



StorPool
DISTRIBUTED STORAGE

Achieving the Ultimate Performance with KVM

Boyan Krosnov
SREcon20 Americas

About me

- Chief Product Officer & co-founder of StorPool
- 20+ years in ISPs, SDN, SDS
- IT infrastructure with a focus on inventions, performance and efficiency



Boyan Krosnov

[linkedin.com/in/krosnov/](https://www.linkedin.com/in/krosnov/)

@bkrosnov

bk@storpool.com

About StorPool

- NVMe software-defined storage for VMs and containers
- Scale-out, HA, API-controlled
- Since 2011, in commercial production use since 2013
- Based in Sofia, Bulgaria; Profitable and growing
- Mostly **virtual disks for KVM** and bare metal Linux hosts
- Also used with VMWare, Hyper-V, XenServer
- Integrations into **OpenStack/Cinder, Kubernetes Persistent Volumes,** CloudStack, OpenNebula, OnApp
- Fully managed service; 24/7; monitoring and analytics; 1000s of servers in 100s of zones

Why performance

- Better application performance
 - e.g. time to load a page, time to rebuild, time to execute specific query
- Happier customers in cloud and multi-tenant environments
- ROI, TCO - Lower cost per delivered resource (per VM) through higher density

- For public cloud - win customers over from your competitors
- For private cloud - do more with less; win applications / workloads / teams over from public cloud

Agenda

1. **Hardware**
2. Compute, Virtualization
3. Networking
4. Storage
5. Conclusion

Compute node hardware

Usual optimization goal

- lowest cost per delivered resource
- fixed performance target
- calculate all costs - power, cooling, space, server, network, support/maintenance

Example: cost per VM with 4x dedicated 3 GHz cores and 16 GB RAM

Unusual

- Best single-thread performance I can get at any cost
- 4+ GHz cores, etc.




Compute node hardware

Brand	Model	release date	ark.inte l.com status	release price (\$)	Cores	TDP (W)	All-Core Turbo Clock (GHz)	Selected 1S or 2S or 4S ?	Total \$ per core	Total \$/GHz
Gold	5220R	January 2020	Launched	\$1,555	24	150	2.9	2S	\$244	\$84
Gold	6230R	January 2020	Launched	\$1,894	26	150	3.0	2S	\$244	\$81
Gold	5218R	January 2020	Launched	\$1,273	20	125	2.9	2S	\$257	\$89
Gold	6238R	January 2020	Launched	\$2,612	28	165	3.0	2S	\$262	\$87
Gold	6222V	May 2019	Launched	\$1,600	20	115	2.4	2S	\$271	\$113
Gold	6240R	January 2020	Launched	\$2,200	24	165	3.2	2S	\$276	\$86
Silver	4216	April 2019	Launched	\$1,002	16	100	2.7	2S	\$277	\$103
Gold U	6212U	April 2019	Launched	\$1,450	24	165	3.1	1S	\$285	\$92
Gold	6230	April 2019	Launched	\$1,894	20	125	2.8	2S	\$290	\$103
Gold	6230T	May 2019	Launched	\$1,988	20	125	2.8	2S	\$294	\$105
Gold	5220	April 2019	Launched	\$1,555	18	125	2.7	2S	\$295	\$109
Gold	6230N	May 2019	Launched	\$2,046	20	125	2.9	2S	\$298	\$103
Gold	5220T	May 2019	Launched	\$1,727	18	105	2.7	2S	\$298	\$110
Gold	6262V	May 2019	Launched	\$2,900	24	135	2.5	2S	\$299	\$120
Gold	5218T	May 2019	Launched	\$1,349	16	105	2.7	2S	\$303	\$112

Compute node hardware

Intel

lowest cost per core:

2.9 GHz: Xeon Gold 5220R - 24 cores (\$244/core)		13%
3.2 GHz: Xeon Gold 6240R - 24 cores (\$276/core)		
3.6 GHz: Xeon Gold 6248R - 24 cores (\$308/core)		12%

lowest cost per GHz:

- Xeon Gold 6230R - 26 cores @ 3.0 GHz (\$81/GHz)

Compute node hardware

AMD

per core @2.5 GHz: AMD EPYC 7702P 1-socket 64 cores (\$210/core)

per core @3.0 GHz: AMD EPYC 7502 2-socket 32 cores (\$251/core)

per core @3.5 GHz: AMD EPYC 7F72 2-socket 24 cores (\$320/core)

per GHz: 7502, 7702P, 7742 tied for first place

Compute node hardware

Form factor

from



to



Compute node hardware

- firmware versions and BIOS settings
- Understand power management -- esp. C-states, P-states, HWP and "bias"
 - Different on AMD EPYC: "power-deterministic", "performance-deterministic"
- Think of rack level optimization - how do we get the lowest total cost per delivered resource?

Agenda

1. Hardware
2. **Compute, Virtualization**
3. Networking
4. Storage
5. Conclusion

Tuning KVM

RHEL7 Virtualization_Tuning_and_Optimization_Guide [link](#)

https://pve.proxmox.com/wiki/Performance_Tweaks

https://events.static.linuxfound.org/sites/events/files/slides/CloudOpen2013_Khoa_Huynh_v3.pdf

<http://www.linux-kvm.org/images/f/f9/2012-forum-virtio-blk-performance-improvement.pdf>

<http://www.slideshare.net/janghoonsim/kvm-performance-optimization-for-ubuntu>

... but don't trust everything you read. Perform your own benchmarking!

CPU and Memory

Recent Linux kernel, KVM and QEMU

... but beware of the bleeding edge

E.g. qemu-kvm-ev from RHEV (repackaged by CentOS)

tuned-adm virtual-host

tuned-adm virtual-guest

CPU

Typical

- (heavy) oversubscription, because VMs are mostly idling
- HT
- NUMA
- route IRQs of network and storage adapters to a core on the NUMA node they are on

Unusual

- CPU Pinning

Understanding oversubscription and congestion

Linux scheduler statistics: `/proc/schedstat`
(`linux-stable/Documentation/scheduler/sched-stats.txt`)

Next three are statistics describing scheduling latency:

7) sum of all time spent running by tasks on this processor (in ms)

8) sum of all time spent waiting to run by tasks on this processor (in ms)

9) # of tasks (not necessarily unique) given to the processor

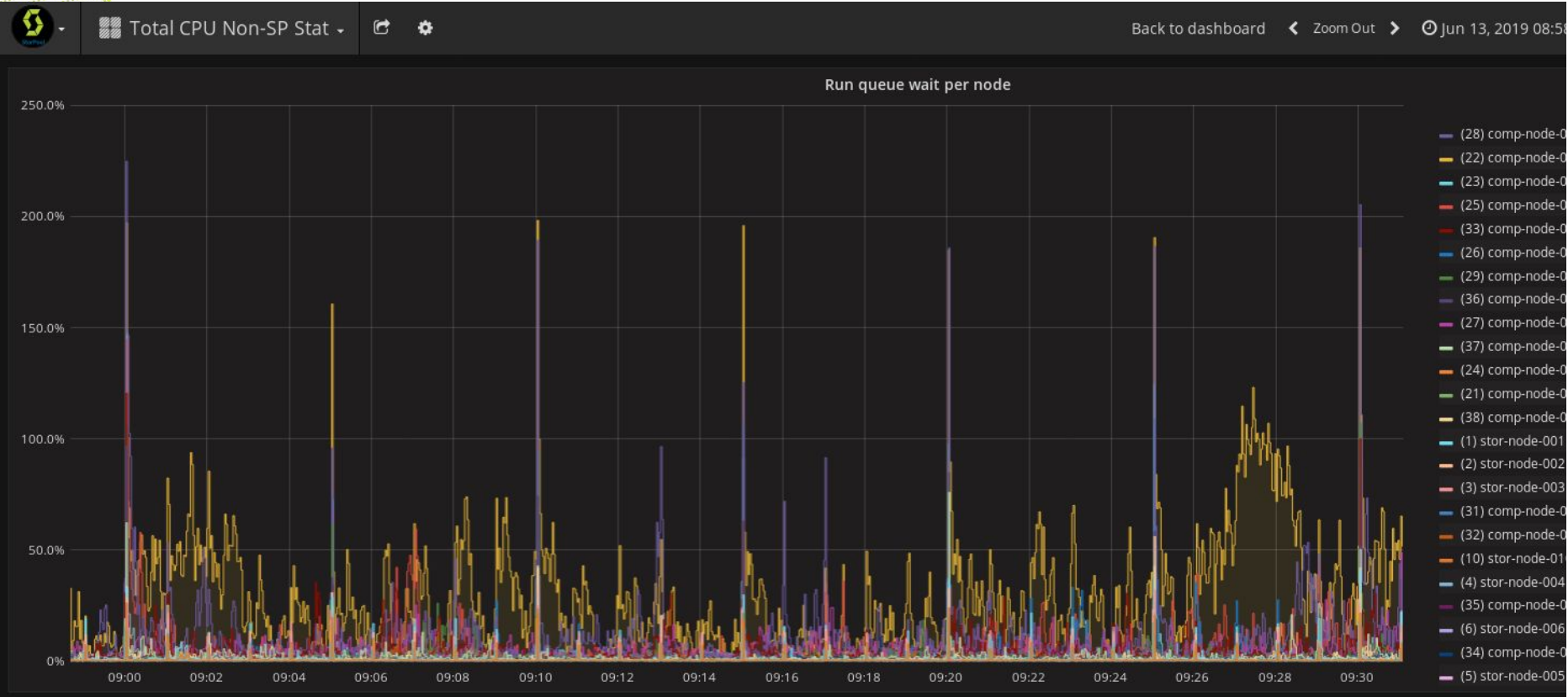
* In nanoseconds, not ms.

20% CPU load with large wait time (bursty congestion) is possible

100% CPU load with no wait time, also possible

Measure CPU congestion!

Understanding oversubscription and congestion



Memory

Typical

- Dedicated RAM
- huge pages, THP
- NUMA
- use local-node memory if you can

Unusual

- Oversubscribed RAM
- balloon
- KSM (RAM dedup)

Agenda

1. Hardware
2. Compute, Virtualization
3. **Networking**
4. Storage
5. Conclusion

Networking

Virtualized networking

- hardware emulation (rtl8139, e1000)
- paravirtualized drivers - virtio-net

regular virtio vs vhost-net vs vhost-user

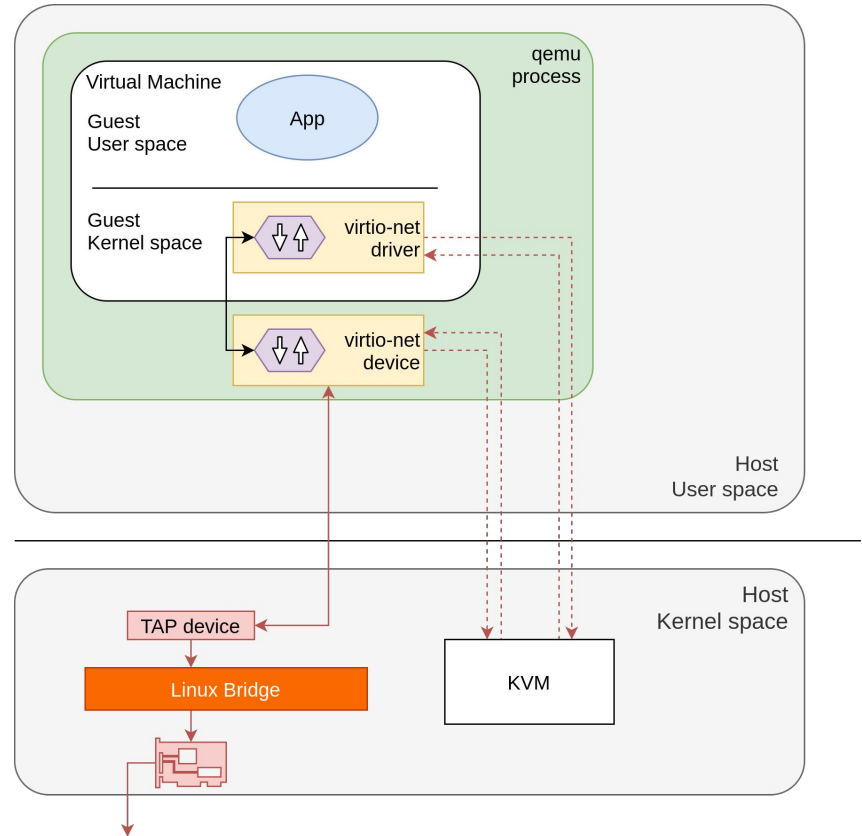
Linux Bridge vs OVS in-kernel vs OVS-DPDK

Pass-through networking

SR-IOV (PCIe pass-through)

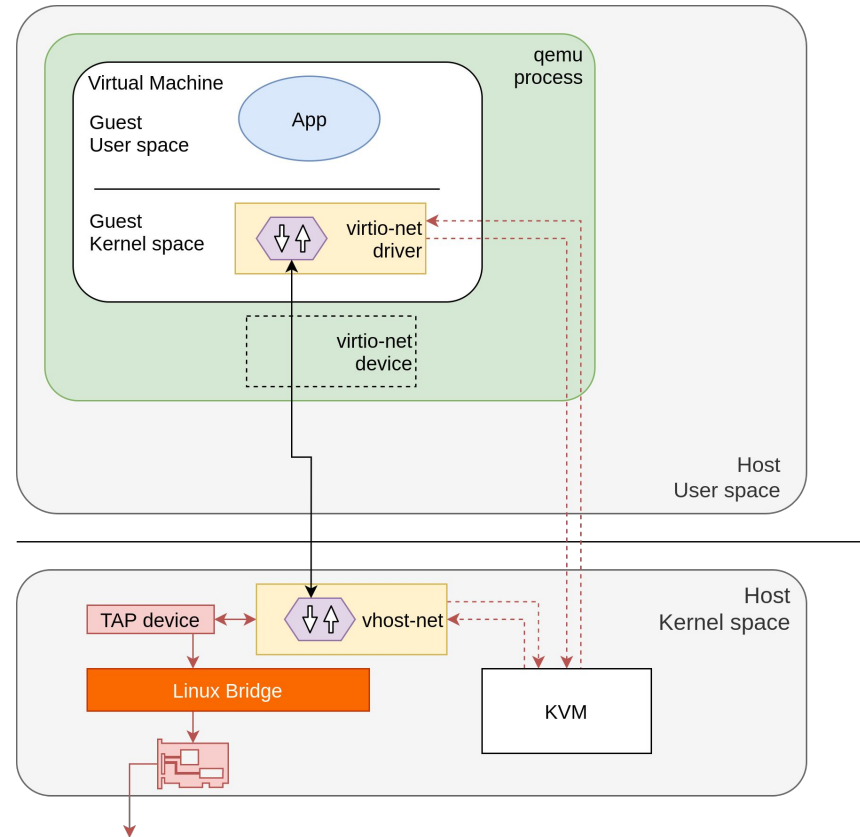
virtio-net QEMU

- Multiple context switches:
 1. virtio-net driver → KVM
 2. KVM → qemu/virtio-net device
 3. qemu → TAP device
 4. qemu → KVM (notification)
 5. KVM → virtio-net driver (interrupt)
- Much more efficient than emulated hardware
- shared memory with qemu process
- qemu thread process packets



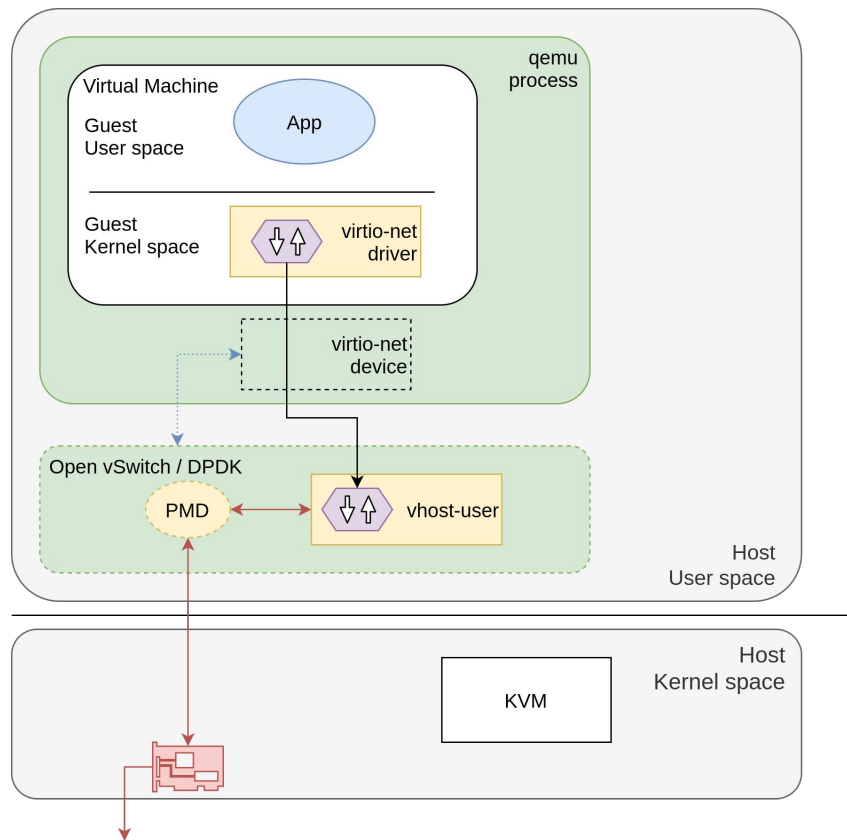
virtio vhost-net

- Two context switches (optional):
 1. virtio-net driver → KVM
 2. KVM → virtio-net driver (interrupt)
- shared memory with the host kernel (vhost protocol)
- Allows Linux Bridge Zero Copy
- qemu / virtio-net device is on the control path only
- kernel thread [vhost] process packets



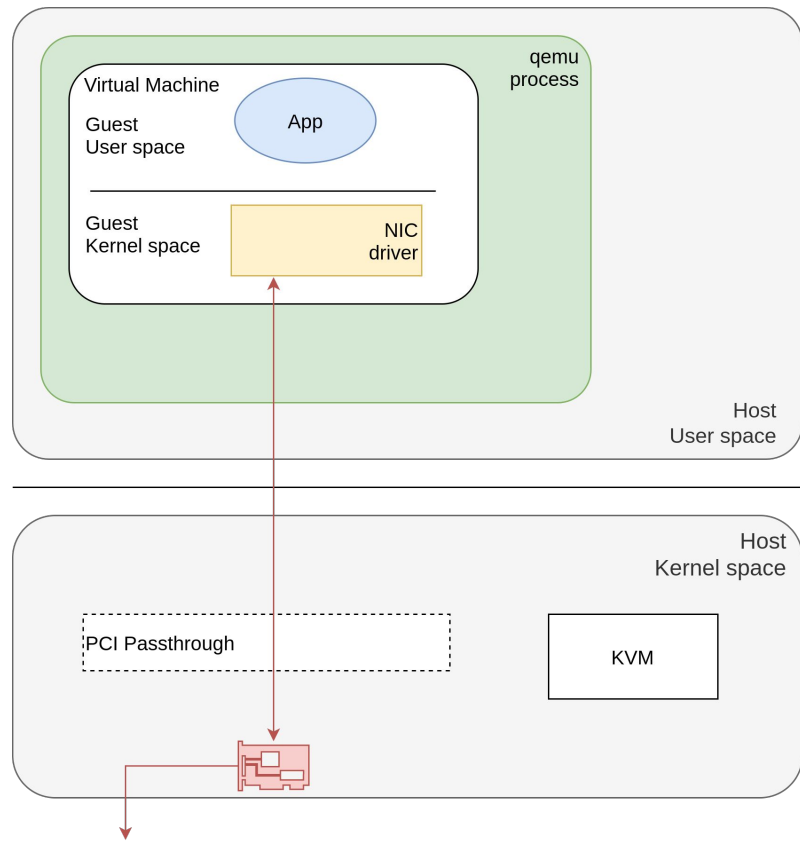
virtio vhost-usr / OVS-DPDK

- No context switches
- shared memory between the guest and the Open vSwitch (requires huge pages)
- Zero copy
- qemu / virtio-net device is on the control path only
- KVM not in the path
- ovs-vswitchd process packets.
- Poll-mode-driver (PMD) takes 1 CPU core, 100%

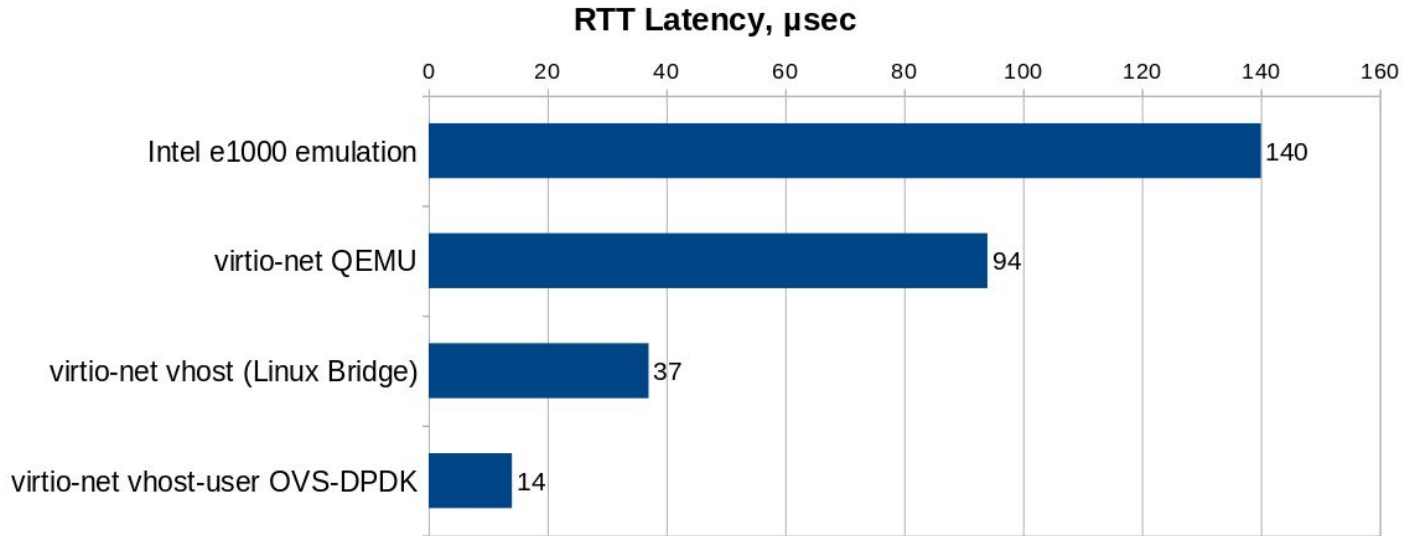


PCI Passthrough

- No paravirtualized devices
- Direct access from the guest kernel to the PCI device
- Host, KVM and qemu are not on the data nor the control path.
- NIC driver in the guest
- No virtual networking
- No live migrations
- No filtering
- No control
- Shared devices via SR-IOV



Virtual Network Performance

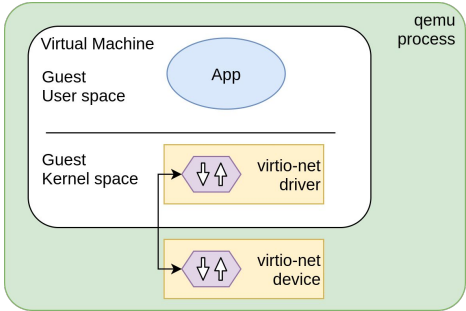


All measurements are between two VMs on the same host
ping -f -c 100000 vm2

virtio-net QEMU

qemu pid

```
# top -H -p 14826
```



```
top - 01:01:31 up 15 days, 21:48, 2 users, load average: 7.28, 6.03, 5.98
Threads: 6 total, 2 running, 4 sleeping, 0 stopped, 0 zombie
%Cpu(s): 16.9 us, 9.5 sy, 0.0 ni, 73.6 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
KiB Mem : 13175620+total, 1550248 free, 12736860+used, 2837352 buff/cache
KiB Swap: 0 total, 0 free, 0 used. 2975956 avail Mem
```

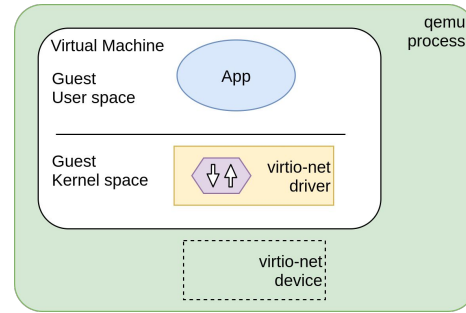
PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	P	COMMAND
14852	oneadmin	20	0	2994152	64440	19700	S	0.0	0.0	0:00.00	28	vnc_worker
14849	oneadmin	20	0	2994152	64440	19700	R	40.0	0.0	0:35.98	18	CPU 1/KVM
14847	oneadmin	20	0	2994152	64440	19700	S	6.0	0.0	0:09.50	31	CPU 0/KVM
14834	oneadmin	20	0	2994152	64440	19700	S	0.0	0.0	0:00.11	30	IO iothread1
14833	oneadmin	20	0	2994152	64440	19700	S	0.0	0.0	0:00.01	9	qemu-kvm
14826	oneadmin	20	0	2994152	64440	19700	R	99.9	0.0	1:09.07	12	qemu-kvm

virtio vhost-net

qemu

vhost thread

```
# top -H -p 18225 -p 18241
```



```
top - 01:09:50 up 15 days, 21:57, 2 users, load average: 7.84, 7.14, 6.62
Threads: 7 total, 2 running, 5 sleeping, 0 stopped, 0 zombie
%Cpu(s): 17.0 us, 10.0 sy, 0.0 ni, 73.0 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
KiB Mem : 13175620+total, 1545220 free, 12737219+used, 2838800 buff/cache
KiB Swap: 0 total, 0 free, 0 used. 2972400 avail Mem
```

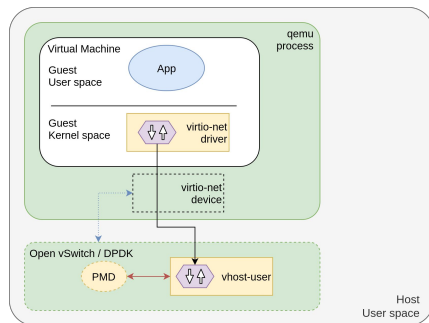
PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	P	COMMAND
18225	oneadmin	20	0	3000308	64260	19524	S	0.0	0.0	0:00.16	22	qemu-kvm
18232	oneadmin	20	0	3000308	64260	19524	S	0.0	0.0	0:00.00	12	qemu-kvm
18234	oneadmin	20	0	3000308	64260	19524	S	0.0	0.0	0:00.13	22	IO iothread1
18241	root	20	0	0	0	0	R	93.7	0.0	1:09.34	30	vhost-18225
18248	oneadmin	20	0	3000308	64260	19524	S	0.7	0.0	0:10.92	23	CPU 0/KVM
18250	oneadmin	20	0	3000308	64260	19524	R	65.4	0.0	0:53.86	13	CPU 1/KVM
18253	oneadmin	20	0	3000308	64260	19524	S	0.0	0.0	0:00.00	21	vnc_worker

virtio vhost-usr / OVS-DPDK

qemu

OVS

```
# top -H -p 18225 -p 27142
```



```
top - 01:19:59 up 15 days, 22:07, 2 users, load average: 4.18, 5.42, 6.22
Threads: 17 total, 2 running, 15 sleeping, 0 stopped, 0 zombie
%Cpu(s): 8.8 us, 3.8 sy, 0.0 ni, 87.4 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
KiB Mem : 13175620+total, 1551636 free, 12736377+used, 2840804 buff/cache
KiB Swap: 0 total, 0 free, 0 used. 2980800 avail Mem
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	P	COMMAND
27157	root	10	-10	18.982g	151148	15868	S	0.0	0.1	0:01.61	5	ovs-vswitchd
27156	root	10	-10	18.982g	151148	15868	S	0.0	0.1	0:02.79	5	ovs-vswitchd
23318	root	10	-10	18.982g	151148	15868	R	99.9	0.1	2845:43	6	pmd21
18253	oneadmin	20	0	3000308	64376	19524	S	0.0	0.0	0:00.00	21	vnc_worker
18250	oneadmin	20	0	3000308	64376	19524	S	1.0	0.0	3:53.02	18	CPU 1/KVM
18248	oneadmin	20	0	3000308	64376	19524	R	99.9	0.0	6:40.92	31	CPU 0/KVM
18234	oneadmin	20	0	3000308	64376	19524	S	0.0	0.0	0:00.13	7	IO iotthread1
18232	oneadmin	20	0	3000308	64376	19524	S	0.0	0.0	0:00.00	12	qemu-kvm
18225	oneadmin	20	0	3000308	64376	19524	S	0.0	0.0	0:00.25	14	qemu-kvm

Additional reading

- Deep dive into Virtio-networking and vhost-net
<https://www.redhat.com/en/blog/deep-dive-virtio-networking-and-vhost-net>
- Open vSwitch DPDK support
<https://docs.openvswitch.org/en/latest/topics/dpdk/>

Agenda

1. Hardware
2. Compute, Virtualization
3. Networking
4. **Storage**
5. Conclusion

Storage - virtualization

Virtualized

live migration

thin provisioning, snapshots, etc.

vs. Full bypass

only speed

Storage - virtualization

Virtualized

cache=none -- direct IO, bypass host buffer cache

io=native -- use Linux Native AIO, not POSIX AIO (threads)

virtio-blk vs virtio-scsi

virtio-scsi multiqueue

iothread

vs. Full bypass

SR-IOV for NVMe devices

Storage - virtualization

Virtualized with io_uring

guest kernel -> qemu ----(Linux Native AIO)---> host kernel

guest kernel -> qemu ----(io_uring)---> host kernel

Virtualized with io_uring passthrough

guest kernel ----(io_uring)---> host kernel

Storage - vhost-user

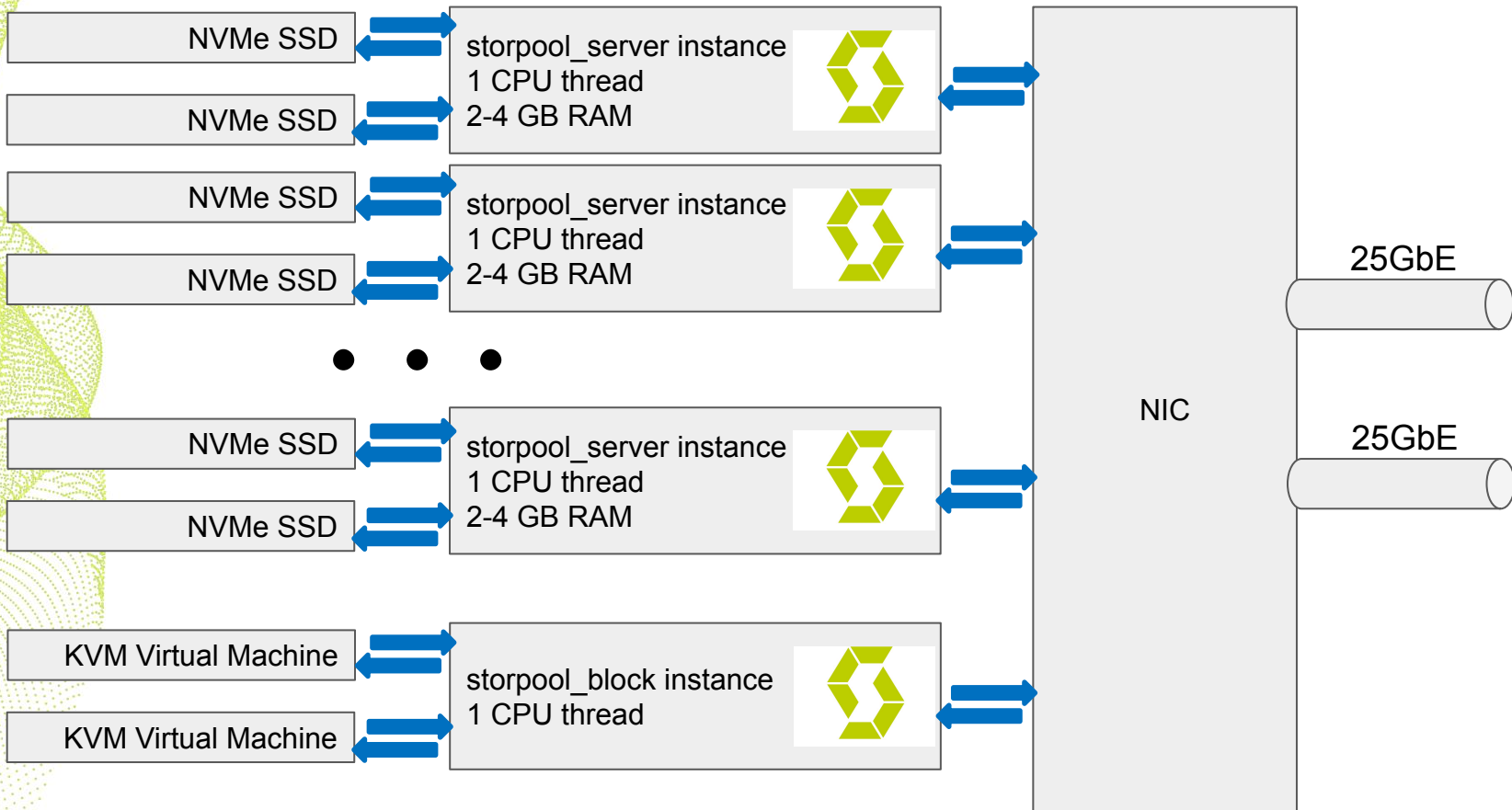
Virtualized with qemu bypass

before:

guest kernel -> host kernel -> qemu -> host kernel -> storage client

with vhost-user:

guest kernel -> storage client

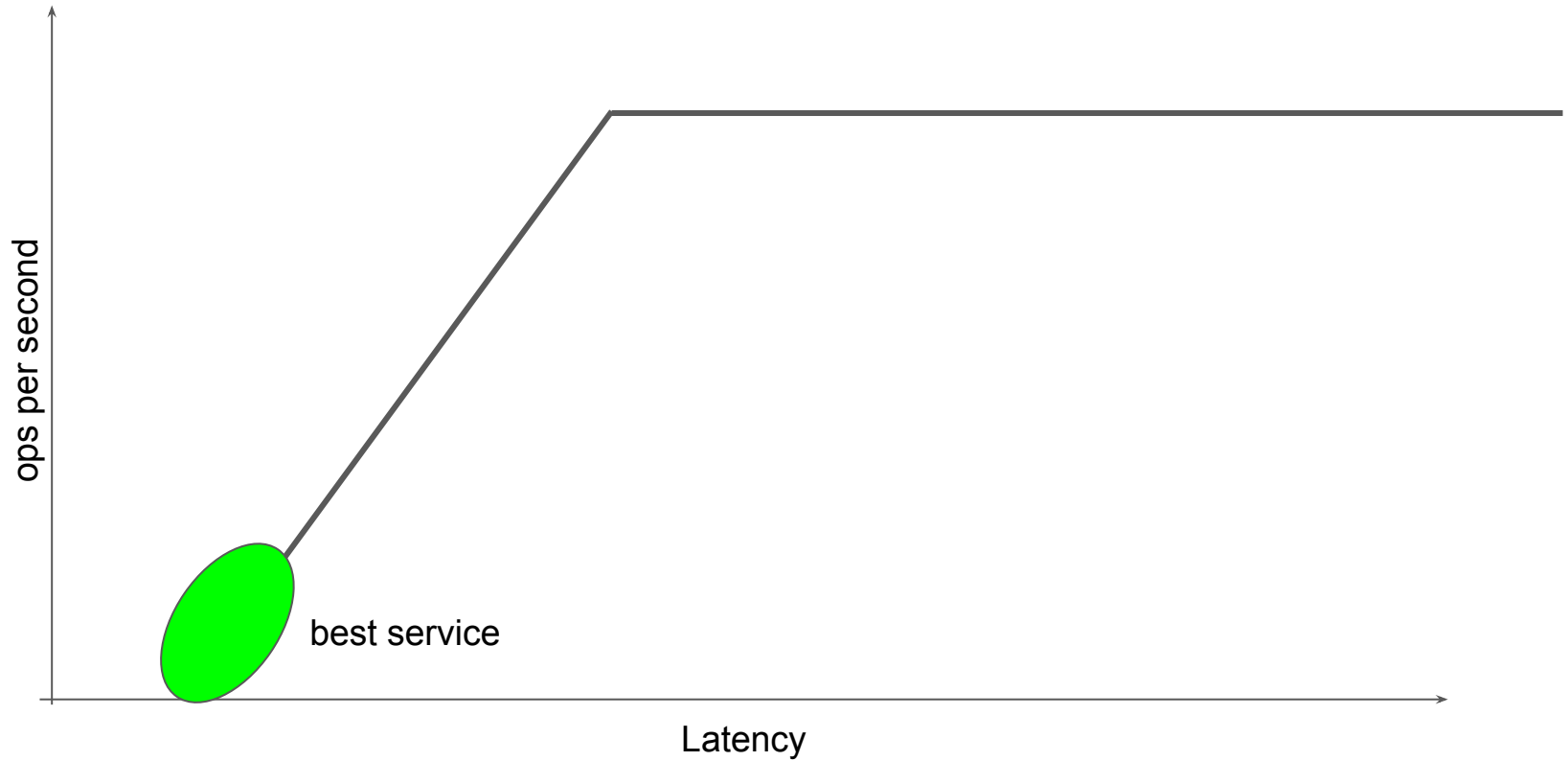


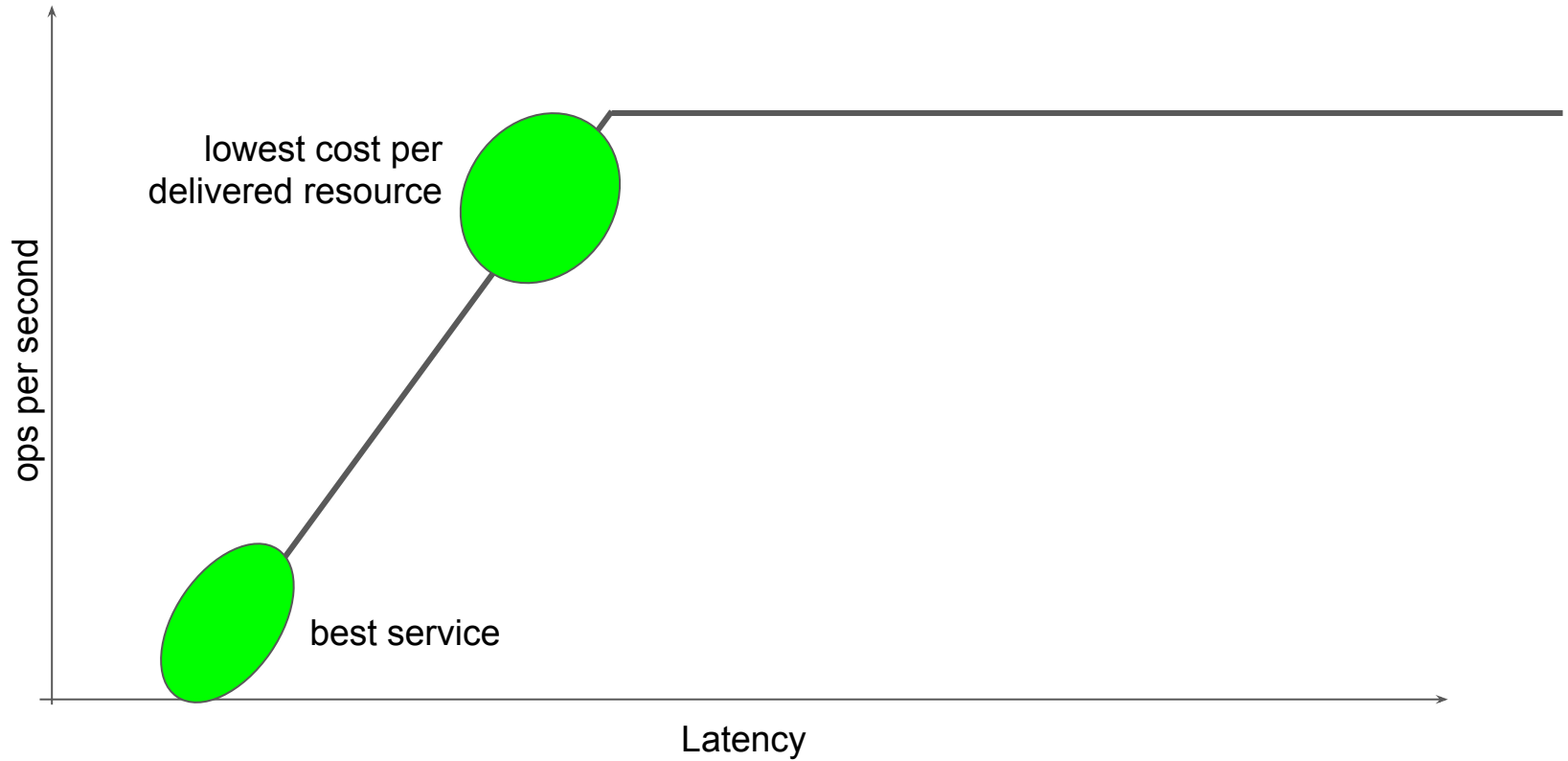
- Highly scalable and efficient architecture
- Scales up in each storage node & out with multiple nodes

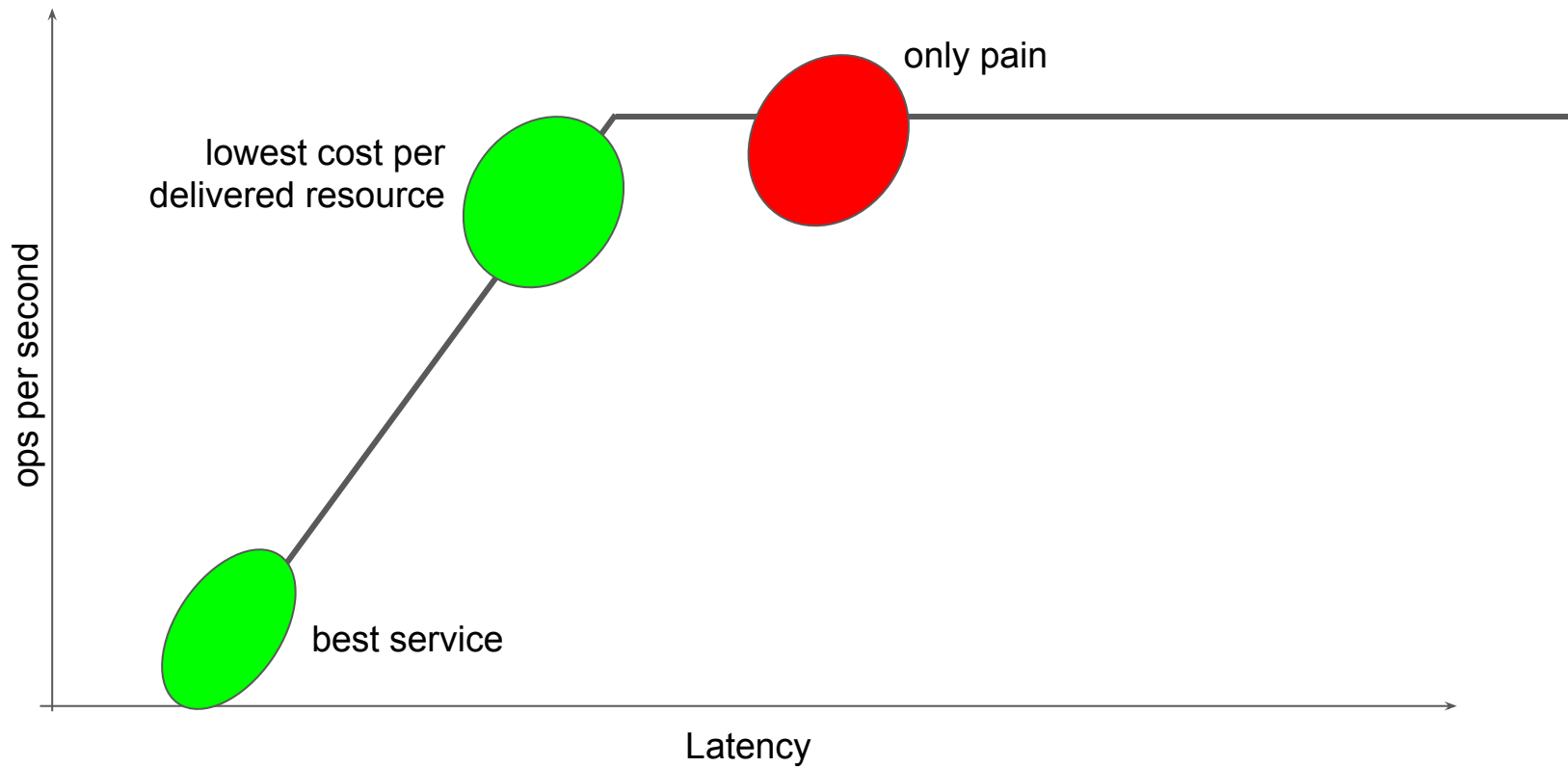
Storage benchmarks

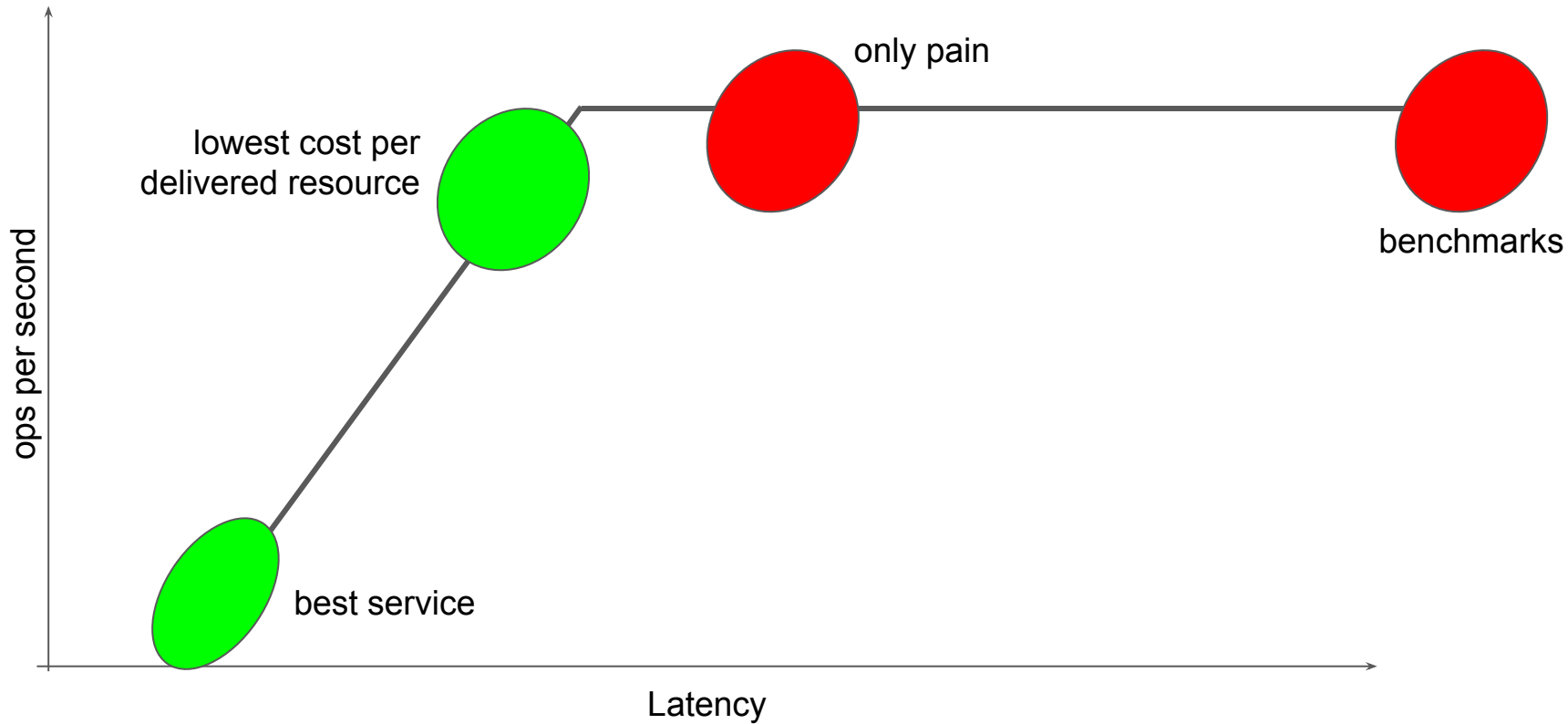
Beware: lots of snake oil out there!

- performance numbers from hardware configurations totally unlike what you'd use in production
- synthetic tests with high iodepth - 10 nodes, 10 workloads * iodepth 256 each. (because why not)
- testing with ramdisk backend
- synthetic workloads don't approximate real world (example)

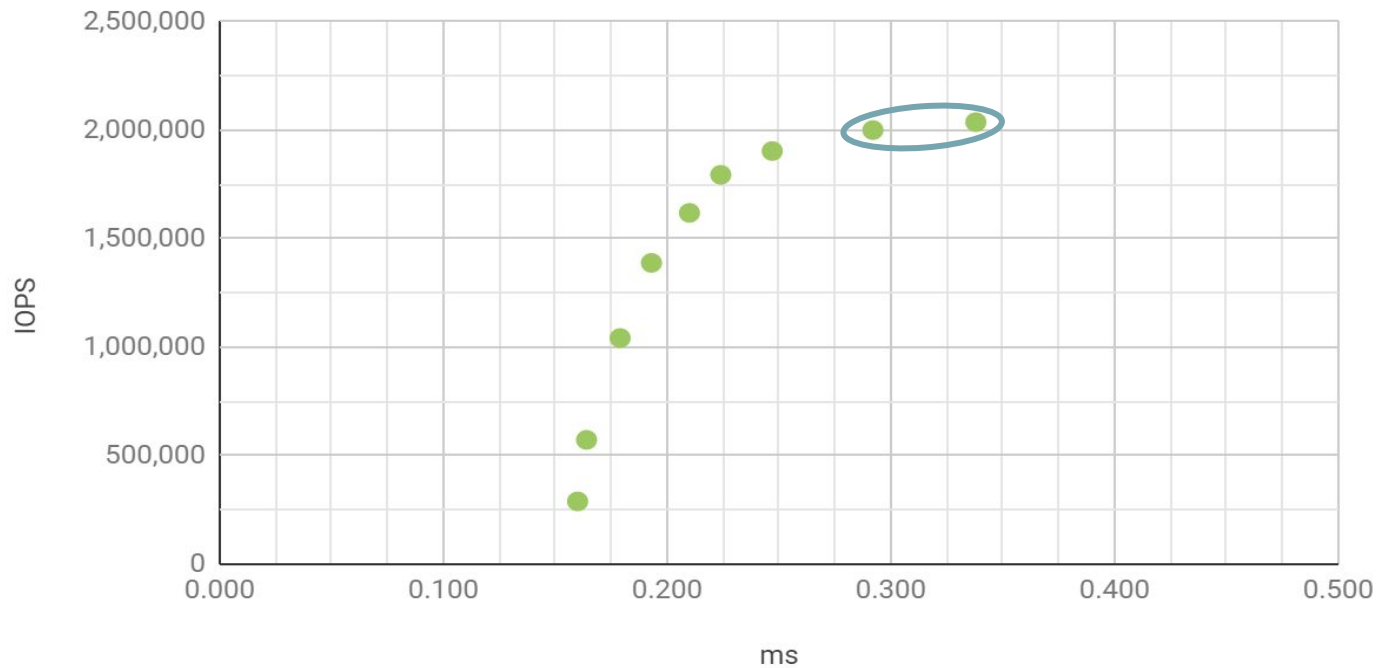




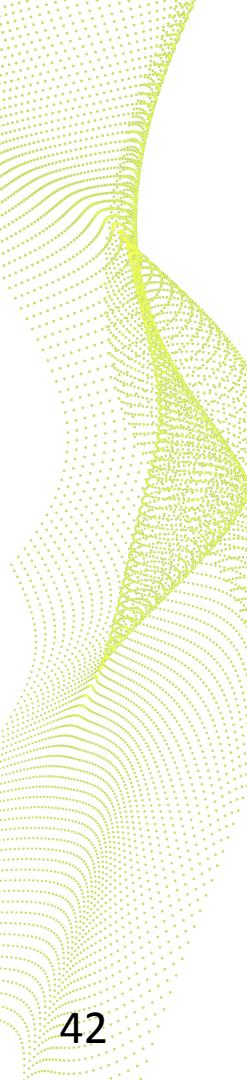




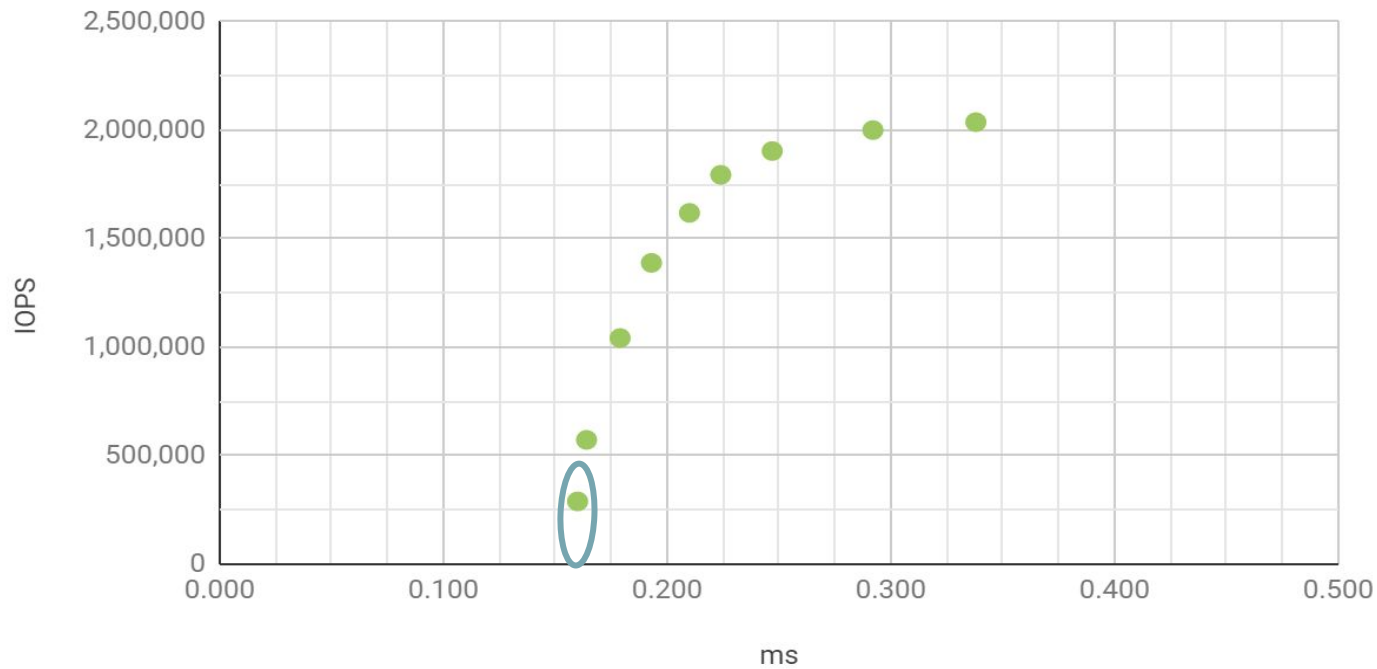
IOPS vs. ms



Benchmarks

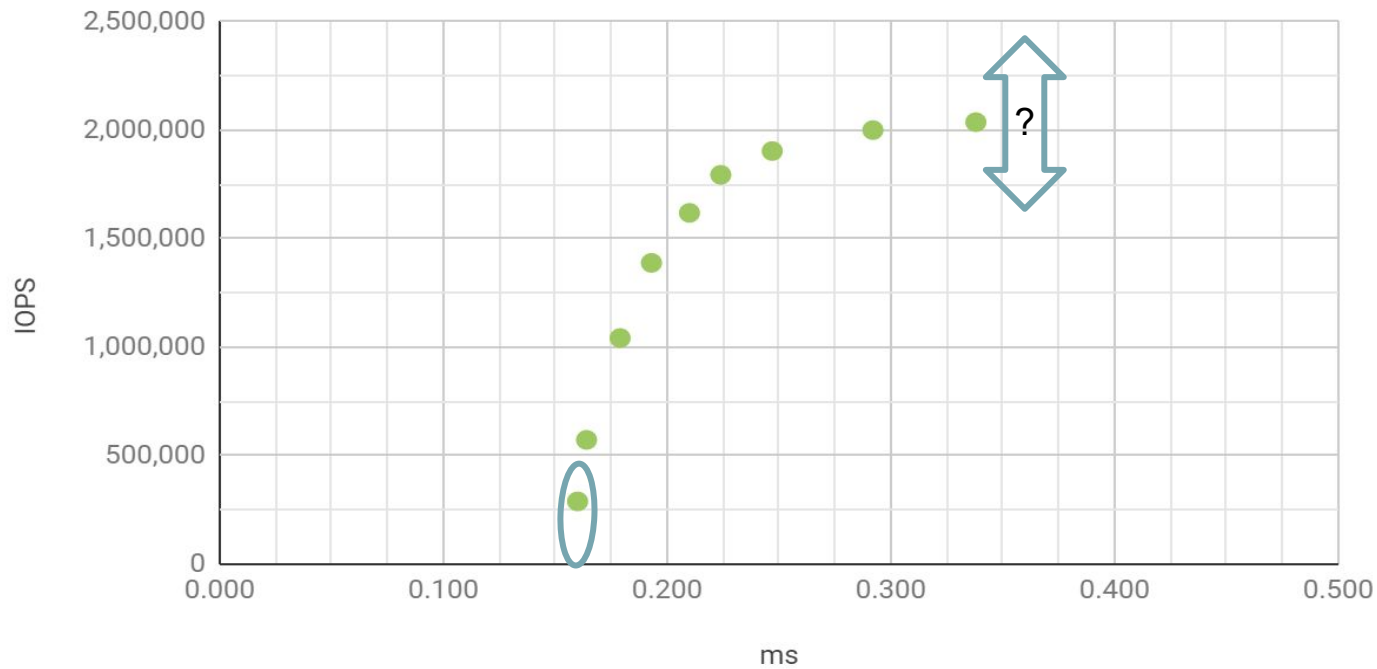


IOPS vs. ms

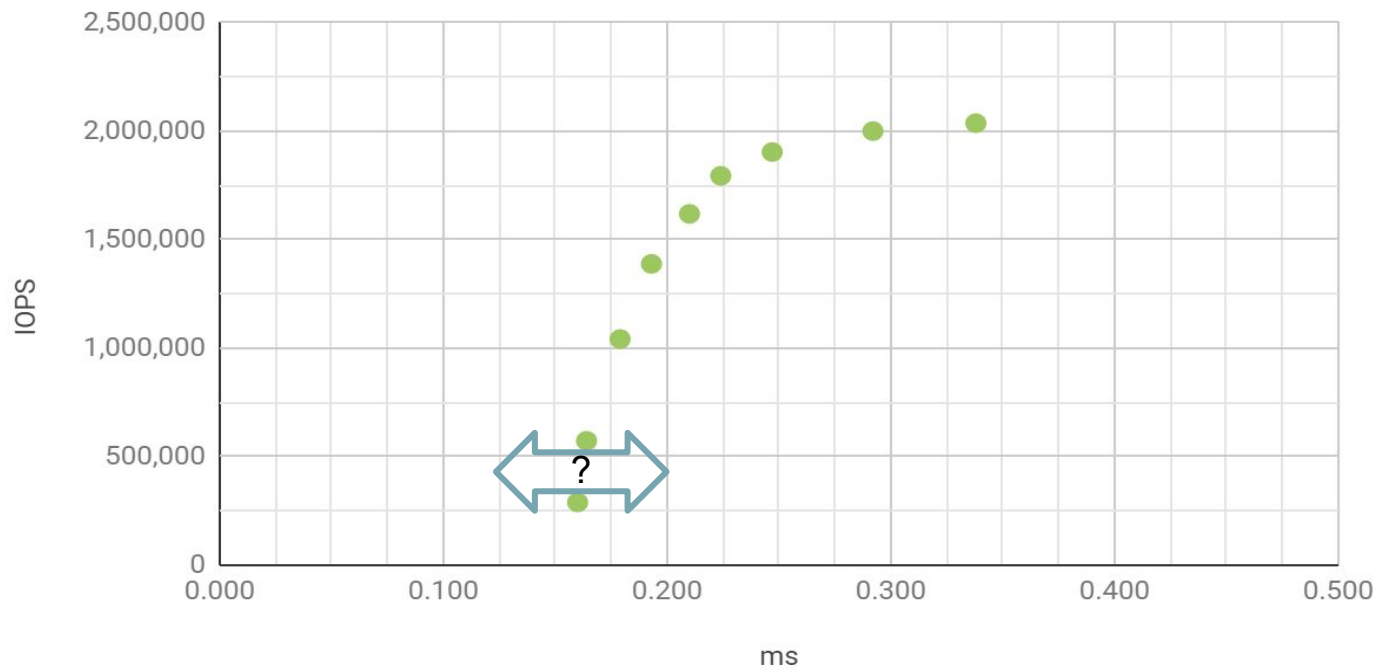


Real load

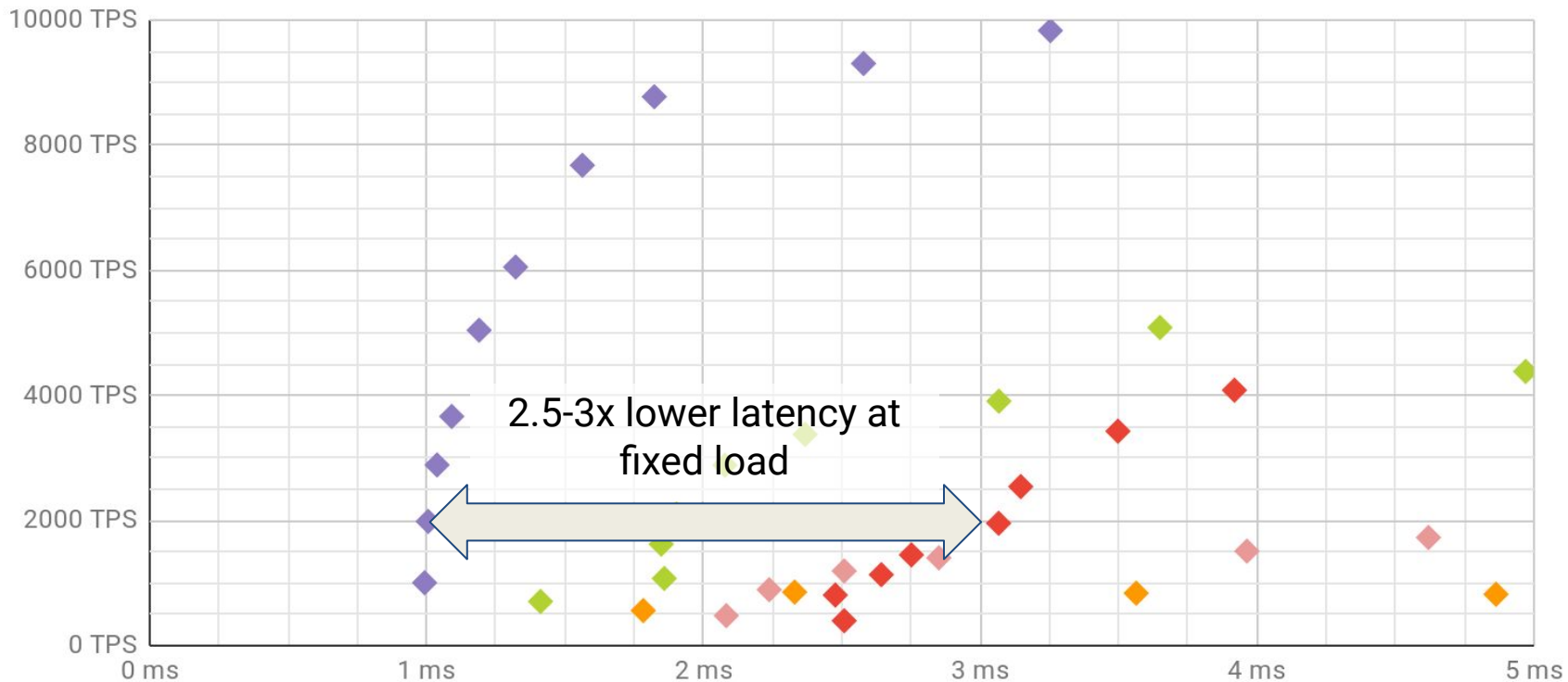
IOPS vs. ms



IOPS vs. ms



pgbench 4x RAM - TPS vs average latency



2.5-3x lower latency at fixed load



Agenda

1. Hardware
2. Compute, Virtualization
3. Networking
4. Storage
5. **Conclusion**

Conclusion

KVM has the right tools to get very good performance, but not by default.

These are complex systems so don't guess, measure!
Measure what matters to your team/company.

Work with partners who understand performance, because you can gain a lot!

Follow StorPool Online



@storpool



StorPool
Storage



StorPool
Storage



StorPool
Storage



Medium

StorPool
Storage



StorPool
Storage



StorPool
DISTRIBUTED STORAGE

Thank you!

Boyan Krosnov
bk@storpool.com

www.storpool.com
@storpool