71.7 MB/s
root@pvews:~# zfs list
NAME                USED  AVAIL     REFER  MOUNTPOINT
rpool              1.55G   898G      104K  /rpool
rpool/ROOT         1.54G   898G       96K  /rpool/ROOT
rpool/ROOT/pve-1   1.54G   898G     1.28G  /
rpool/data          104K   898G      104K  /rpool/data
test               21.3G  16.1T     25.4K  /test
test/dateisystem     24K  16.1T       24K  /test/dateisystem
test/volume        11.0G  16.1T     1.07G  -
test/volume16k     10.3G  16.1T     1.01G  -
root@pvews:~# cfdisk /dev/zvol/test/volume16k

root@pvews:~# cfdisk /dev/zvol/test/volume16k

Syncing disks.
root@pvews:~# mkfs.
mkfs.bfs      mkfs.cramfs  mkfs.ext3    mkfs.fat     mkfs.msdos   mkfs.xfs
mkfs.btrfs    mkfs.ext2    mkfs.ext4    mkfs.minix   mkfs.vfat
root@pvews:~# mkfs.
mkfs.bfs      mkfs.cramfs  mkfs.ext3    mkfs.fat     mkfs.msdos   mkfs.xfs
mkfs.btrfs    mkfs.ext2    mkfs.ext4    mkfs.minix   mkfs.vfat
root@pvews:~# mkfs.vfat /dev/zvol/test/volume16k
volume16k        volume16k-part1
root@pvews:~# mkfs.vfat /dev/zvol/test/volume16k-part1
```

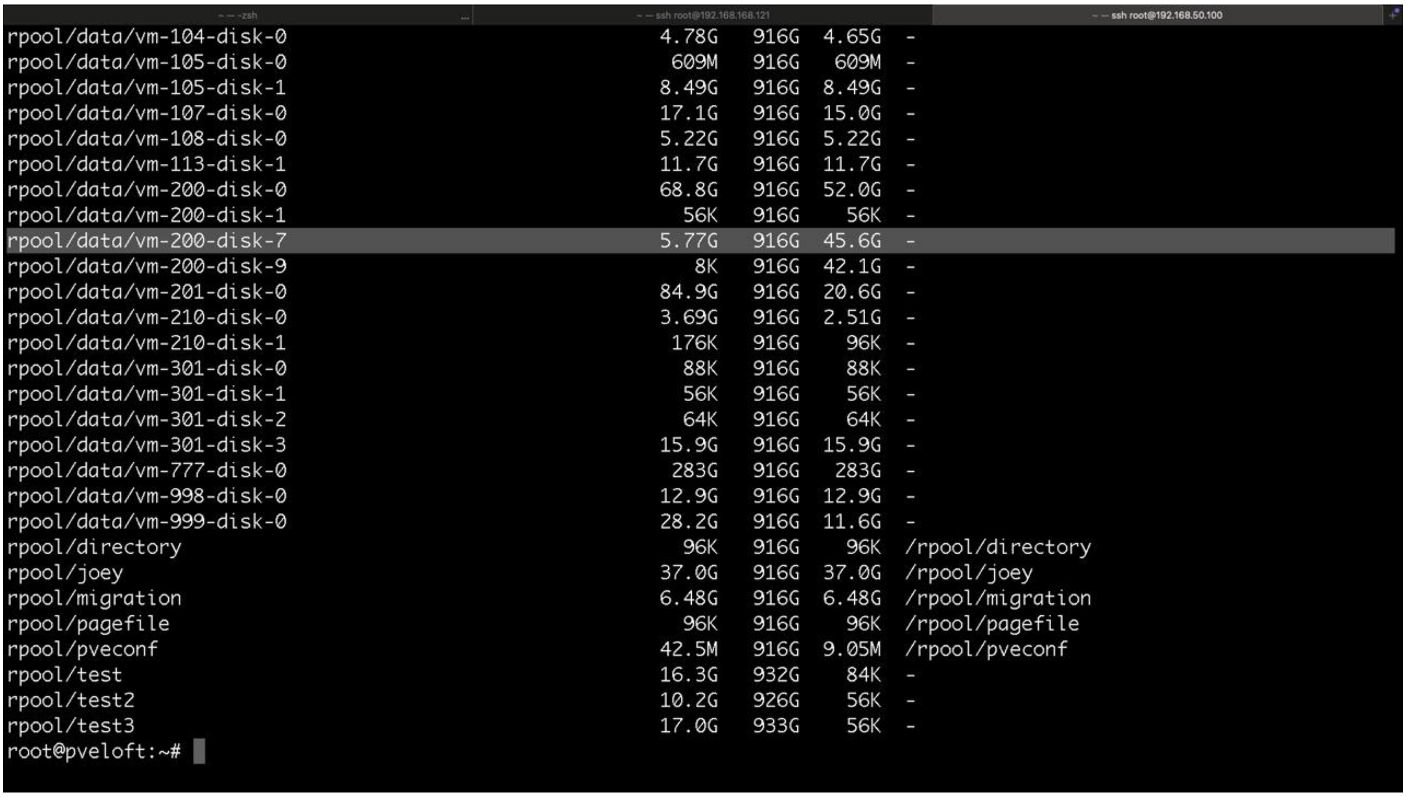
```
├─sdd1     8:49   0   2.7T  0 part
└─sdd9     8:57   0     8M  0 part
sde        8:64   0   2.7T  0 disk
├─sde1     8:65   0   2.7T  0 part
└─sde9     8:73   0     8M  0 part
sdf        8:80   0   2.7T  0 disk
├─sdf1     8:81   0   2.7T  0 part
└─sdf9     8:89   0     8M  0 part
sdg        8:96   0   2.7T  0 disk
├─sdg1     8:97   0   2.7T  0 part
└─sdg9     8:105  0     8M  0 part
sdh        8:112  0 931.5G  0 disk
├─sdh1     8:113  0  1007K  0 part
├─sdh2     8:114  0     1G  0 part
└─sdh3     8:115  0 930.5G  0 part
sdi        8:128  0 931.5G  0 disk
├─sdi1     8:129  0  1007K  0 part
├─sdi2     8:130  0     1G  0 part
└─sdi3     8:131  0 930.5G  0 part
sdk        8:160  0 447.1G  0 disk
├─sdk1     8:161  0 447.1G  0 part
└─sdk9     8:169  0     8M  0 part
sdl        8:176  0 447.1G  0 disk
├─sdl1     8:177  0 447.1G  0 part
└─sdl9     8:185  0     8M  0 part
zd16     230:16   0    10G  0 disk
└─zd16p1 230:17   0    10G  0 part
root@pvews:~# mkdir /mnt/restore
root@pvews:~# mount /dev/zd16p1 /mnt/restore/
```

```
devpts on /dev/pts type devpts (rw,nosuid,noexec,relatime,gid=5,mode=620,ptmxmode=000)
tmpfs on /run type tmpfs (rw,nosuid,nodev,noexec,relatime,size=9895684k,mode=755,inode64)
rpool/ROOT/pve-1 on / type zfs (rw,relatime,xattr,noacl)
securityfs on /sys/kernel/security type securityfs (rw,nosuid,nodev,noexec,relatime)
tmpfs on /dev/shm type tmpfs (rw,nosuid,nodev,inode64)
tmpfs on /run/lock type tmpfs (rw,nosuid,nodev,noexec,relatime,size=5120k,inode64)
cgroup2 on /sys/fs/cgroup type cgroup2 (rw,nosuid,nodev,noexec,relatime)
pstore on /sys/fs/pstore type pstore (rw,nosuid,nodev,noexec,relatime)
bpf on /sys/fs/bpf type bpf (rw,nosuid,nodev,noexec,relatime,mode=700)
systemd-1 on /proc/sys/fs/binfmt_misc type autofs (rw,relatime,fd=30,pgrp=1,timeout=0,minproto=5,maxproto=5,direct,pipe
_ino=24541)
mqueue on /dev/mqueue type mqueue (rw,nosuid,nodev,noexec,relatime)
hugetlbfs on /dev/hugepages type hugetlbfs (rw,relatime,pagesize=2M)
debugfs on /sys/kernel/debug type debugfs (rw,nosuid,nodev,noexec,relatime)
tracefs on /sys/kernel/tracing type tracefs (rw,nosuid,nodev,noexec,relatime)
fusectl on /sys/fs/fuse/connections type fusectl (rw,nosuid,nodev,noexec,relatime)
configfs on /sys/kernel/config type configfs (rw,nosuid,nodev,noexec,relatime)
sunrpc on /run/rpc_pipefs type rpc_pipefs (rw,relatime)
rpool on /rpool type zfs (rw,relatime,xattr,noacl)
rpool/ROOT on /rpool/ROOT type zfs (rw,relatime,xattr,noacl)
rpool/data on /rpool/data type zfs (rw,relatime,xattr,noacl)
lxcfs on /var/lib/lxcfs type fuse.lxcfs (rw,nosuid,nodev,relatime,user_id=0,group_id=0,allow_other)
/dev/fuse on /etc/pve type fuse (rw,nosuid,nodev,relatime,user_id=0,group_id=0,default_permissions,allow_other)
tmpfs on /run/user/0 type tmpfs (rw,nosuid,nodev,relatime,size=9895680k,nr_inodes=2473920,mode=700,inode64)
test on /test type zfs (rw,xattr,noacl)
test/dateisystem on /test/dateisystem type zfs (rw,xattr,noacl)
/dev/zd16p1 on /mnt/restore type vfat (rw,relatime,fmask=0022,dmask=0022,codepage=437,iocharset=iso8859-1,shortname=mix
ed,errors=remount-ro)
root@pvews:~#
```

Proxmox legt seine Datasets für LXC so ab

```
rpool/ROOT/klon                          744K   916G  33.8G  /rpool/ROOT/klon
rpool/ROOT/pve-1                        89.1G   916G  75.2G  /
rpool/azubi                               96K   916G    96K  /rpool/azubi
rpool/clone                               96K   916G    96K  /rpool/clone
rpool/data                               627G   916G   160K  /rpool/data
rpool/data/subvol-100-disk-0            26.0G  74.3G  25.7G  /rpool/data/subvol-100-disk-0
rpool/data/subvol-106-disk-0            1.36G  15.1G   900M  /rpool/data/subvol-106-disk-0
rpool/data/subvol-121-disk-0            1.76G  30.3G  1.68G  /rpool/data/subvol-121-disk-0
rpool/data/subvol-131-disk-0             958M  31.1G   958M  /rpool/data/subvol-131-disk-0
rpool/data/subvol-131-disk-1            3.19G  96.8G  3.19G  /rpool/data/subvol-131-disk-1
rpool/data/subvol-132-disk-0            1.21G  31.2G   866M  /rpool/data/subvol-132-disk-0
rpool/data/subvol-132-disk-1             220K   100G   220K  /rpool/data/subvol-132-disk-1
rpool/data/subvol-133-disk-0             707M  31.3G   707M  /rpool/data/subvol-133-disk-0
rpool/data/subvol-133-disk-1            18.4M   100G  18.4M  /rpool/data/subvol-133-disk-1
rpool/data/subvol-134-disk-0            1.05G  31.3G   687M  /rpool/data/subvol-134-disk-0
rpool/data/vm-101-disk-0                7.48G   916G  7.48G  -
rpool/data/vm-102-disk-0                  96K   916G    96K  -
rpool/data/vm-102-disk-1                22.5G   916G  22.5G  -
rpool/data/vm-102-disk-2                  64K   916G    64K  -
rpool/data/vm-103-disk-0                8.92G   916G  8.92G  -
rpool/data/vm-103-disk-1                3.06M   916G  3.06M  -
rpool/data/vm-103-disk-2                1.03G   917G   176K  -
rpool/data/vm-103-disk-3                1.65M   916G  1.65M  -
rpool/data/vm-104-disk-0                4.78G   916G  4.65G  -
rpool/data/vm-105-disk-0                 609M   916G   609M  -
rpool/data/vm-105-disk-1                8.49G   916G  8.49G  -
rpool/data/vm-107-disk-0                17.1G   916G  15.0G  -
rpool/data/vm-108-disk-0                5.22G   916G  5.22G  -
rpool/data/vm-113-disk-1                11.7G   916G  11.7G  -
```

und so die ZVOLs für VMs mit KVM, hier findet man die virtuellen Disk und /dev/zvol...



```
rpool/data/vm-104-disk-0                         4.78G  916G  4.65G  -
rpool/data/vm-105-disk-0                          609M  916G   609M  -
rpool/data/vm-105-disk-1                         8.49G  916G  8.49G  -
rpool/data/vm-107-disk-0                         17.1G  916G  15.0G  -
rpool/data/vm-108-disk-0                         5.22G  916G  5.22G  -
rpool/data/vm-113-disk-1                         11.7G  916G  11.7G  -
rpool/data/vm-200-disk-0                         68.8G  916G  52.0G  -
rpool/data/vm-200-disk-1                          56K   916G    56K  -
rpool/data/vm-200-disk-7                         5.77G  916G  45.6G  -
rpool/data/vm-200-disk-9                           8K   916G  42.1G  -
rpool/data/vm-201-disk-0                         84.9G  916G  20.6G  -
rpool/data/vm-210-disk-0                         3.69G  916G  2.51G  -
rpool/data/vm-210-disk-1                          176K  916G    96K  -
rpool/data/vm-301-disk-0                           88K  916G    88K  -
rpool/data/vm-301-disk-1                           56K  916G    56K  -
rpool/data/vm-301-disk-2                           64K  916G    64K  -
rpool/data/vm-301-disk-3                         15.9G  916G  15.9G  -
rpool/data/vm-777-disk-0                          283G  916G   283G  -
rpool/data/vm-998-disk-0                         12.9G  916G  12.9G  -
rpool/data/vm-999-disk-0                         28.2G  916G  11.6G  -
rpool/directory                                    96K  916G    96K  /rpool/directory
rpool/joey                                       37.0G  916G  37.0G  /rpool/joey
rpool/migration                                  6.48G  916G  6.48G  /rpool/migration
rpool/pagefile                                     96K  916G    96K  /rpool/pagefile
rpool/pveconf                                    42.5M  916G  9.05M  /rpool/pveconf
rpool/test                                       16.3G  932G    84K  -
rpool/test2                                      10.2G  926G    56K  -
rpool/test3                                      17.0G  933G    56K  -
root@pveloft:~#
```

Thin- und Thickprovisioning für Datasets und Volumes

> Am Beispiel von Proxmox VE via GUI wird hier eine Thinprovision Festplatte für VM 301 mit 32GB und einer Standardblockgröße von 16k erzeugt. 8k sind nur bei Raid1 und 10 möglich. Hakt man den Thinprovision Haken nicht an, wird beim Erzeugen noch die Option -o refreservation=32G ergänzt. Diese macht bei ZFS mit Autosnapshots aber keinen Sinn

zfs create -s -b 16k -V 33554432k rpool/data/vm-301-disk-4

rpool/data/vm-301-disk-4   56K 1.17T   56K - **#hier kein Mount sonder unter /dev/zvol/...**

> Bei LXC werden Datasets genutzt und der Platz via Reservierung genutzt, wobei bei der genutzten Refquota auch die Snapshotaufhebezeiten den Nettoplatz verkleinern, daher größer dimensionieren, gerne mal doppelt so groß

zfs create -o acltype=posixacl -o xattr=sa -o refquota=33554432k rpool/data/subvol-106-disk-1

```
rpool/data/subvol-106-disk-1                    96K  32.0G    96K  /rpool/data/subvol-106-disk-1 #hostzugriff möglich
```

> Virtuelle Diskimages wie QCOW2, VMDK oder RAW-Dateien legt man üblicherweise nicht in Datasets, bestenfalls für NFS oder iSCSI Server

Snapshots

> Snapshots können jederzeit erstellt oder gelöscht werden. Die Datasets und Volumes befinden sich immer in einem Livezustand der sich durch die Vektorkette der eventuell vorhandenen Snapshots ergibt. Entfernt man einen Snapshot, so verändert sich der Weg der Vektoren. Das Ganze geschieht fast ohne Last!
>
> ZFS selbst bringt weder Workflows für Snapshotting, noch Replikationen mit. Es stellt nur die Werkzeuge bereit
>
> Snapshots generiert man bei Proxmox offiziell mit der GUI manuell oder für Replikationen, jedoch leistet dieser Workflow nicht genug.
>
> Daher nutzen wir...

```
apt install zfs-auto-snapshot -y
```

> ZFS-AUTO-SNAPSHOT legt seine Verknüpfungen in alle Cronordner. Proxmox VE weiß erst mal nichts davon. Zu berücksichtigen ist hier lediglich daß wir den Pool nicht über 80% befüllen wollen. Gehen wir darüber hinaus, müssen wir in den Cronordnern die Verknüpfungen anpassen um die Aufhebezeiten zu reduzieren.

```
nano /etc/crontab #kann angepasst werden, die viertelstündlichen finden sich unter cron.d!
```

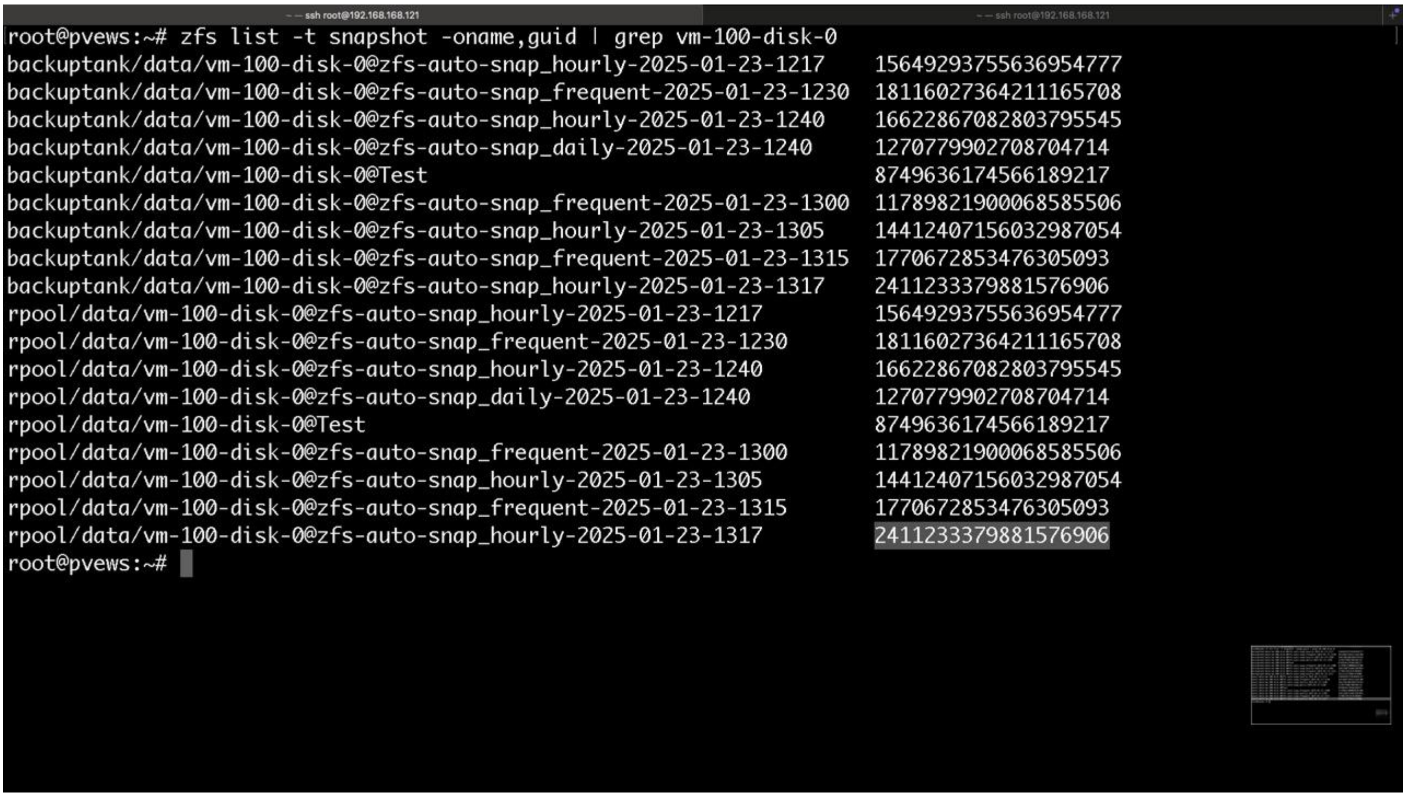
```
# .---------------- minute (0 - 59)
# |  .------------- hour (0 - 23)
# |  |  .---------- day of month (1 - 31)
# |  |  |  .------- month (1 - 12) OR jan,feb,mar,apr ...
# |  |  |  |  .---- day of week (0 - 6) (Sunday=0 or 7) OR sun,mon,tue,wed,thu,fri,sat
# |  |  |  |  |
# *  *  *  *  * user-name command to be executed
```

17 * * * * root cd / && run-parts --report /etc/**cron.hourly**
25 6 * * * root test -x /usr/sbin/anacron || { cd / && run-parts --report /etc/**cron.daily**; }
47 6 * * 7 root test -x /usr/sbin/anacron || { cd / && run-parts --report /etc/**cron.weekly**; }
52 6 1 * * root test -x /usr/sbin/anacron || { cd / && run-parts --report /etc/**cron.monthly**; }

> cat /etc/cron.hourly/zfs-auto-snapshot

> exec zfs-auto-snapshot --quiet --syslog --label=hourly **--keep=96** // #also in dem Fall 96 Stunden

> Wir empfehlen für den Start folgende Settings: Drei Monate, sechs Wochen, zehn Tage, 96h und 12 Viertelstunden
>
> Damit kommt man auf ca. 2,5x- 3x so viele Daten wie belegt wären ohne Snapshots. Erfahrungswert.
>
> Das wären pro virtueller Disk oder LXC Mountpoint etwa 127 Snaphots, bei 20 Disks über 2540.
>
> Nicht davon abschrecken lassen, es geht auch easy fünfstellig, wenn die IO stimmt!
>
> Die in PVE eingebaute Snapshotfunktion sollte nicht mit den Autosnapshots kombiniert werden.
>
> Im Problemfall stoppen wir die VM, manipulieren was zu machen ist und starten sie wieder!
>
> Der eigentliche Vorteil der Snapshots in der GUI ist die Notierung der VM Definition zur Zeit der Erstellung, was wir über ein Backup im hauseigenen Postinstaller kompensieren. Dort findet man die Historie der PVE Config



Kleines manuelles Beispiel, was für uns aber Z-A-S übernimmt

Snapshot

> zfs snapshot tank/sub/datasetodervolume@snapshotname

Auflisten

> zfs list -t snapshot tank/sub/datasetodervolume

VM oder LXC ausmachen

Rollback

> zfs rollback -r rpool/data/vm-100-disk-0@Test #danach ist alles neuere weg!

VM oder LXC starten

Dynamik von Snapshots

Wir unterscheiden zwei Sorten von Datennutzung

- Zunehmende Datennutzung wie bei einem NAS
  - Es kommt nur was dazu und wird selten gelöscht
  - Viele Snapshots auf lange Zeit ändern nichts an der Gesamtnutzung des Pools
  - Replikation außer Haus kann in kleinen Schritten passieren, da die Gesamtübertragung gleich groß ist
- Rotierende Datennutzung wie bei einem Datenbankserver
  - Daten verändern sich permanent, wenig Zuwachs
  - Viele Snapshots auf lange Zeit ändern **brachial viel** an der Gesamtnutzung des Pools
  - Replikation außer Haus sollte in einem Schritt, ohne Zwischensnapshots passieren, da die Gesamtübertragung sonst ein vielfaches größer sein kann
  - Nur essenzielle Wiederherstellungspunkte sollten aufgehoben werden um das System nicht unnötig zu befüllen

Dynamik von Snapshots

> Dank ZFS kann man langfristig Replikationen inkrementell vornehmen, jedoch dauert die erste Replikation natürlich länger, wenn schon Datenbestand da war

Übliche Vorgehensweise einer Datenübertragung

PC NTFS>SMB> Datei > SAMBA Server BTRFS LVM > EXT4 Datei = Konsistent? Bitrod?

> Zu viele Köche verderben die Köchin

Funktion von Snapshotreplikation

Von ZFS zu Datei, was wenig Praxis findet

> zfs send tank/dataset@snapshot > datei.zfs

> zfs recv tank < datei.zfs

Von ZFS zu ZFS, was üblich ist

> zfs send | zfs recv #laufen immer gleichzeitig

zfs send rpool/data/vm-100-disk-0@Test | zfs recv -dvF backuptank
#überträgt den Zustand Test auf ein lokalen weiteren Pool, wobei das zfs Tool immer doppelt läuft

Pushreplikation

zfs send -pvwRI rpool/data/vm-100-disk-0@Test rpool/data/vm-100-disk-0@zfs-auto-snap_hourly-2025-01-23-1317 | zfs recv -dvF backuptank
#Fortschritt, Infos, Rohdaten plus Historie, ideal für verschlüsselte Volumes, erspart das De- und Komprimieren

Workflow

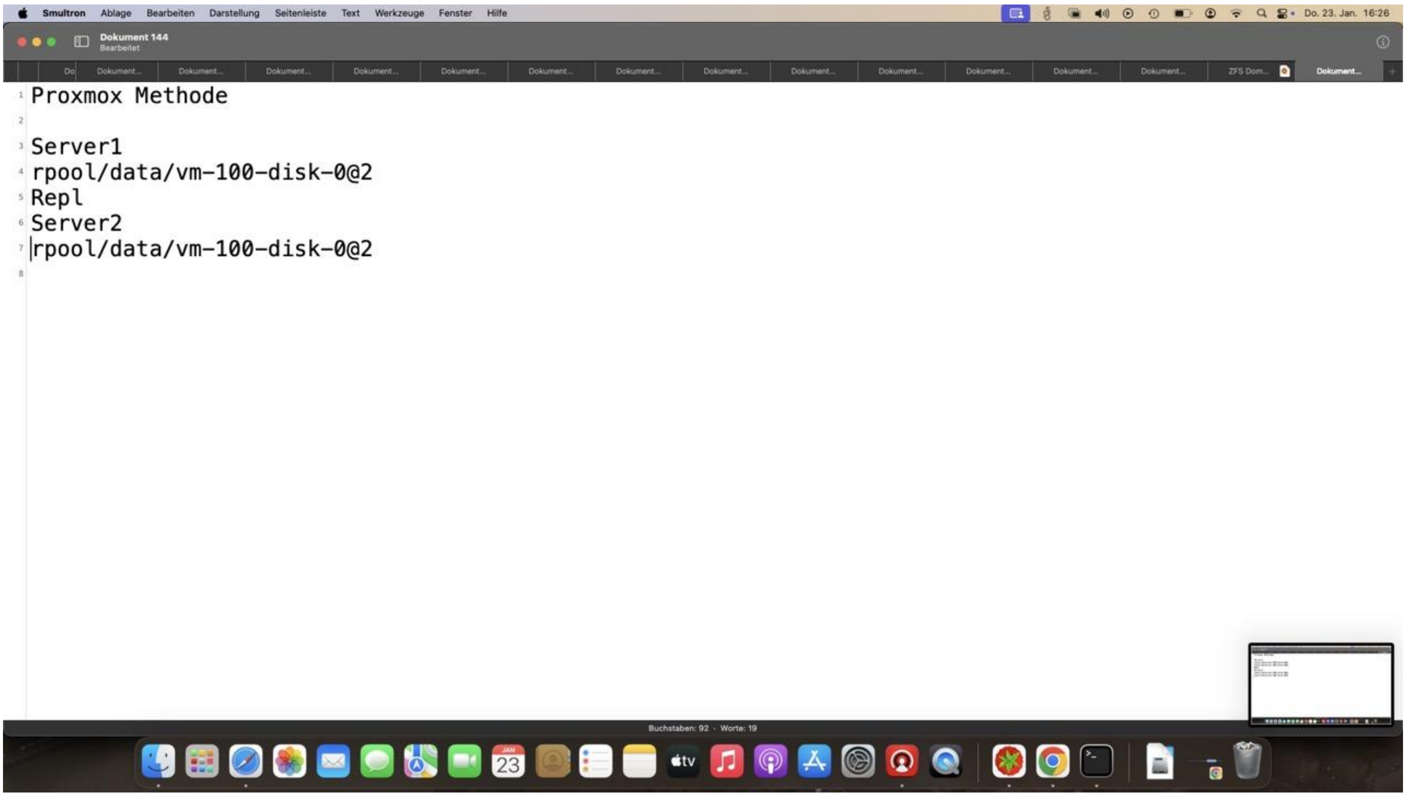
- Erzeugen eines Datasets oder Volumes
- Snapshot auf Dataset oder Volume ausführen
- Übertragung des Snapshotzustands auf Ziel, optional mit Zwischenständen

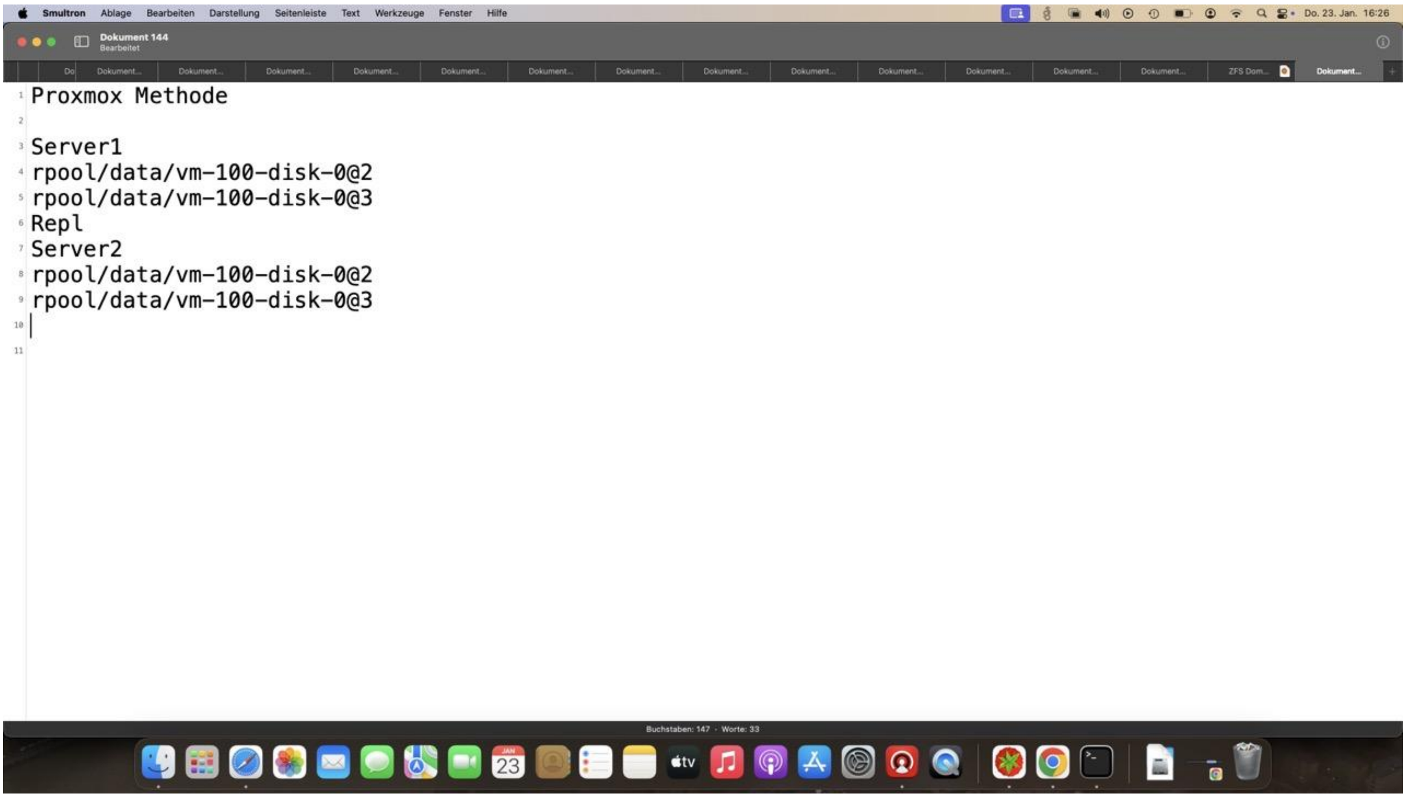
Trojanerproblem: Wer auf Quelle sitzt kann Pushziele löschen

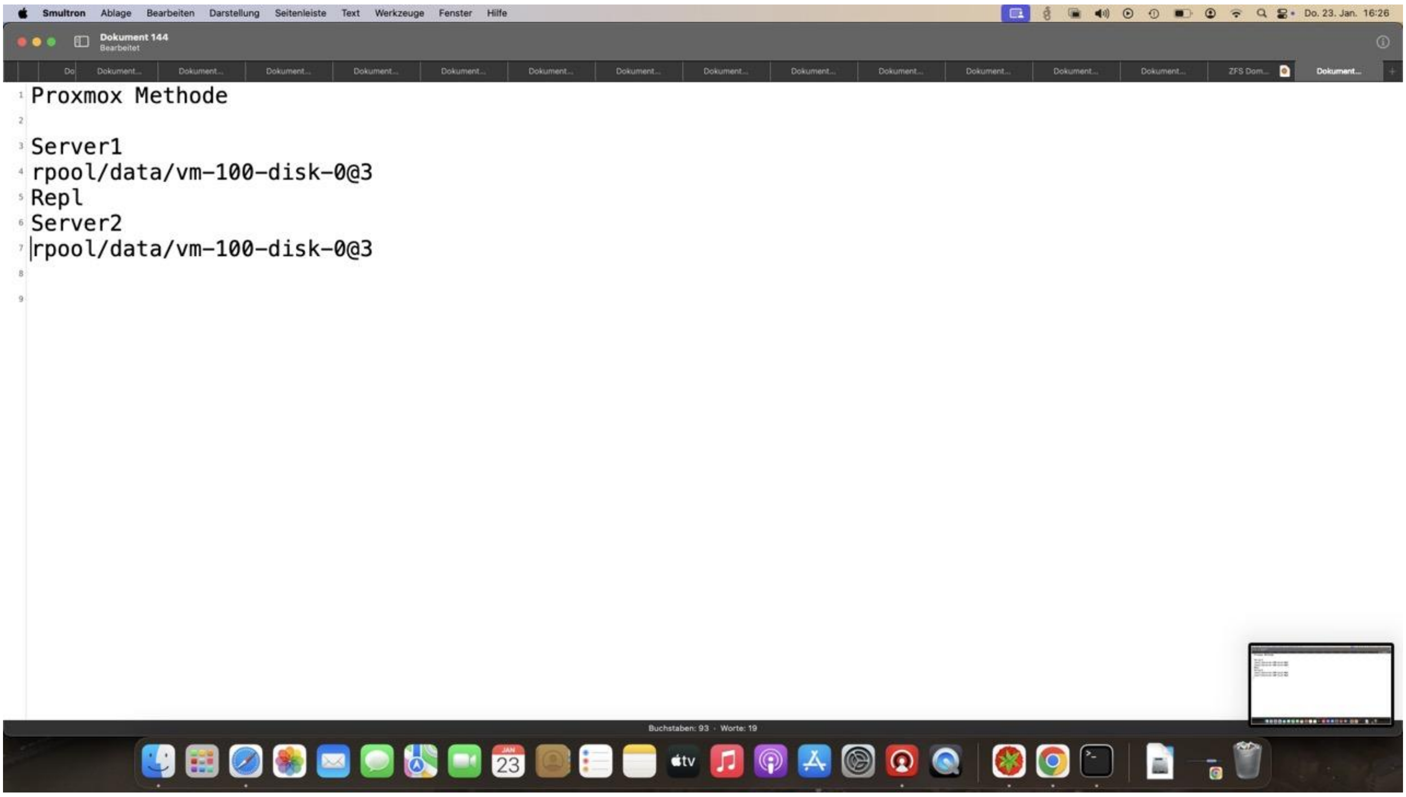
Lösung: Pullreplikation, ähnlich Backupsoftware, Transport SSH, Daten ZFS
Quelle |> Pullserver

zfs **recv** | ssh root@quelle zfs **send** #Pull Replikation via SSH

Von PVE zu PVE, was uns zu wenig ist


Smultron Ablage Bearbeiten Darstellung Seitenleiste Text Werkzeuge Fenster Hilfe
Do. 23. Jan. 16:26
Dokument 144
Bearbeitet
Proxmox Methode
Server1
rpool/data/vm-100-disk-0@2
Repl
Server2
rpool/data/vm-100-disk-0@2
Buchstaben: 92 · Worte: 19


Smultron Ablage Bearbeiten Darstellung Seitenleiste Text Werkzeuge Fenster Hilfe
Do. 23. Jan. 16:26
Dokument 144
Bearbeitet
Proxmox Methode
Server1
rpool/data/vm-100-disk-0@2
rpool/data/vm-100-disk-0@3
Repl
Server2
rpool/data/vm-100-disk-0@2
rpool/data/vm-100-disk-0@3
Buchstaben: 147 · Worte: 33



> Proxmox baut selbst keine Notfallpunkte auf und nutzt ZFS nur als Transport, nimmt jedoch manuelle und automatisierte Snapshots mit, auch von Z-A-S!

> Weitere Kopien per ZFS sind nur per Pushverfahren möglich, was ideal für Angreifer ist wenn sie alles löschen möchten. Ein Backupserver ist hier unverzichtbar, bei Pullreplikation gehts ggf. auch ohne. Außer Haus ist es nicht möglich ein ZFS Replikat zu senden, da sich der Partner im lokalen Cluster befinden muss!

Nondestruktiver Rollback nach Trojanerbefallt bei Forensikbedarf

---

Version #3
Erstellt: 31 Januar 2025 12:18:16 von Christian Zengel (sysops GmbH)
Zuletzt aktualisiert: 31 Januar 2025 15:20:12 von Christian Zengel (sysops GmbH)