ZFS Benchmark

Performance in hyper-converged infrastructures with Proxmox VE and integrated ZFS Storage.

To optimize performance in hyper-converged deployments with Proxmox VE and ZFS storage, the appropriate hardware setup is essential. This benchmark presents a possible setup and its resulting performance, with the intention of supporting Proxmox users in making better decisions.

Hyper-converged setups with ZFS can be deployed with Proxmox VE, starting from a single node and growing to a cluster. We recommend the use of enterprise-class NVMe SSDs and at least a 10-gigabit network for Proxmox VE storage replication1. And as long as CPU power and memory are sufficient, a single node can reach reasonably good performance levels.

- By default, ZFS is a combined file system and logical volume manager, with various redundancy levels. Virtual machines and containers can both share the storage.
- ZFS is 100% software-defined and fully open-source.
- ZFS employs continuous integrity checks and protection against data corruption.
- A Proxmox VE and ZFS storage can be extended with additional disks on the fly, without any downtime, to match growing workloads.[1]
- The Proxmox VE virtualization platform has integrated ZFS storage since the release of Proxmox VE 3.4, in 2014. Since then, it has been used on thousands of servers worldwide, which has provided us with enormous amounts of feedback and experience.

TABLE OF CONTENTS

1 Expansion is limited on some vdev types

TESTBED CONFIGURATION

All benchmarks summarized in this paper were conducted in December 2020, on standard server hardware, with a default Proxmox VE/ZFS server installation. The following section describes the testbed configuration.

SERVER HARDWARE

For the benchmarks, we used a server with the below specifications:

CPU:	Single AMD EPYC 7302P 16-Core Processor
Mainboard:	GIGABYTE MZ32-AR0-00
Case:	2U Supermicro Chassis 8x Hotswap
Memory:	8 x 16 GB DDR4 FSB3200 288-pin 2Rx8 Samsung M393A2K43DB3-CWE
Disk:	4 x Micron 9300 Max 3.2 TB (MTFDHAL3T2TDR)

SOFTWARE VERSION (AUG/SEP 2020)

This benchmark was conducted using Proxmox VE 6.2 5.4.55-1-pve, ZFS 0.8.4 (Appendix, 2.).

SSD FOR ZFS

It's essential to use reliable, enterprise-class SSDs, with high endurance and power-loss protection. We also recommend testing each SSD's write performance with the Flexible I/O (fio) tester[2], before using them for ZFS.

The following table shows fio write results from a traditional spinning disk and from various selected SSDs:

	Bandwidth (KB)	4K IO/s	Latency (ms)
Intel Optane SSD DC P4800X Series 375 GB	322931	80732	0.01
Intel SSD DC P3700 Series 800 GB	300650	75162	0.01
Micron 9300 MAX 3.2 TB	205824	51000	0.02
Samsung SM863 240GB 2.5inch SSD	69942	17485	0.06
Intel DC S3510 120GB	50075	12518	0.08
Intel DC S3500 120GB	48398	12099	0.08
Samsung SSD 850 EVO 1TB	1359	339	2.94
Crucial MX100 512GB	1017	254	3.93
Seagate Constellation 7200.2 500 GB	471	117	8.47

2 fio, https://fio.readthedocs.io/

Based on these fio tests, we decided to use 4 x Micron 9300 MAX (MTFDHAL3T2TDR), 2.5", 3.20 TB U.2 SSD. We connected 4 U.2 SSDs to the server, using the on board SLink connectors.

We used the following fio command for the device tests:

```
fio --ioengine=libaio --filename=/dev/sdx --direct=1 --sync=1 --rw=write --bs=4K
          --numjobs=1 --iodepth=1 --runtime=60 --time_based --name=fio
```

Note: This command will destroy any data on your disk.

STORAGE

For the benchmarks, we created a single ZFS pool on the Micron 9300 MAX. No ZFS Intent Log[3] or Special Device[4] was used.

The benchmarks show the following redundancy levels:

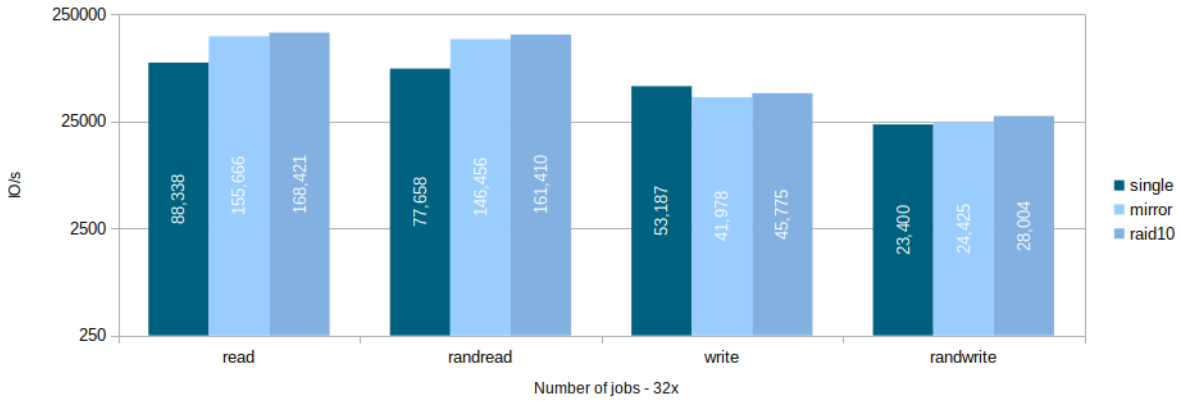
- Single vdev[5] (single disk)
- Mirrored vdevs (RAID1, 2x disks)
- Striped Mirrored vdevs (RAID10, 4x disks)

ZFS was used with the adaptations below:

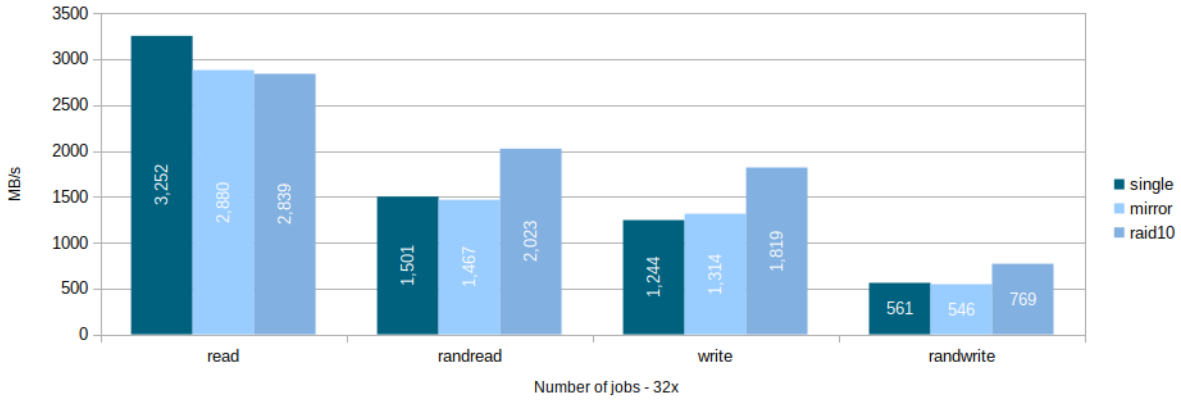
```
:~# cat /etc/modprobe.d/zfs.conf
# limit maximum ARC size
options zfs zfs_arc_max=4294967296

:~# update-initramfs -u
```

3 ZFS Intent Log (ZIL), https://manpages.debian.org/unstable/zfsutils-linux/zpool.8.en.html#Intent_Log
4 Special Device, https://manpages.debian.org/unstable/zfsutils-linux/zpool.8.en.html#Special_Allocation_Class
5 Virtual Device (vdev), https://manpages.debian.org/unstable/zfsutils-linux/zpool.8.en.html#Virtual_Devices_(vdevs)

ZFS POOL COMPARISON



6



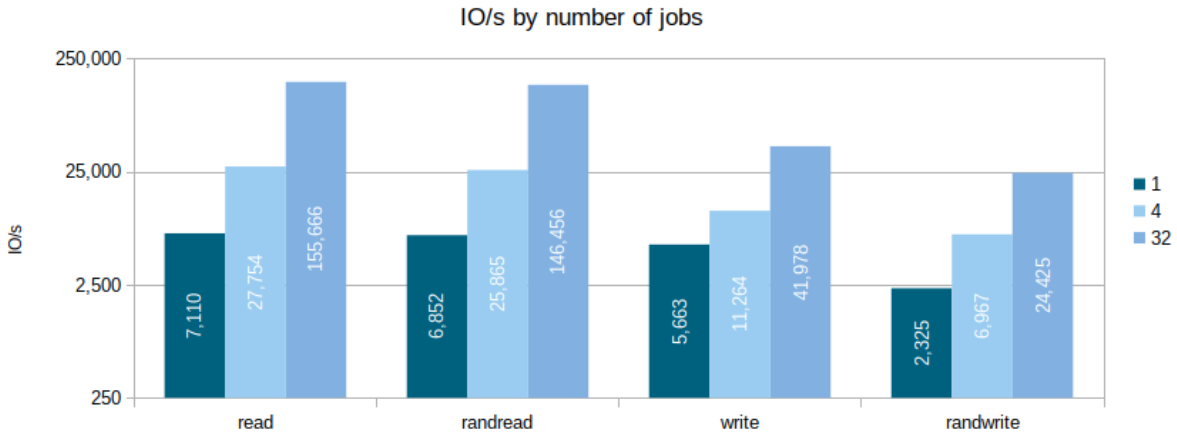
SUMMARY

The high sequential read measurements are the result of fio running the jobs simultaneously with the same data. Since ZFS keeps recently used data in its ARC[7], fio was able to read most data directly from memory.

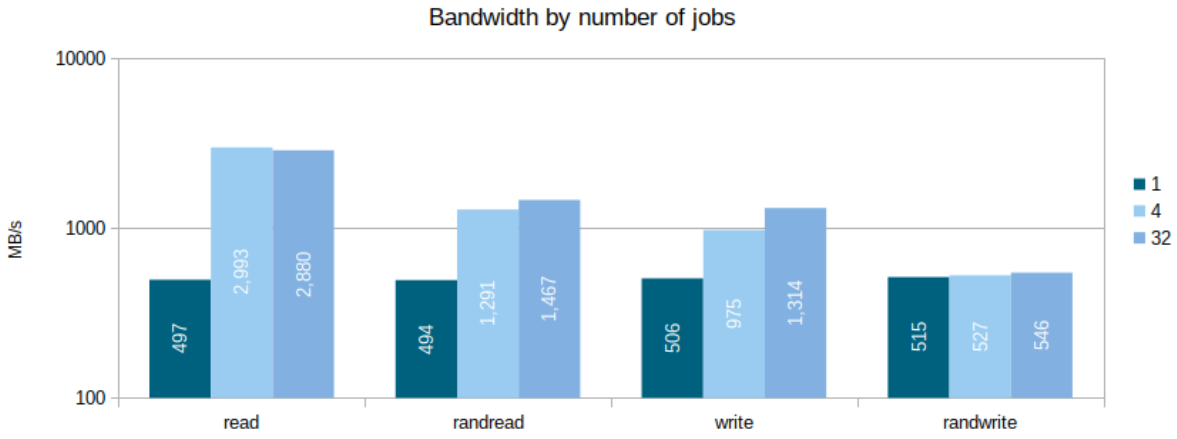
The created zvol used the '*primarycache=metadata*'[8] option to reduce data caching during read benchmarks.

6 The graphs is in logarithmic scale.
7 ARC (adaptive replacement cache)
8 Appendix 6, zvol properties

A SINGLE CLIENT IS NOT ENOUGH (MIRROR)



9



10

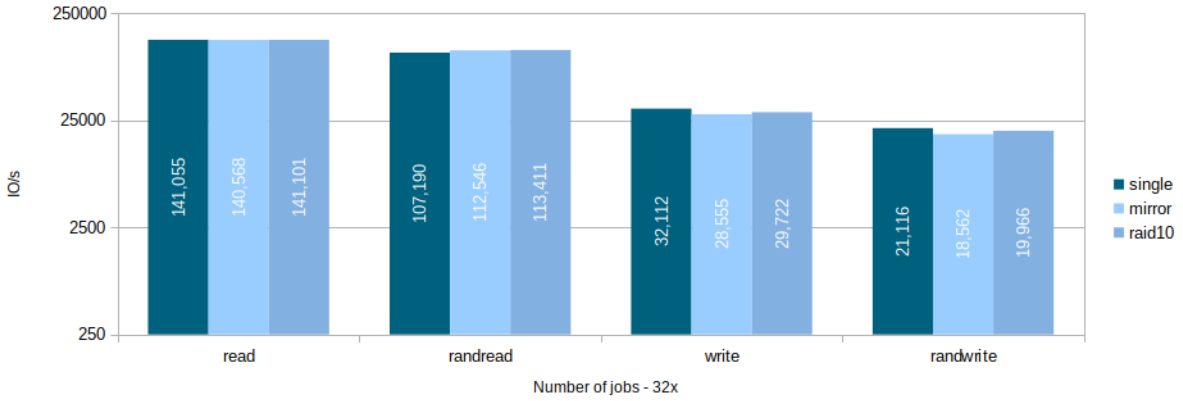
SUMMARY

The fio test was run on a ZFS pool with a vdev mirror using two U.2. Micron 9300 MAX. The graphs show that the IO/s and bandwidth rise, as IO concurrency increases.

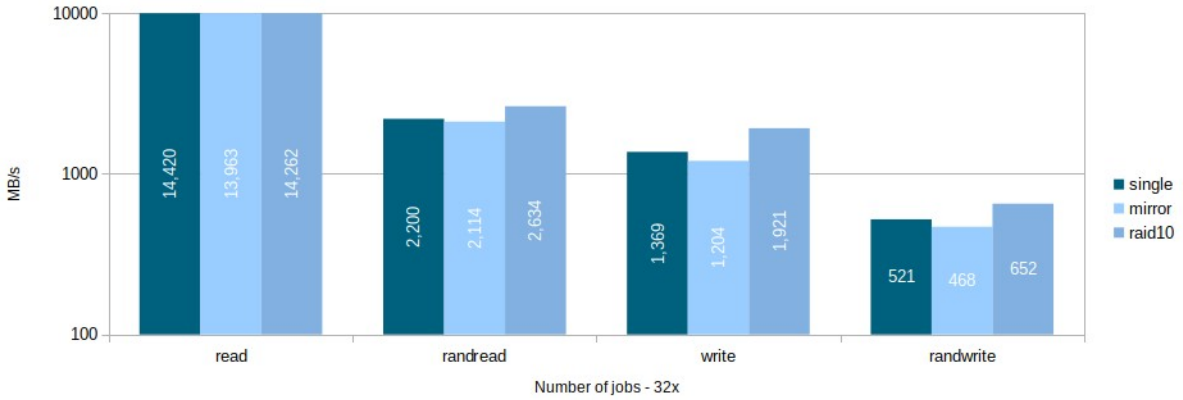
The created zvol used the '*primarycache=metadata*'[11] option to reduce data caching during read benchmarks.

9 The graphs is in logarithmic scale.
10 The graphs is in logarithmic scale.
11 Appendix 6, zvol properties

LINUX VIRTUAL MACHINE



12



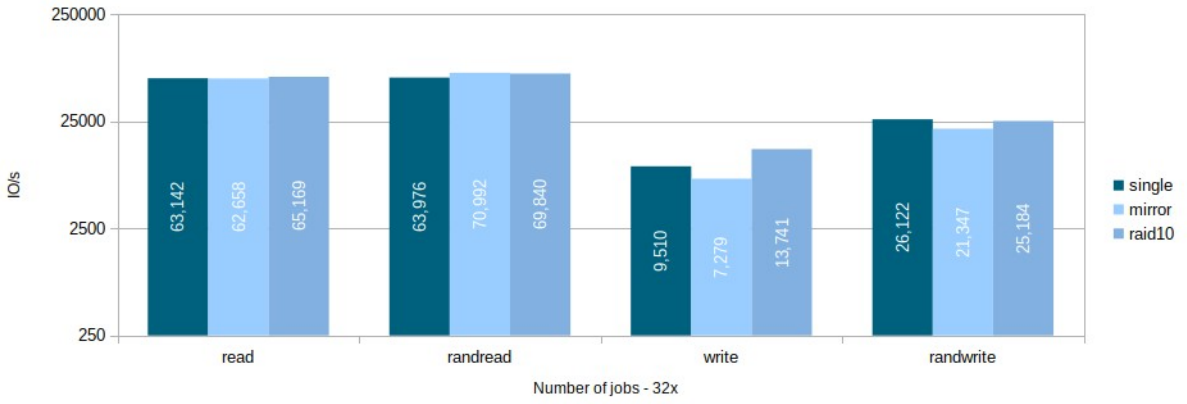
13

SUMMARY

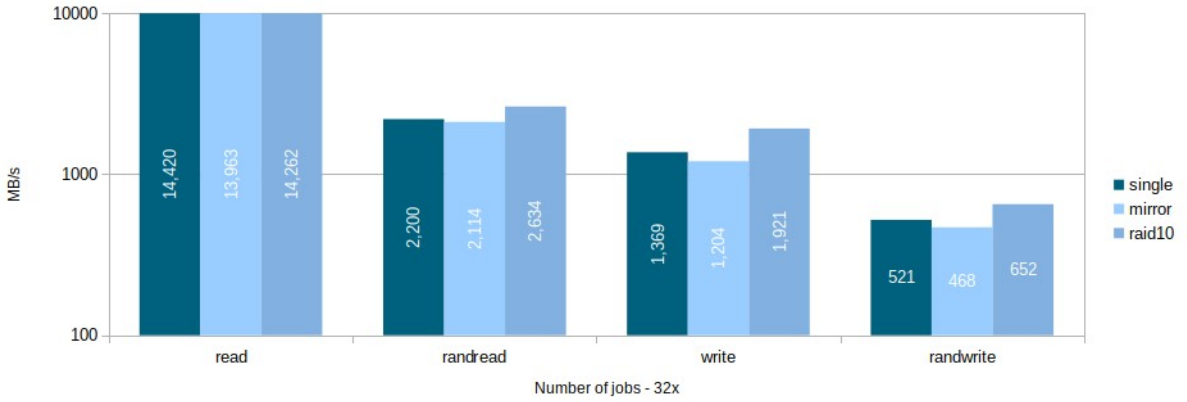
As stated in the pool comparison, fio reads the same data in all of its jobs. Therefore the bandwidth jumps in rand/-read are due to ZFS's ARC[14].

12 The graphs are in logarithmic scale.
13 The graphs are in logarithmic scale.
14 ARC (adaptive replacement cache)

WINDOWS VIRTUAL MACHINE



15



16

SUMMARY

As stated in the pool comparison, fio reads the same data in all of its jobs. Therefore, the bandwidth jumps in rand/-read are due to ZFS's ARC[17].

15 The graphs are in logarithmic scale.
16 The graphs are in logarithmic scale.
17 ARC (adaptive replacement cache)

HARDWARE FAQ

Can I use NVMe SSDs, for example M2 or PCI-Express cards?

Yes, but the U.2 NVMe SSDs provide better disk performance compared to M.2 variants. It also allows for a more space-saving build than with PCI-Express cards.

Can I create a fast pool with SSDs and a storage pool with HDDs?

Yes, building several pools can help in situations where budget is limited.

Which CPUs are better: More cores or a higher frequency?

CPUs with both a lot of cores and a high frequency are the best choice. This is true for Intel Xeon and AMD Epyc CPUs.

Should I use NUMA or UMA systems?

The use of *Non-Uniform Memory Access* (NUMA) or *Uniform Memory Access* (UMA) is a secondary factor. It can be compared to better speed through locality vs easier configuration and management.

How much RAM do I need per server?

ZFS needs memory for its Adaptive Replacement Cache. Also, VM/CT workloads will need memory. ZFS uses 50% of the available free memory. The bigger the available free memory, the more caching ZFS can do. In general, the best recommendation is to use as much RAM as possible.

Why did you use 3.20 TB U.2 NVMe SSDs for your tests?

To utilize the AMD Epyc Zen2 platform, it needs multiple modern U.2 disks.

Can I use consumer or pro-sumer SSDs, as these are much cheaper than enterprise-class SSDs?

No. Never. These SSDs wont provide the required performance, reliability or endurance. See the fio results from before and/or run your own fio tests.

Can I mix various disk types?

It is possible, but best not to mix disk types in a vdev. Otherwise, the performance will drop and a slower disk might be falsely marked as faulty (too slow).

However, combinations of vdev types like log and special can be of a different type.

Can I mix different disk sizes?

Yes, but for any mirror or raidz, the smallest disk will determine the available space.

APPENDIX

1 – PROXMOX VE SOFTWARE VERSIONS

proxmox-ve	6.2-1 (running kernel 5.4.55-1-pve)		
pve-manager	6.2-11 (running version: 6.2-11/db10c37a)		
pve-kernel-5.4	6.2-5	lxc-pve	4.0.3-1
pve-kernel-helper	6.2-5	lxcfs	4.0.3-pve3
pve-kernel-5.4.55-1-pve	5.4.55-1	novnc-pve	1.1.0-1
ceph	15.2.4-pve1	proxmox-backup-client	0.8.11-1
ceph-fuse	15.2.4-pve1	proxmox-mini-journalreader	1.1-1
corosync	3.0.4-pve1	proxmox-widget-toolkit	2.2-10
criu	3.11-3	pve-cluster	6.1-8
glusterfs-client	5.5-3	pve-container	3.1-12
ifupdown	0.8.35+pve1	pve-docs	6.2-5
ksm-control-daemon	1.3-1	pve-edk2-firmware	2.20200531-1
libjs-extjs	6.0.1-10	pve-firewall	4.1-2
libknet1	1.16-pve1	pve-firmware	3.1-2
libproxmox-acme-perl	1.0.4	pve-ha-manager	3.0-9
libpve-access-control	6.1-2	pve-i18n	2.1-3
libpve-apiclient-perl	3.0-3	pve-qemu-kvm	5.1.0-1
libpve-common-perl	6.2-1	pve-xtermjs	4.7.0-1
libpve-guest-common-perl	3.1-2	qemu-server	6.2-13
libpve-http-server-perl	3.0-6	smartmontools	7.1-pve2
libpve-storage-perl	6.2-6	spiceterm	3.1-1
libqb0	1.0.5-1	vncterm	1.6-2
libspice-server1	0.14.2-4~pve6+1	zfsutils-linux	0.8.4-pve1
lvm2	2.03.02-pve4		

2 – BIOS SETTINGS GIGABYTE MZ32-AR0-00)

Advanced: PCI Subsystem Settings PCIE_7 Lanes	 [x4 x4 x4 x4]	 ##	 bifurcation for U.2 SSD
AMD CBS: CPU Common Options Performance Custom Core Pstates Custom Pstate0 Custom Pstate1 Custom Pstate2	 [Auto] [Disabled] [Disabled]	 ##	 fix P-state, leave sub-settings default
Global C-state Control Local APIC Mode	[Disabled] [x2APIC]		
DF Common Options Memory Addressing NUMA nodes per socket	 [NPS0]	 ##	 check as last option, will be set automatically by other options
UMC Common Options DDR4 Common Options Enforce POR Overclock Memory Clock Speed	 [Enabled] [1600MHz]	 ##	 fixed memory speed (uncoupled from Infinity Fabric)
Security TSME	 [Disabled]	 ##	 Transparent Secure Memory Encryption root (5-7 ns)
NBIO Common Options SMU Common Options Power Policy Quick Setting Determinism Control Determinism Slider APBDIS DF Cstates Fixed SOC Pstate CPPC	 [Best Performance] [Manual] [Performance] [1] [Disabled] [P0] [Disabled]	 ## ## ## ##	 identical performance for all cores fixed Infinity Fabric P-state control Infinity Fabric power states Allows OS to make performance/power optimization requests using ACPI CPPC

3 – PROXMOX VE HOST SETTINGS

```
# /etc/default/grub
GRUB_CMDLINE_LINUX_DEFAULT="quiet pcie_aspm=off amd_iommu=on iommu=pt
mitigations=off"
```

4 – FIO TESTS

The fio command was repeatedly called with the values in brackets.

4.1 – fio command for Proxmox VE node

```
fio --ioengine=psync --filename=/dev/zvol/tank/fio --size=9G --time_based
--name=fio --group_reporting --runtime=600 --direct=1 --sync=1 --iodepth=1
--rw=<write|read|randwrite|randread> --threads --bs=<4K|4M> --numjobs=<1|4|32>
```

4.2 - fio command for "VM Performance (Windows)"

```
fio --ioengine=windowsaio --filename=test_fio --size=9G --time_based
--name=fio --group_reporting --runtime=600 --direct=1 --sync=1 -iodepth=1
--rw=<write|read|randwrite|randread> --threads --bs=<4K|4M> --numjobs=<1|4|32>
```

4.3 - fio command for "VM Performance (Linux)"

```
fio --ioengine=psync --filename=/dev/mapper/test_fio --size=9G --time_based
--name=fio --group_reporting --runtime=600 --direct=1 --sync=1 --iodepth=1
--rw=<write|read|randwrite|randread> --threads --bs=<4K|4M> --numjobs=<1|4|32>
```

5 – VM CONFIGURATION

qemu cache=writeback	
qm config 101 (Proxmox VE 6.2)	
agent:	1
bios:	ovmf
bootdisk:	scsi0
cores:	4
efidisk0:	tank:vm-100-disk-0,format=raw,size=128K
machine:	q35
memory:	16384
name:	PVE6
net0:	virtio=CE:F4:46:8C:FF:95,bridge=vmbr0,firewall=1
numa:	0
ostype:	l26
scsi0:	tank:vm-100-disk-1,discard=on,format=raw,iothread=1,size=64G,ssd=1
scsihw:	virtio-scsi-single
smbios1:	uuid=65f3eff6-cbeb-4f12-871d-4e14cd42e1db
sockets:	1
vga:	virtio
vmgenid:	72fdba72-015f-489d-bb4e-6921111982ab
Edition	Proxmox VE 6.2
OS build	proxmox-ve:6.2-1 (running kernel: 5.4.60-1-pve)
Version	pve-manager:6.2-11 (running version: 6.2-11/22fb4983)
qm config 100 (Windows 2k19)	
agent:	1
bios:	ovmf
bootdisk:	scsi0
cores:	4
efidisk0:	tank:vm-101-disk-1,size=1M
machine:	q35
memory:	16384
name:	Win2019
net0:	virtio=56:43:AC:12:39:4E,bridge=vmbr0,firewall=1
numa:	0
ostype:	win10
scsi0:	tank:vm-101-disk-0,discard=on,iothread=1,size=64G,ssd=1
scsihw:	virtio-scsi-single
smbios1:	uuid=e83e6d64-bd8a-4814-b221-72dea14e2199
sockets:	1
vga:	virtio
vmgenid:	ed59f395-e7c1-4564-b577-567a51d81258
Edition	Windows Server 2019 Standard Evaluation
OS build	17763.1457
Version	1809 (latest updates)

6 – ZVOL PROPERTIES

```
:~# zfs get all tank/onetest
NAME          PROPERTY              VALUE                  SOURCE
tank/onetest  type                  volume                 -
tank/onetest  creation              Thu Dec 10 11:41 2020  -
tank/onetest  used                  9.28G                  -
tank/onetest  available             2.81T                  -
tank/onetest  referenced            9.09G                  -
tank/onetest  compressratio         1.00x                  -
tank/onetest  reservation           none                   default
tank/onetest  volsize               9G                     local
tank/onetest  volblocksize          8K                     default
tank/onetest  checksum              on                     default
tank/onetest  compression           off                    default
tank/onetest  readonly              off                    default
tank/onetest  createtxg             14                     -
tank/onetest  copies                1                      default
tank/onetest  refreservation        9.28G                  local
tank/onetest  guid                  8203829856728317805    -
tank/onetest  primarycache          metadata               local
tank/onetest  secondarycache        all                    default
tank/onetest  usedbysnapshots       0B                     -
tank/onetest  usedbydataset         9.09G                  -
tank/onetest  usedbychildren        0B                     -
tank/onetest  usedbyrefreservation  199M                   -
tank/onetest  logbias               latency                default
tank/onetest  objsetid              135                    -
tank/onetest  dedup                 off                    default
tank/onetest  mlslabel              none                   default
tank/onetest  sync                  standard               default
tank/onetest  refcompressratio      1.00x                  -
tank/onetest  written               9.09G                  -
tank/onetest  logicalused           9.04G                  -
tank/onetest  logicalreferenced     9.04G                  -
tank/onetest  volmode               default                default
tank/onetest  snapshot_limit        none                   default
tank/onetest  snapshot_count        none                   default
tank/onetest  snapdev               hidden                 default
tank/onetest  context               none                   default
tank/onetest  fscontext             none                   default
tank/onetest  defcontext            none                   default
tank/onetest  rootcontext           none                   default
tank/onetest  redundant_metadata    all                    default
tank/onetest  encryption            off                    default
tank/onetest  keylocation           none                   default
tank/onetest  keyformat             none                   default
tank/onetest  pbkdf2iters           0                      default
```

LEARN MORE

Wiki: https://pve.proxmox.com

Community Forums: https://forum.proxmox.com

Bugtracker: https://bugzilla.proxmox.com

Code repository: https://git.proxmox.com

HOW TO BUY

Find an authorized reseller in your area: www.proxmox.com/partners or

Visit the Proxmox Online Shop to purchase a subscription: https://shop.maurer-it.com

SALES AND INQUIRIES

https://www.proxmox.com

Proxmox Customer Portal

https://my.proxmox.com

TRAINING PROXMOX VE

Learn Proxmox VE easily, visit https://www.proxmox.com/training

ABOUT PROXMOX

Proxmox Server Solutions GmbH is a privately held company based in Vienna, Europe.