Ceph SPDK NVMe-oF Gateway Evaluation on openEuler

Xinliang Liu, Senior Engineer, Server, Linaro – sig-SDS

openEuler sig-SDS Meetup 2024 October

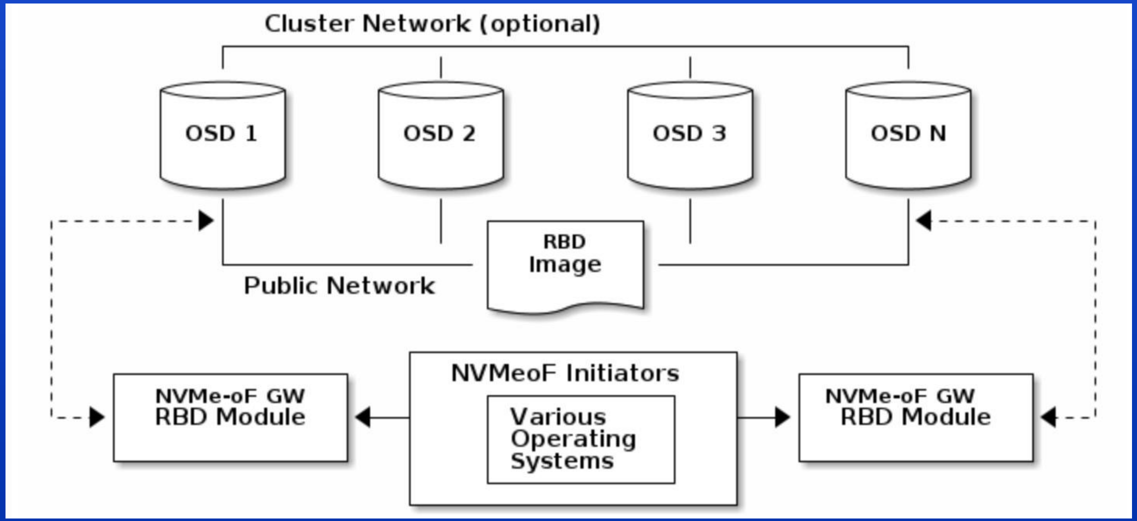
Agenda

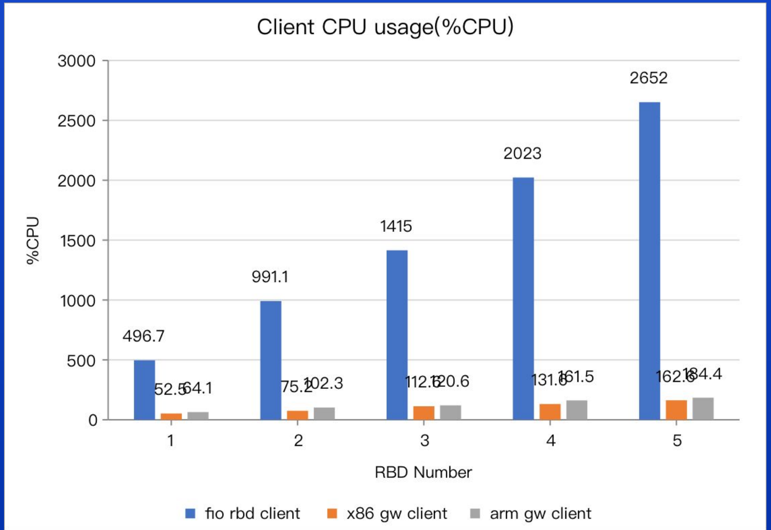
- Ceph SPDK NVMe-oF Gateway介绍
- Ceph SPDK NVMe-oF Gateway Arm上游支持
- Ceph SPDK NVMe-oF Gateway性能测试

Ceph SPDK NVMe-oF Gateway介绍

Ceph SPDK NVMe-oF Gateway介绍 - Overview

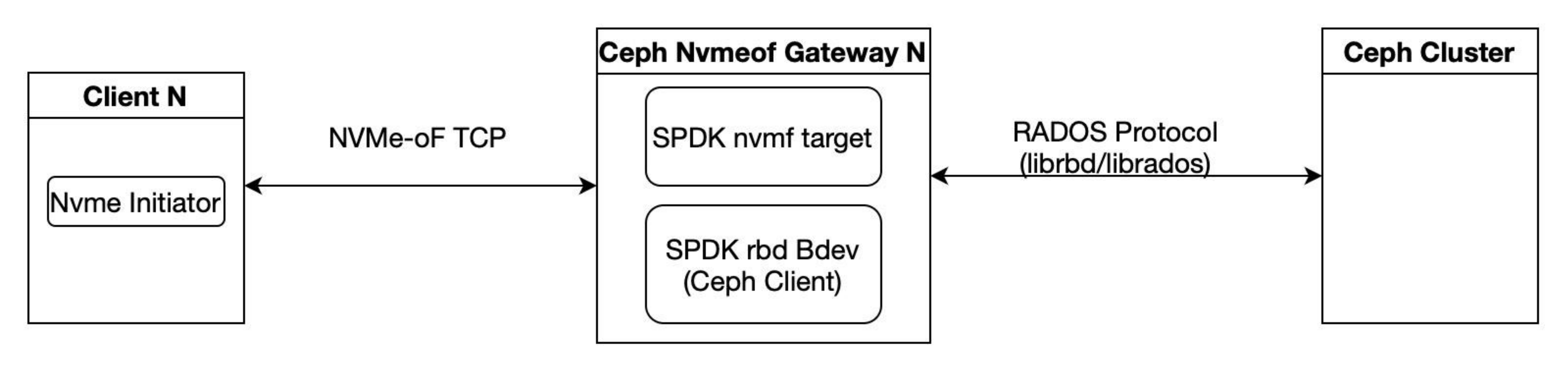
- What and why is NVMe-oF Gateway?
  - Replace Ceph-iSCSI, convert RBD to NVMe-of Disk gateway
  - NVMe-oF SAN similar network disk solution
  - Very low client %CPU，move ceph client to gw node, more generic network storage protocol.
- Usecase
  - Possibly all the SAN and NVMe-oF useacases, requires client support nvme initiator
  - For no ceph client support environment
  - VMware Virtual Machine, bare metal boot volumes, Virtual Desktop
  - PDUs and hardware accelerators
  - Databases and Analytics, AI etc.
- References
  - https://www.spiceworks.com/tech/networking/articles/storage-area-network/
  - https://www.starwindsoftware.com/blog/what-is-nvme-of-nvme-over-fabrics/





Ceph SPDK NVMe-oF Gateway介绍 - Overview

- The NVMe-oF Gateway presents an NVMe-oF target that exports RADOS Block Device (RBD) images as NVMe namespaces.
- The NVMe-oF protocol allows clients (initiators) to send NVMe commands to storage devices (targets) over a TCP/IP network.
- Enabling clients without native Ceph client support to access Ceph block storage.
- More details see: https://docs.ceph.com/en/latest/rbd/nvmeof-overview/



Client N

Nvme Initiator

NVMe-oF TCP

Ceph Nvmeof Gateway N

SPDK nvmf target

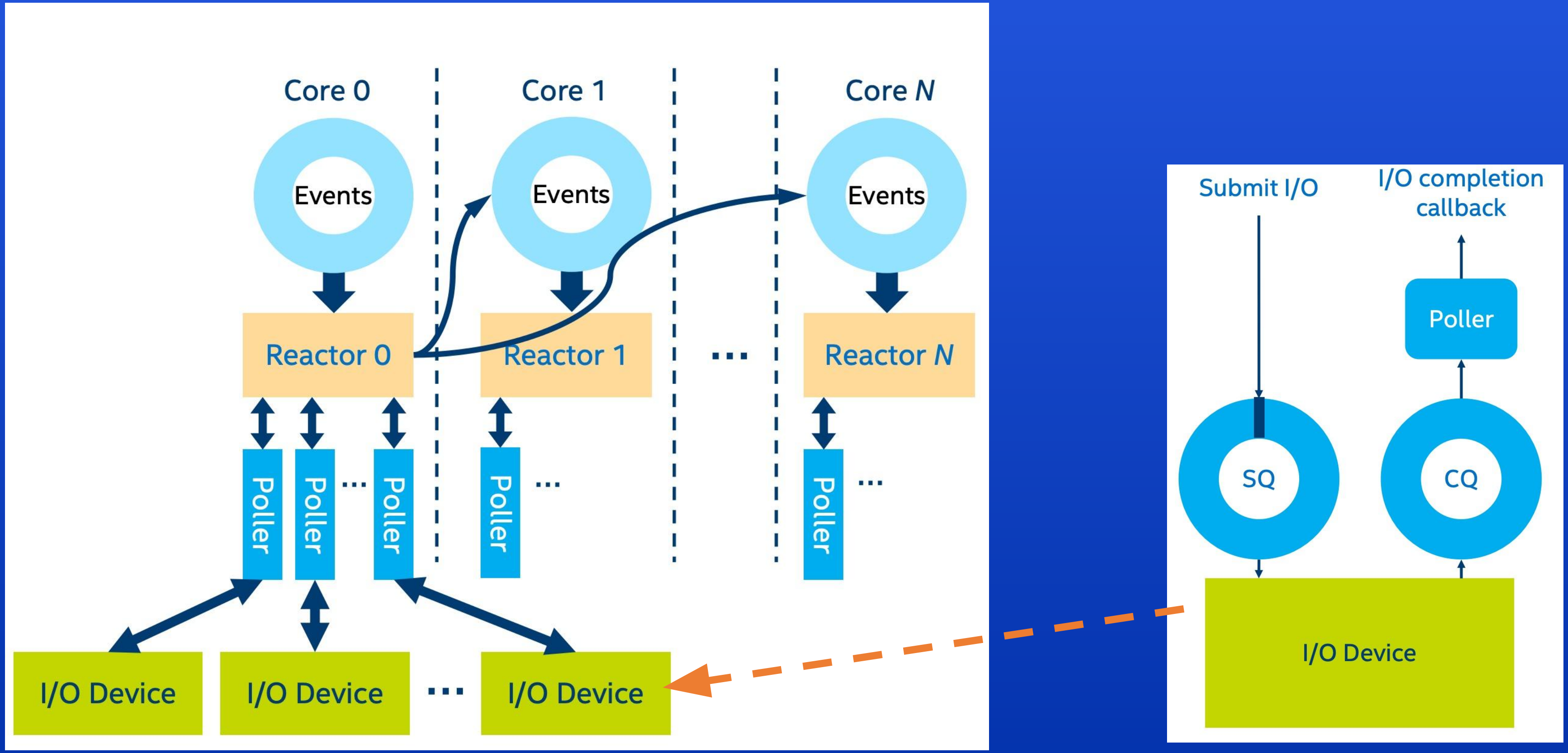
SPDK rbd Bdev (Ceph Client)

RADOS Protocol (librbd/librados)

Ceph Cluster

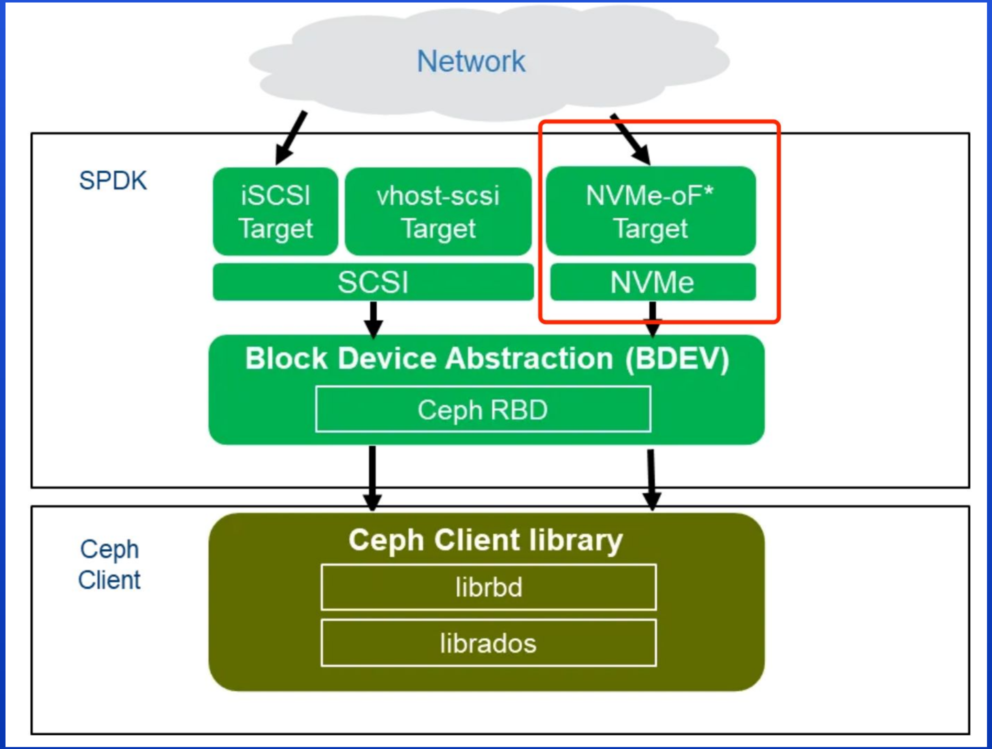
Ceph SPDK NVMe-oF Gateway介绍 - Overview

- SPDK overview
- References:
  - https://spdk.io/doc/overview.html
  - https://www.cnblogs.com/yi-mu-xi/p/12821103.html



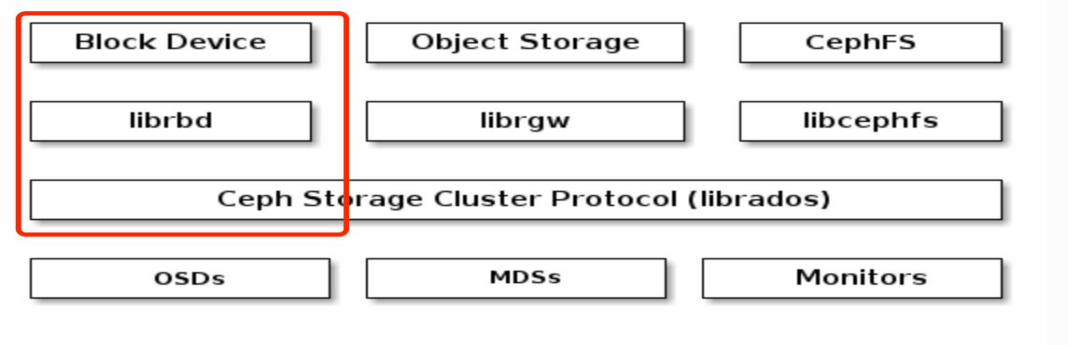
Ceph SPDK NVMe-oF Gateway介绍 - SPDK RBD Bdev

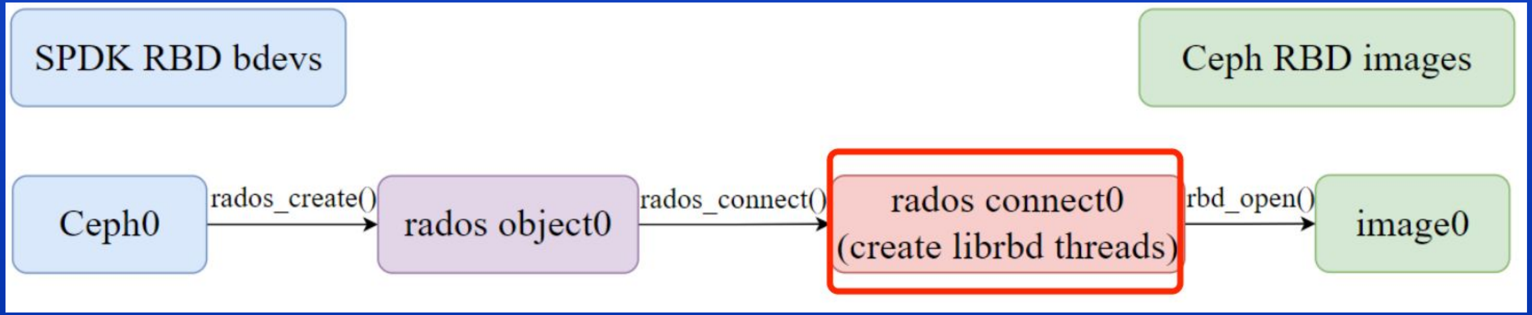
- Convert Ceph RBD to SPDK bdev.
- Allow export to iSCSI/vhost-user/NVMe-oF targets.
- Reference: 管中窥豹SPDK RBD bdev 模块



Ceph SPDK NVMe-oF Gateway介绍 - Ceph rbd client

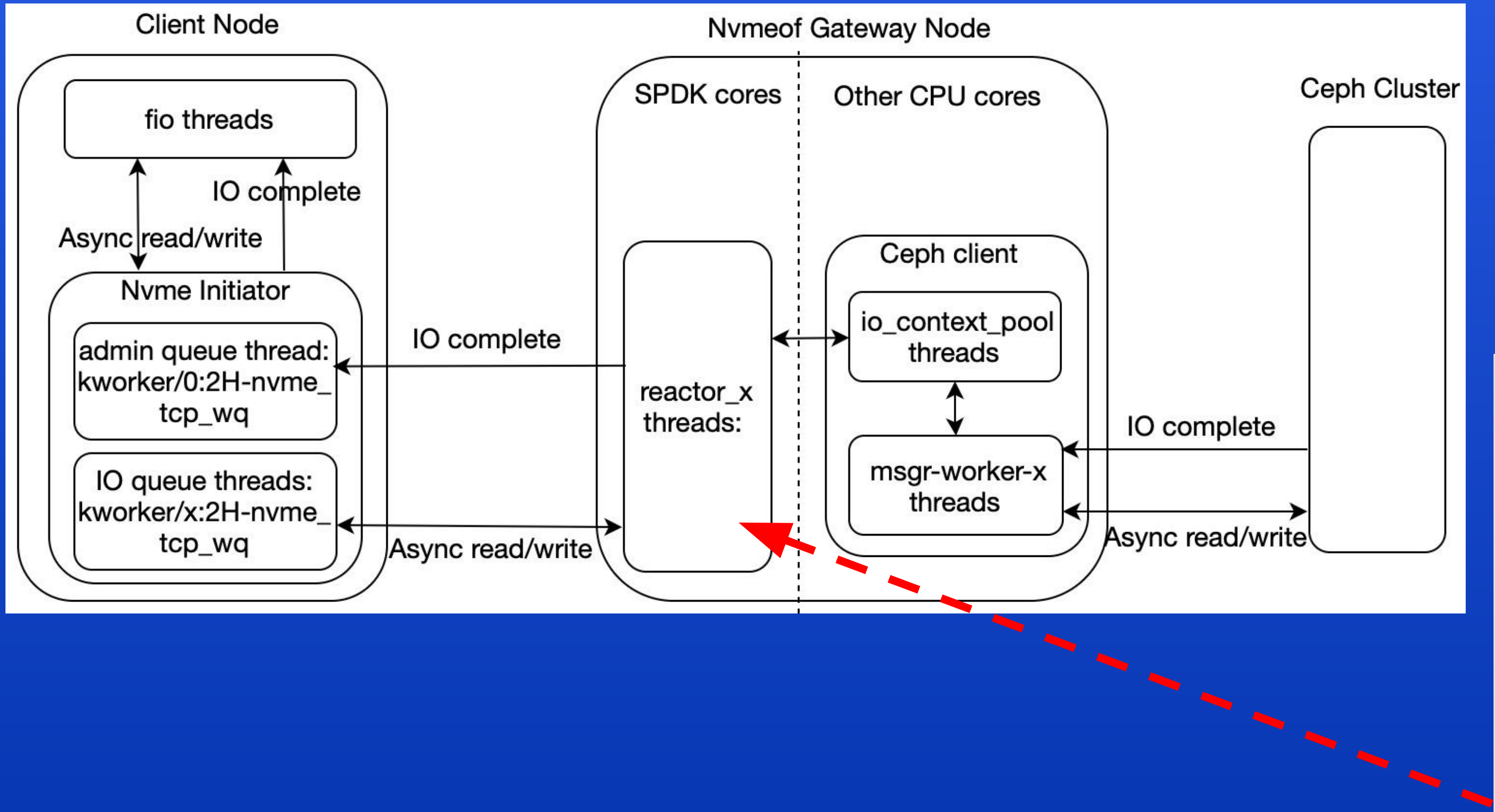
1. Rados cluster object create: rados_create()
2. Connect to cluster: rados_connect()
   - ○ Create io_context_pool threads: io prepare and put to msg queues
   - ○ Create msgr-worker-x threads: msg send/recive
3. Rbd io/image ctx open: rados_ioctx_create(), rbd_open()
4. Rbd image async read/write: rbd_aio_read/readv/write/writev/flush()
5. More details see: https://github.com/spdk/spdk/blob/master/module/bdev/rbd/bdev_rbd.c

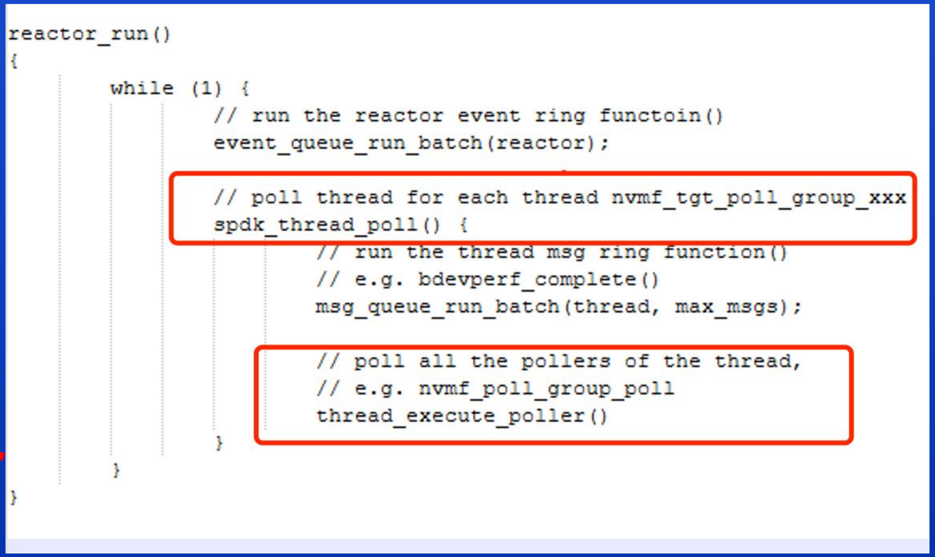




Ceph SPDK NVMe-oF Gateway介绍 - thread model

Nvmeof Gateway whole thread model

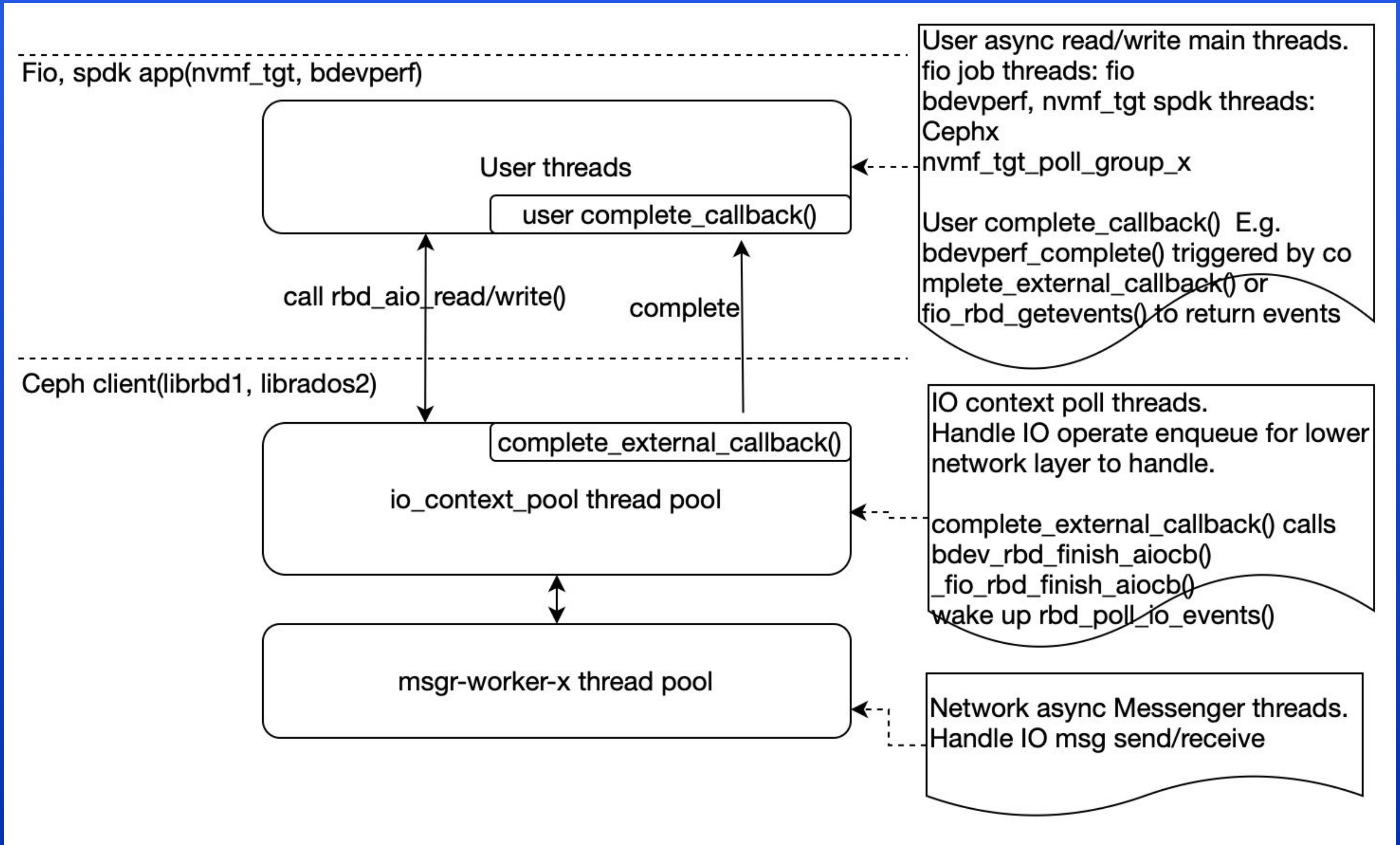




```
reactor_run()
{
        while (1) {
                // run the reactor event ring functoin()
                event_queue_run_batch(reactor);

                // poll thread for each thread nvmf_tgt_poll_group_xxx
                spdk_thread_poll() {
                        // run the thread msg ring function()
                        // e.g. bdevperf_complete()
                        msg_queue_run_batch(thread, max_msgs);

                        // poll all the pollers of the thread,
                        // e.g. nvmf_poll_group_poll
                        thread_execute_poller()
                }
        }
}
```

Ceph SPDK NVMe-oF Gateway介绍 - thread model

Gateway node thread model



Fio, spdk app(nvmf_tgt, bdevperf)

User threads

user complete_callback()

call rbd_aio_read/write()

complete

Ceph client(librbd1, librados2)

complete_external_callback()

io_context_pool thread pool

msgr-worker-x thread pool

User async read/write main threads.
fio job threads: fio
bdevperf, nvmf_tgt spdk threads:
Cephx
nvmf_tgt_poll_group_x

User complete_callback() E.g.
bdevperf_complete() triggered by complete_external_callback() or
fio_rbd_getevents() to return events

IO context poll threads.
Handle IO operate enqueue for lower
network layer to handle.

complete_external_callback() calls
bdev_rbd_finish_aiocb()
_fio_rbd_finish_aiocb()
wake up rbd_poll_io_events()

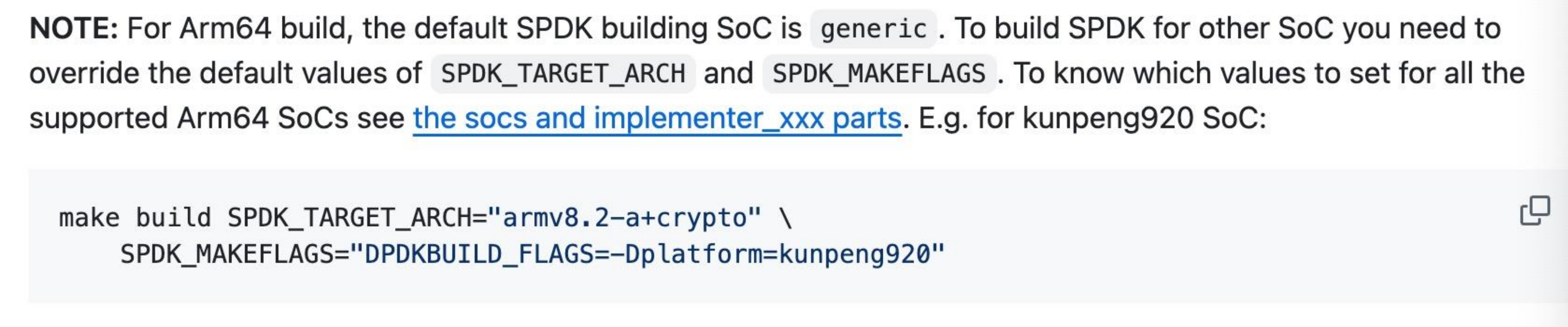
Network async Messenger threads.
Handle IO msg send/receive

Ceph SPDK NVMe-oF Gateway Arm上游支持

Ceph SPDK NVMe-oF Gateway Arm上游支持 – Enable

Enable building nvmeof container images for Arm64

- Ticket: https://github.com/ceph/ceph-nvmeof/issues/781
- PRs
  - Fix x86 hardcode: https://github.com/ceph/ceph-nvmeof/pull/782
  - Add Arm build: https://github.com/ceph/ceph-nvmeof/pull/783
- Arm Build Note: https://github.com/ceph/ceph-nvmeof/blob/devel/README.md



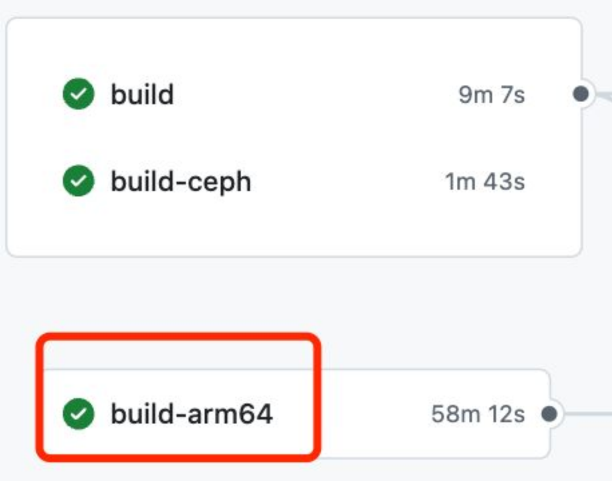
NOTE: For Arm64 build, the default SPDK building SoC is `generic`. To build SPDK for other SoC you need to override the default values of `SPDK_TARGET_ARCH` and `SPDK_MAKEFLAGS`. To know which values to set for all the supported Arm64 SoCs see the socs and implementer_xxx parts. E.g. for kunpeng920 SoC:

```
make build SPDK_TARGET_ARCH="armv8.2-a+crypto" \
    SPDK_MAKEFLAGS="DPDKBUILD_FLAGS=-Dplatform=kunpeng920"
```

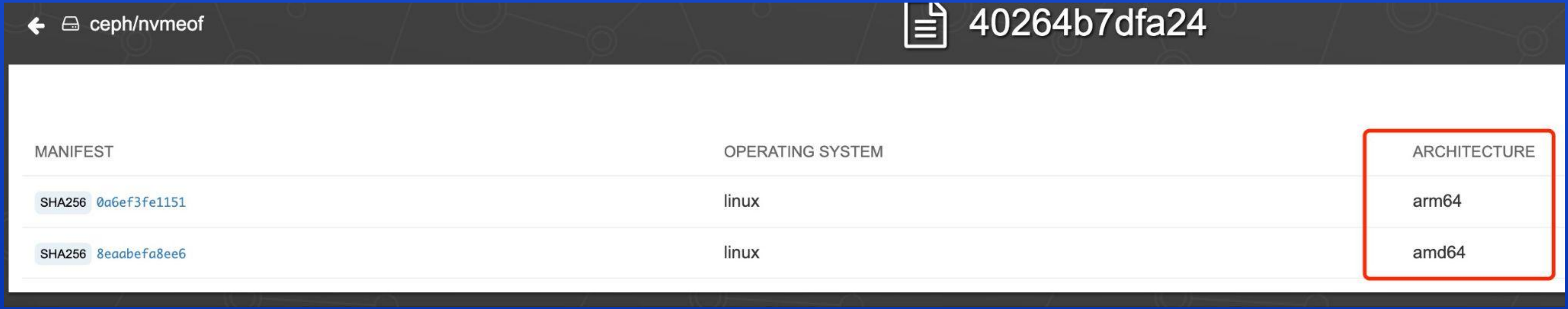
Ceph SPDK NVMe-oF Gateway Arm上游支持 – CI

Add build-arm64 job, multi-platform build

- Ticket: https://github.com/ceph/ceph-nvmeof/issues/805
- PR: https://github.com/ceph/ceph-nvmeof/pull/806
  - Migrate docker-compose from v1 to v2.
  - Support qemu multi-platform container image build.
  - Add build-arm64 job run per PR and daily.
  - Build and publish nvmeof, nvmeof-cli images to quay.io/ceph/.



✓ build	9m 7s
✓ build-ceph	1m 43s
✓ build-arm64	58m 12s



ceph/nvmeof

40264b7dfa24

MANIFEST	OPERATING SYSTEM	ARCHITECTURE
SHA256 0a6ef3fe1151	linux	arm64
SHA256 8eaabefa8ee6	linux	amd64

Ceph SPDK NVMe-oF Gateway性能测试

Ceph SPDK NVMe-oF Gateway性能测试 - Testbed

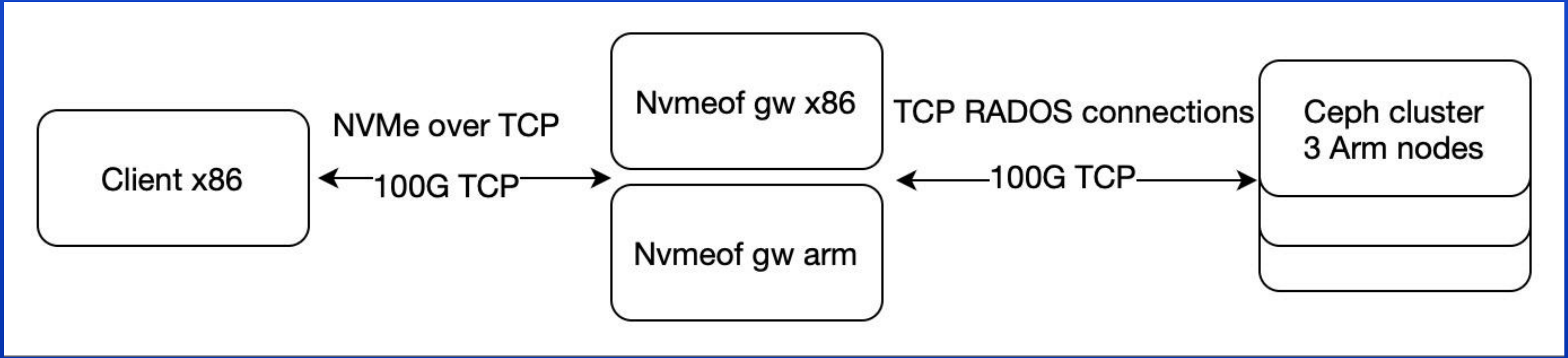
Test Environment

Hardware

- Arm CPU: Kunpeng 920, 2.6GHz, 4 numa nodes
- X86 (Gateway) CPU: Intel(R) Xeon(R) Platinum 8180 CPU @ 2.50GHz, 112 cpus, 2 numa nodes
- x86-2 (Client) CPU: Intel(R) Xeon(R) Gold 6248R CPU @ 3.00GHz, 96 cpus, 2 numa nodes
- Disk: 3 x ES3000 V6 NVMe SSD 3.2T per Arm server
- Network: 1x MLNX ConnectX-5 100Gb IB，1x1G tcp

Software

- OS: openEuler 22.03 LTS SP3，kernel 5.10.0-192.0.0.105.oe2203sp3
- **Ceph**: main-nvmeof **require revert commit** "nvmeof gw monitor: disable by default"
- SPDK: 24.05
- nvmeof: 1.3.2
- fio: fio-3.29

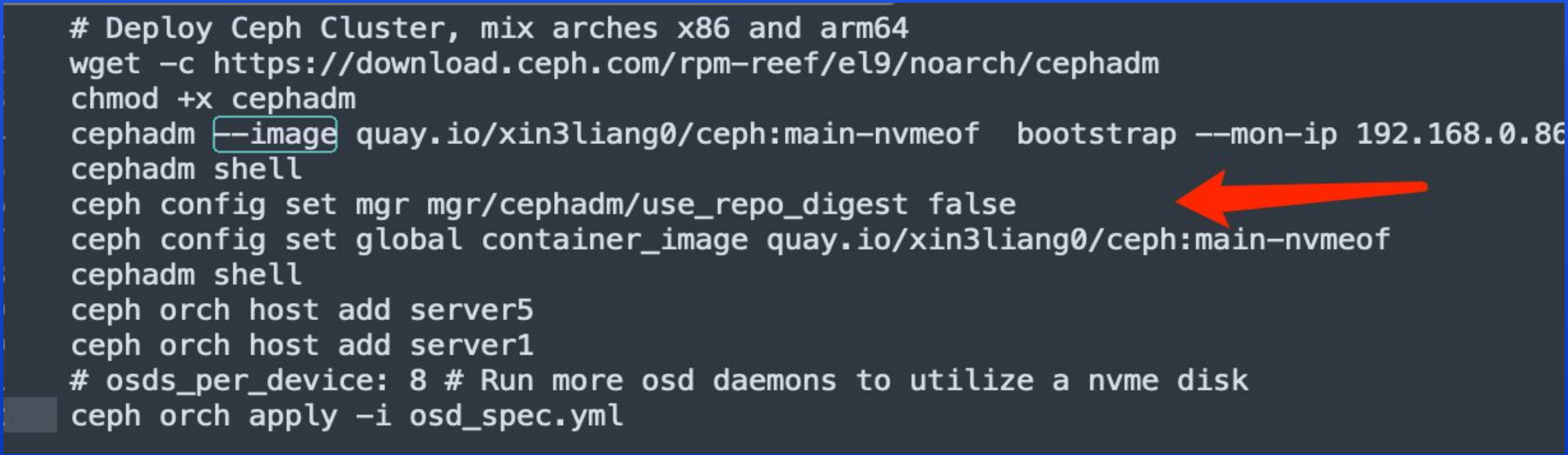


Ceph SPDK NVMe-oF Gateway性能测试 - Testbed

Arm and x86 hybrid deployment

Reference:

- https://docs.ceph.com/en/latest/cephadm/install/#bootstrap-a-new-cluster
- https://docs.ceph.com/en/latest/cephadm/operations/#purging-a-cluster



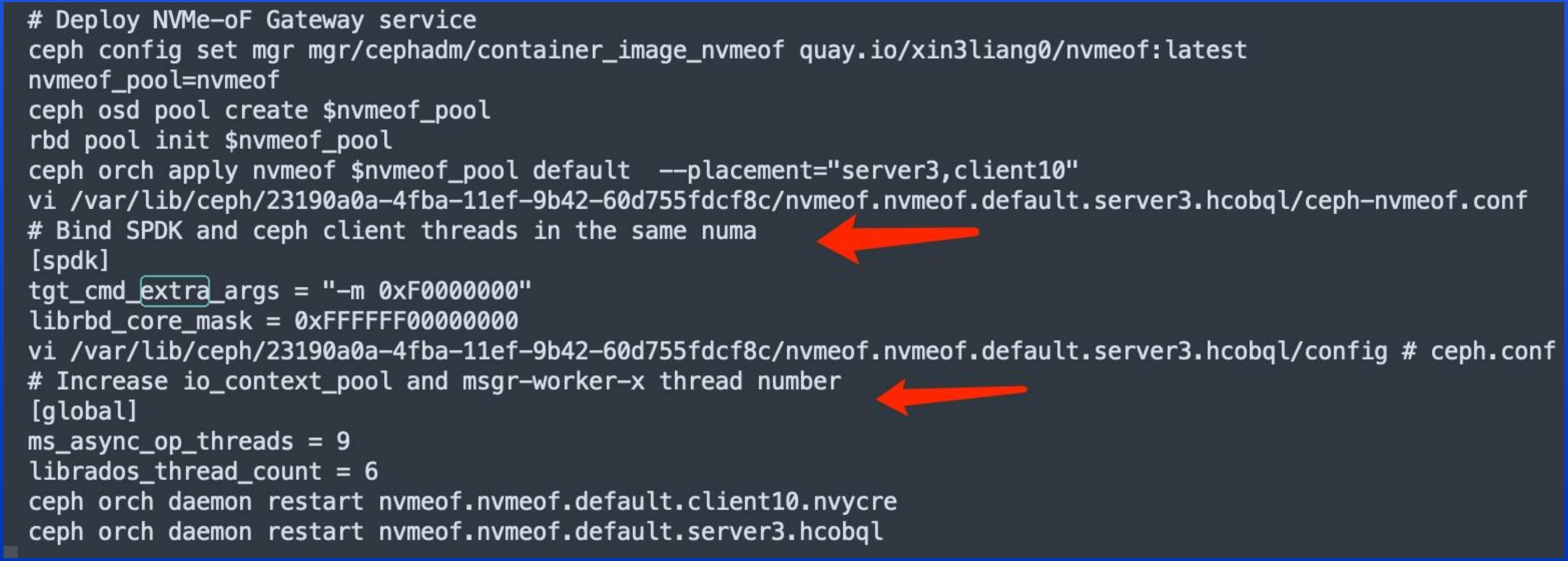
```
# Deploy Ceph Cluster, mix arches x86 and arm64
wget -c https://download.ceph.com/rpm-reef/el9/noarch/cephadm
chmod +x cephadm
cephadm --image quay.io/xin3liang0/ceph:main-nvmeof  bootstrap --mon-ip 192.168.0.86
cephadm shell
ceph config set mgr mgr/cephadm/use_repo_digest false
ceph config set global container_image quay.io/xin3liang0/ceph:main-nvmeof
cephadm shell
ceph orch host add server5
ceph orch host add server1
# osds_per_device: 8 # Run more osd daemons to utilize a nvme disk
ceph orch apply -i osd_spec.yml
```

Ceph SPDK NVMe-oF Gateway性能测试 - Testbed

Arm and x86 hybrid deployment

Reference

- https://docs.ceph.com/en/latest/rbd/nvmeof-target-configure/
- https://ci.spdk.io/download/2022-virtual-forum-prc/D2_4_Yue_A_Performance_Study_for_Ceph_NVMeoF_Gateway.pdf
- https://ci.spdk.io/download/2022-virtual-forum-prc/D2_3_Yifan_The_usage_of_SPDK_RBD_bdev_and_performance_tuning.pdf

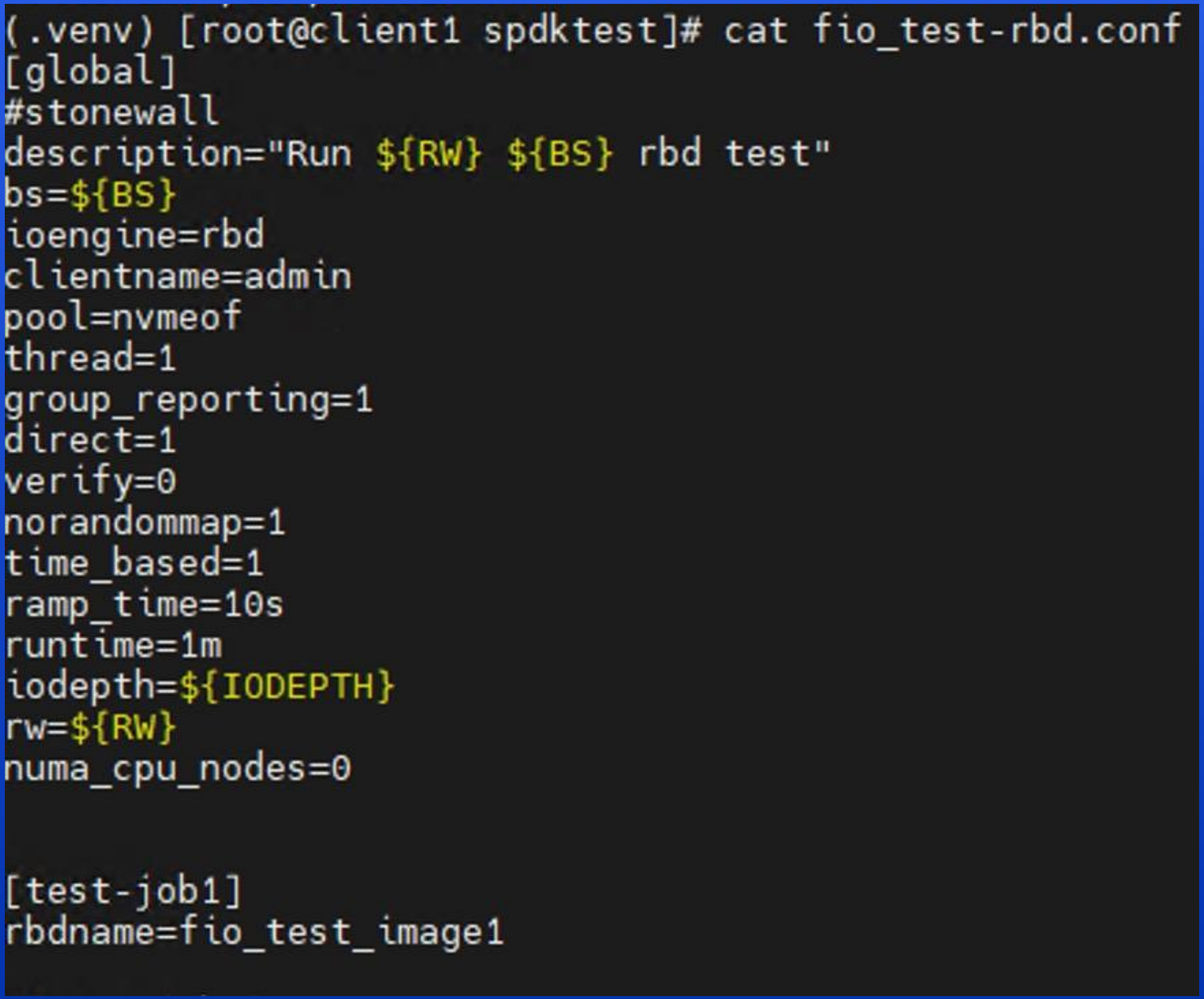


```
# Deploy NVMe-oF Gateway service
ceph config set mgr mgr/cephadm/container_image_nvmeof quay.io/xin3liang0/nvmeof:latest
nvmeof_pool=nvmeof
ceph osd pool create $nvmeof_pool
rbd pool init $nvmeof_pool
ceph orch apply nvmeof $nvmeof_pool default  --placement="server3,client10"
vi /var/lib/ceph/23190a0a-4fba-11ef-9b42-60d755fdcf8c/nvmeof.nvmeof.default.server3.hcobql/ceph-nvmeof.conf
# Bind SPDK and ceph client threads in the same numa
[spdk]
tgt_cmd_extra_args = "-m 0xF0000000"
librbd_core_mask = 0xFFFFFF00000000
vi /var/lib/ceph/23190a0a-4fba-11ef-9b42-60d755fdcf8c/nvmeof.nvmeof.default.server3.hcobql/config # ceph.conf
# Increase io_context_pool and msgr-worker-x thread number
[global]
ms_async_op_threads = 9
librados_thread_count = 6
ceph orch daemon restart nvmeof.nvmeof.default.client10.nvycre
ceph orch daemon restart nvmeof.nvmeof.default.server3.hcobql
```

Ceph SPDK NVMe-oF Gateway性能测试 - 1 rbd test

FIO config

- FIO rbd config

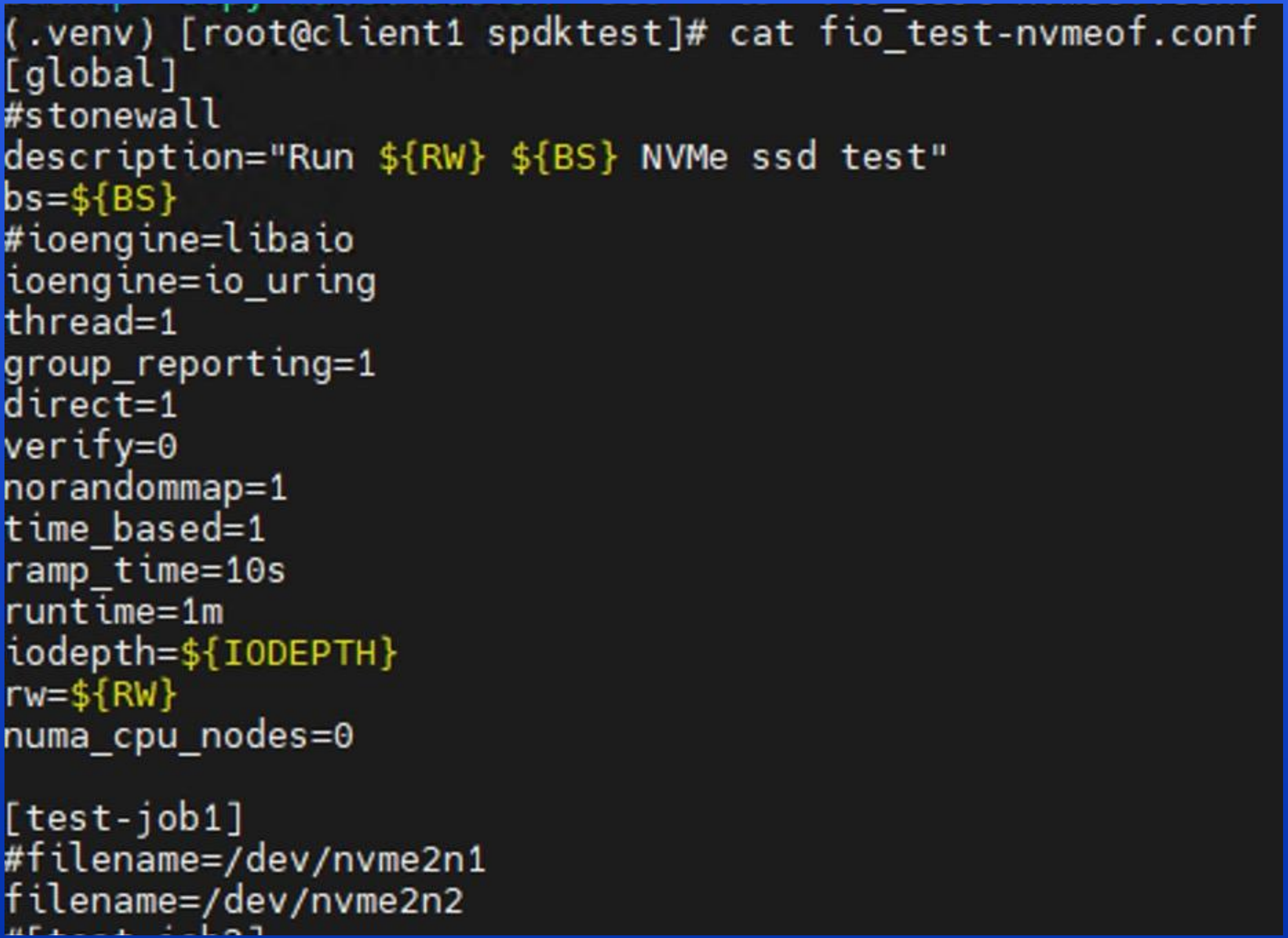


```
(.venv) [root@client1 spdktest]# cat fio_test-rbd.conf
[global]
#stonewall
description="Run ${RW} ${BS} rbd test"
bs=${BS}
ioengine=rbd
clientname=admin
pool=nvmeof
thread=1
group_reporting=1
direct=1
verify=0
norandommap=1
time_based=1
ramp_time=10s
runtime=1m
iodepth=${IODEPTH}
rw=${RW}
numa_cpu_nodes=0


[test-job1]
rbdname=fio_test_image1
```

Ceph SPDK NVMe-oF Gateway性能测试 - 1 rbd test

FIO config

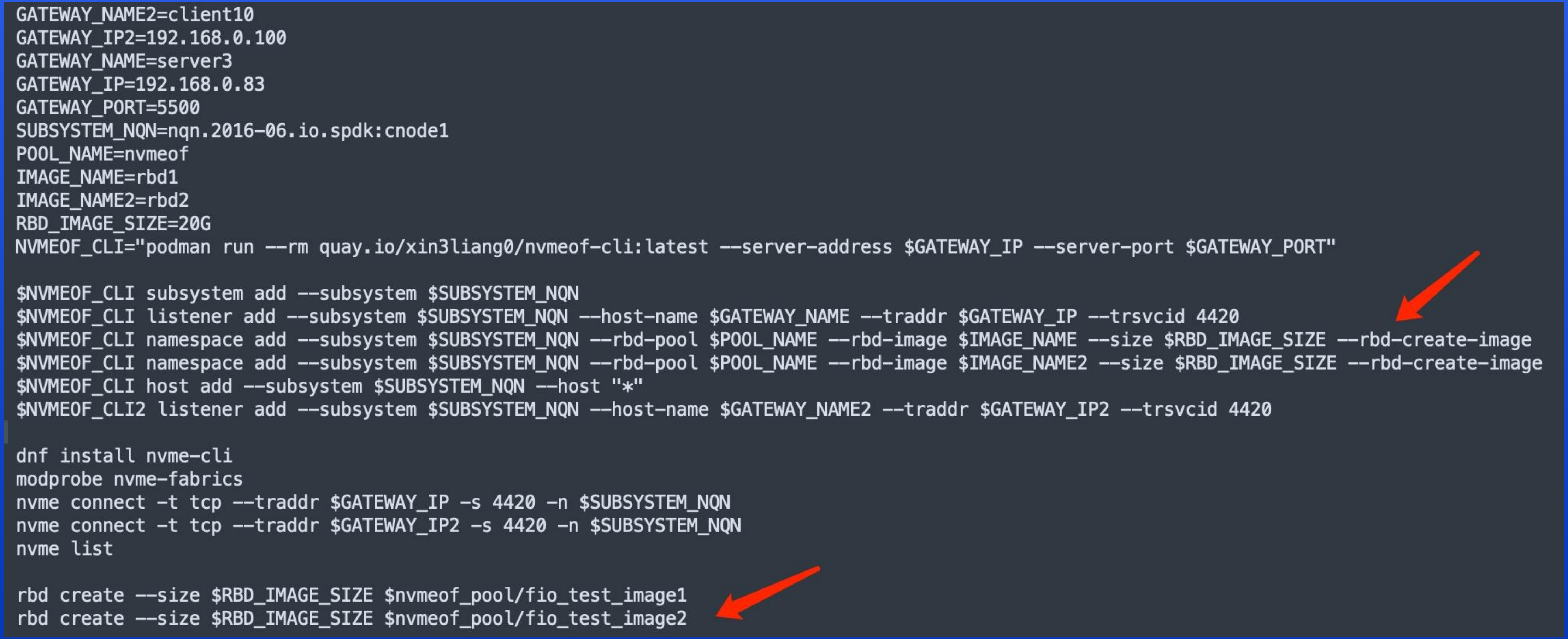
- FIO nvmeof config



```
(.venv) [root@client1 spdktest]# cat fio_test-nvmeof.conf
[global]
#stonewall
description="Run ${RW} ${BS} NVMe ssd test"
bs=${BS}
#ioengine=libaio
ioengine=io_uring
thread=1
group_reporting=1
direct=1
verify=0
norandommap=1
time_based=1
ramp_time=10s
runtime=1m
iodepth=${IODEPTH}
rw=${RW}
numa_cpu_nodes=0

[test-job1]
#filename=/dev/nvme2n1
filename=/dev/nvme2n2
```

Ceph SPDK NVMe-oF Gateway性能测试 - 1 rbd test

Create test rbds

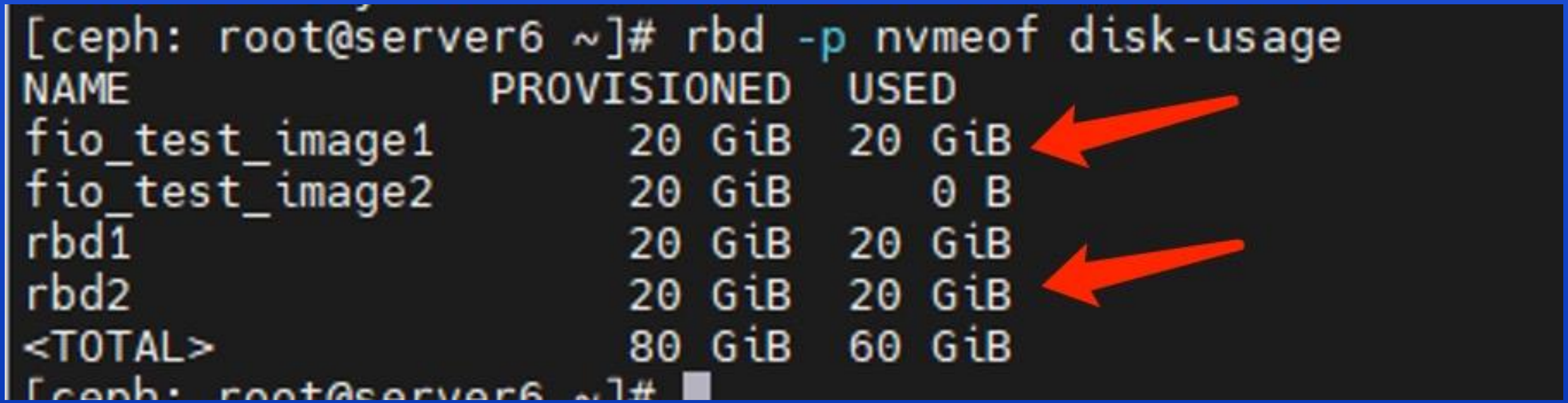


```
GATEWAY_NAME2=client10
GATEWAY_IP2=192.168.0.100
GATEWAY_NAME=server3
GATEWAY_IP=192.168.0.83
GATEWAY_PORT=5500
SUBSYSTEM_NQN=nqn.2016-06.io.spdk:cnode1
POOL_NAME=nvmeof
IMAGE_NAME=rbd1
IMAGE_NAME2=rbd2
RBD_IMAGE_SIZE=20G
NVMEOF_CLI="podman run --rm quay.io/xin3liang0/nvmeof-cli:latest --server-address $GATEWAY_IP --server-port $GATEWAY_PORT"

$NVMEOF_CLI subsystem add --subsystem $SUBSYSTEM_NQN
$NVMEOF_CLI listener add --subsystem $SUBSYSTEM_NQN --host-name $GATEWAY_NAME --traddr $GATEWAY_IP --trsvcid 4420
$NVMEOF_CLI namespace add --subsystem $SUBSYSTEM_NQN --rbd-pool $POOL_NAME --rbd-image $IMAGE_NAME --size $RBD_IMAGE_SIZE --rbd-create-image
$NVMEOF_CLI namespace add --subsystem $SUBSYSTEM_NQN --rbd-pool $POOL_NAME --rbd-image $IMAGE_NAME2 --size $RBD_IMAGE_SIZE --rbd-create-image
$NVMEOF_CLI host add --subsystem $SUBSYSTEM_NQN --host "*"
$NVMEOF_CLI2 listener add --subsystem $SUBSYSTEM_NQN --host-name $GATEWAY_NAME2 --traddr $GATEWAY_IP2 --trsvcid 4420

dnf install nvme-cli
modprobe nvme-fabrics
nvme connect -t tcp --traddr $GATEWAY_IP -s 4420 -n $SUBSYSTEM_NQN
nvme connect -t tcp --traddr $GATEWAY_IP2 -s 4420 -n $SUBSYSTEM_NQN
nvme list

rbd create --size $RBD_IMAGE_SIZE $nvmeof_pool/fio_test_image1
rbd create --size $RBD_IMAGE_SIZE $nvmeof_pool/fio_test_image2
```

Ceph SPDK NVMe-oF Gateway性能测试 - 1 rbd test

SSD rbd Preconditioning(预处理)

- Sequential write 2X+ size of rbd to get SSD steady state.
  - RW=write BS=128k IODEPTH=128 fio ./[fio_test-rbd.conf|fio_test-nvmeof.conf] --numjobs=1 # run 1 min
- Reference
  - SSD Preconditioning: https://ci.spdk.io/download/performance-reports/SPDK_nvme_bdev_perf_report_2405.pdf



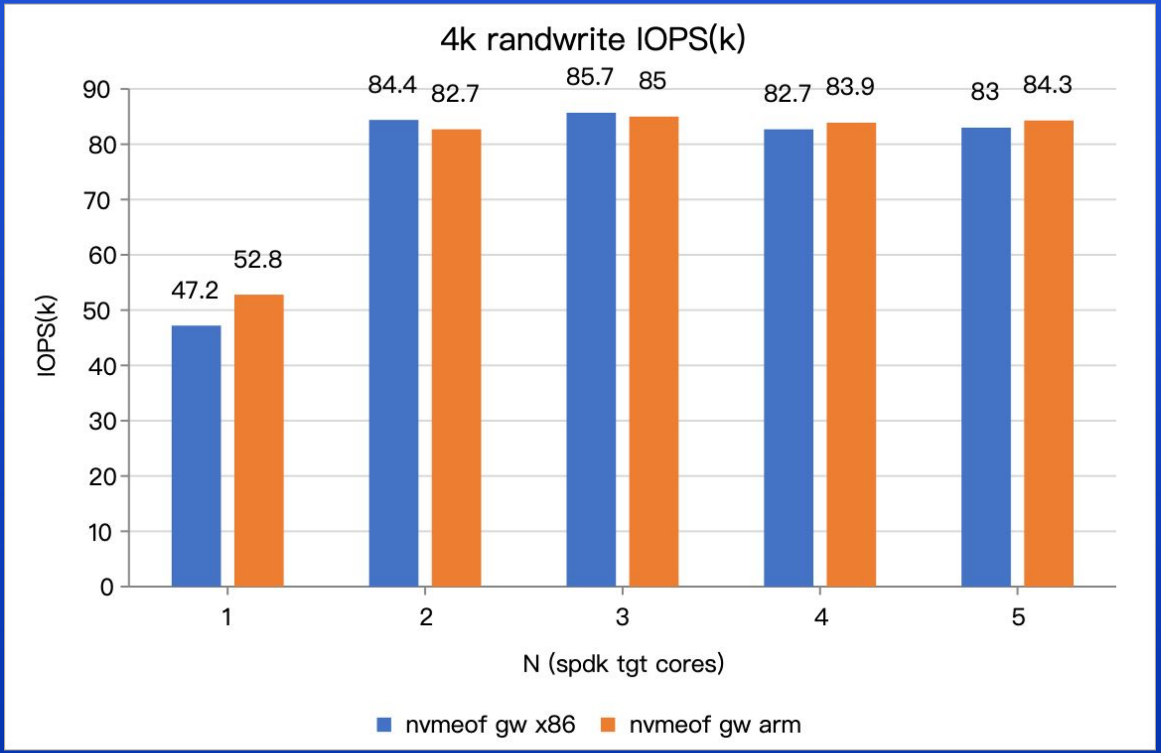
```
[ceph: root@server6 ~]# rbd -p nvmeof disk-usage
NAME              PROVISIONED  USED
fio_test_image1        20 GiB  20 GiB
fio_test_image2        20 GiB     0 B
rbd1                   20 GiB  20 GiB
rbd2                   20 GiB  20 GiB
<TOTAL>                80 GiB  60 GiB
[ceph: root@server6 ~]#
```

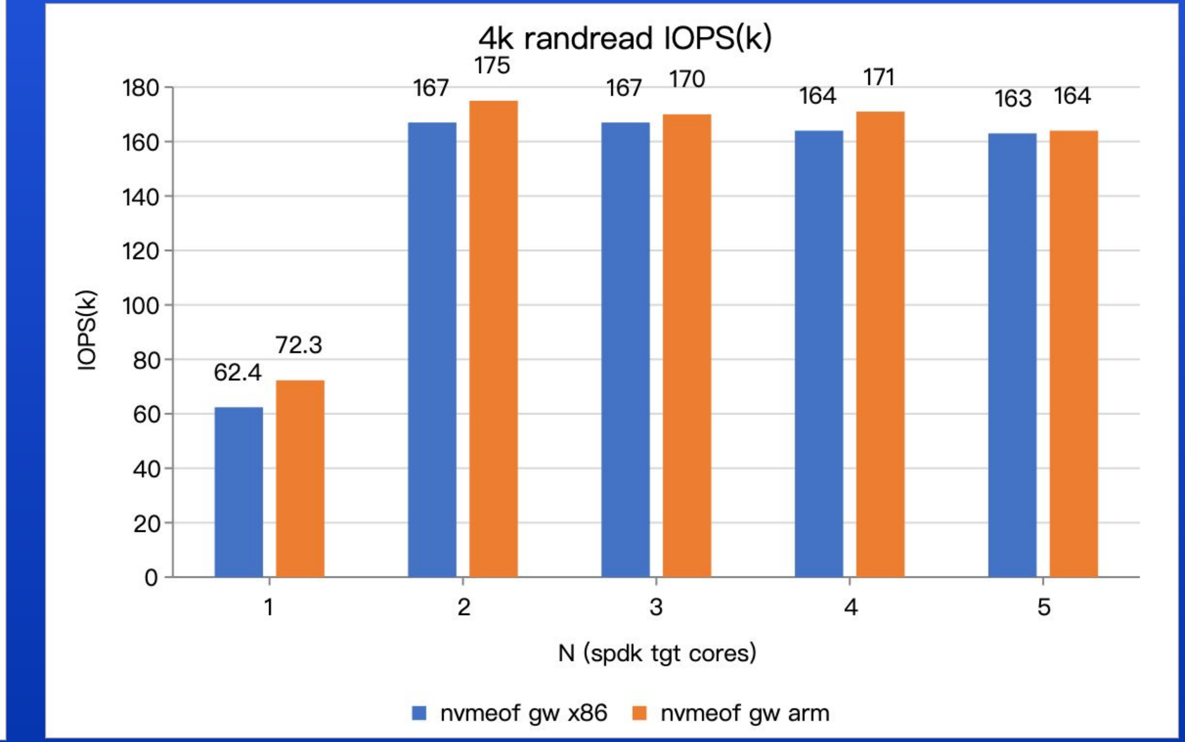
Ceph SPDK NVMe-oF Gateway性能测试 - 1 rbd test

Case 1: scale spdk cores (triple ceph client msg and io enqueue threads, bind ceph client, spdk tgt threads in the same numa as NIC)

4k RW=randwrite|randread IODEPTH=128 fio ~/spdktest/fio_test-nvmeof.conf --numjobs=8

- ～50k IOPS per core
- Find out cpus via spdk_top cpu not 100%



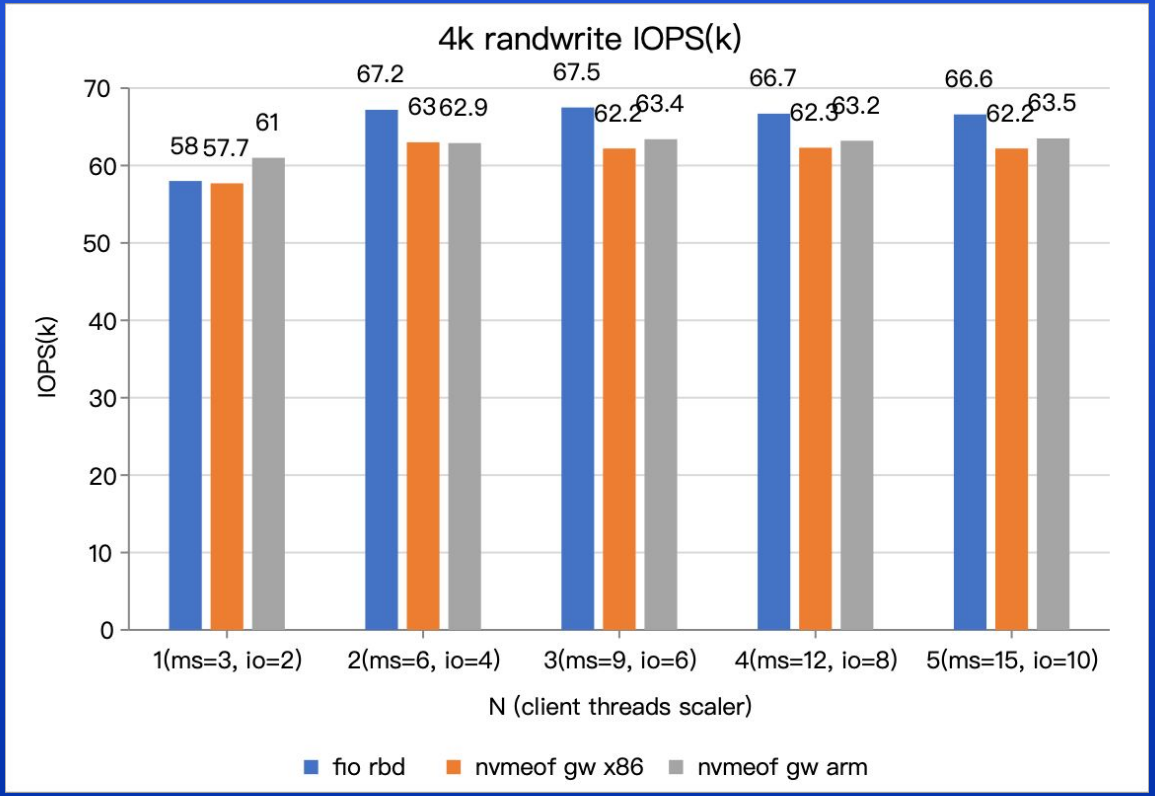


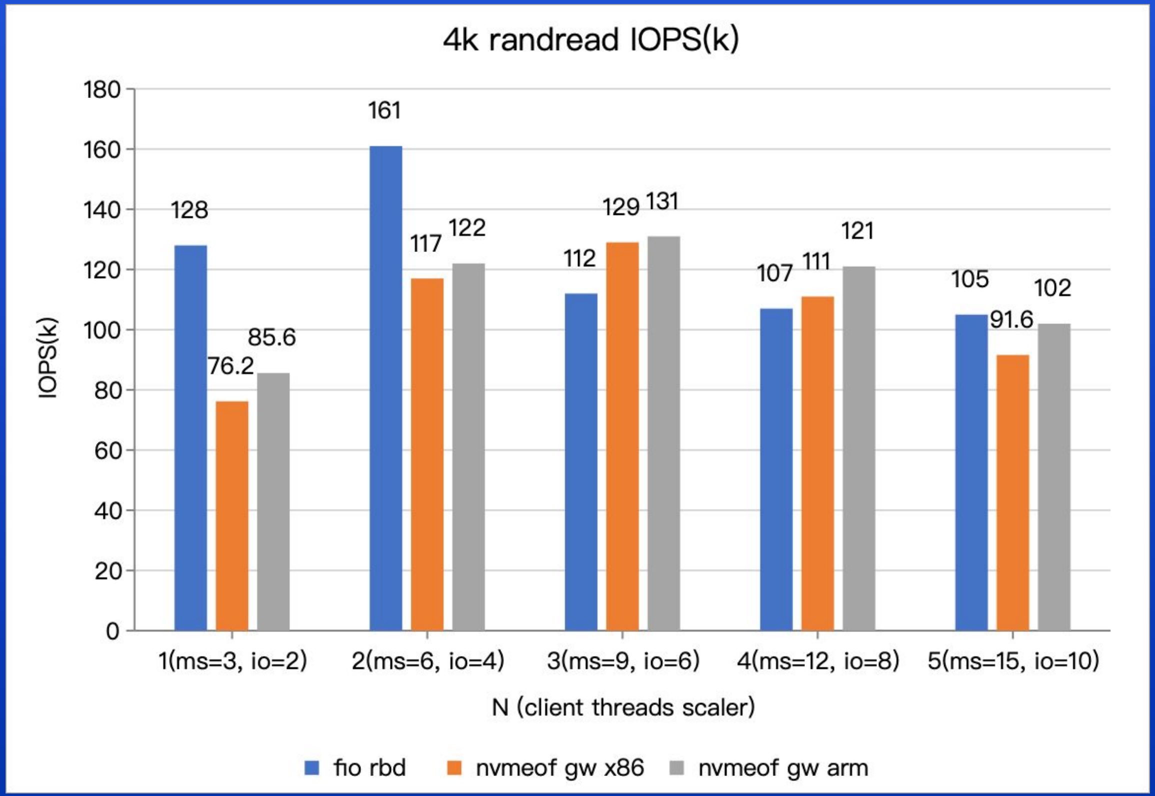
Ceph SPDK NVMe-oF Gateway性能测试 - 1 rbd test

Case 2: scale ceph client msg and io enqueue threads (4 spdk cores , bind ceph client, spdk tgt threads in the same numa as NIC)

4k RW=randwrite IODEPTH=128 fio ~/spdktest/[fio_test-rbd.conf|fio_test-nvmeof.conf] --numjobs=1

- ms_async_op_threads = 3 * N , librados_thread_count = 2 * N
- Find out thread's cpu not 100% via top, N = 2 fio (N = 3 nvmeof) get best performance number





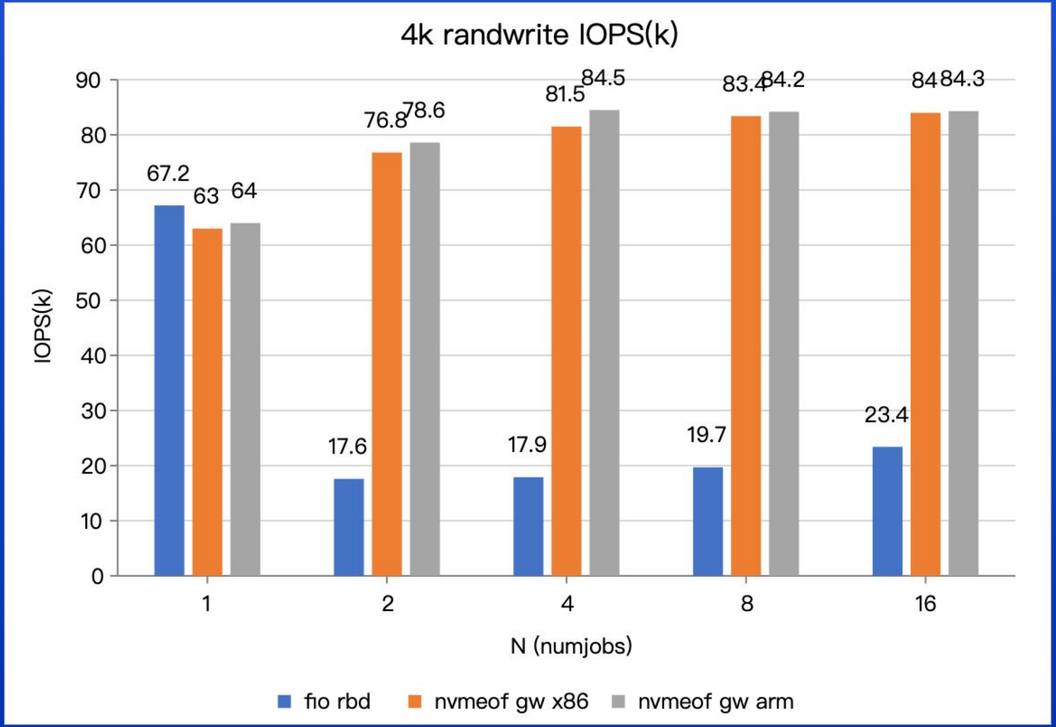
Ceph SPDK NVMe-oF Gateway性能测试 - 1 rbd test

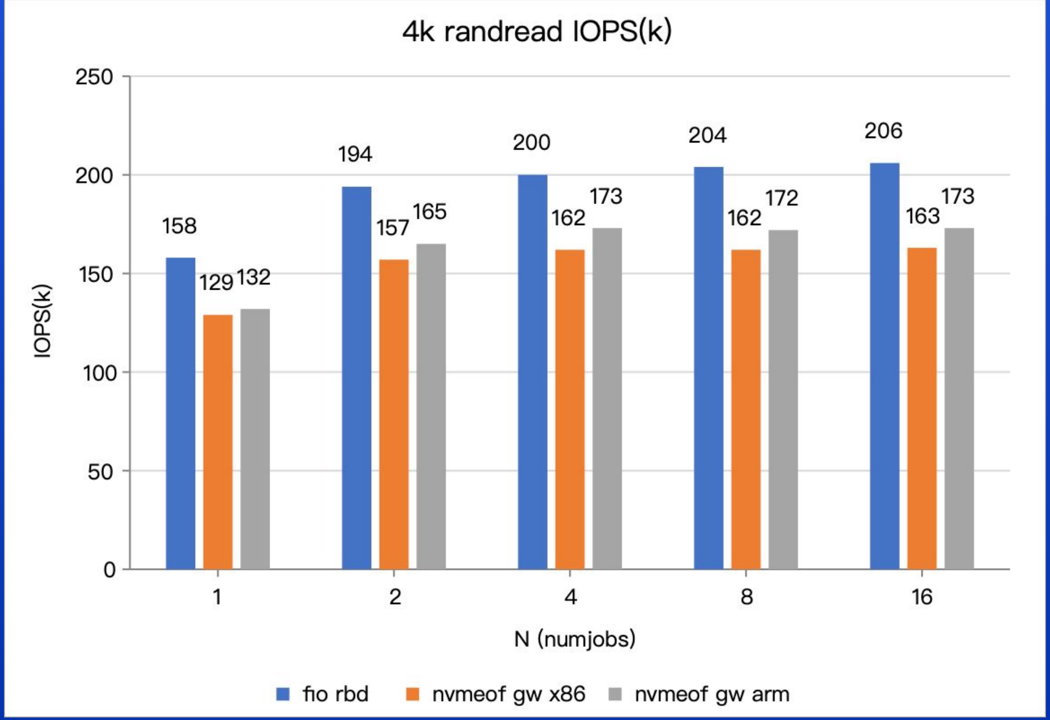
Case 3: double/triple client msg and io enqueue threads (4 spdk cores, bind ceph client, spdk tgt threads in the same numa as NIC)

4k RW=randwrite IODEPTH=128 fio ~/spdktest/[fio_test-rbd.conf|fio_test-nvmeof.conf] --numjobs=N

- FIO rbd: ms_async_op_threads = 6 # 3->6, librados_thread_count = 4 # 2->4
- FIO rbd: Should numjobs=1, because 1 rados connection, per job multi rados connect performance drop

  Due to the rbd exclusive write lock
- Nvmeof : ms_async_op_threads = 9 # 3->9 , librados_thread_count = 6 # 2->6 , X86 gw ~20% drop, Arm gw ~15% drop

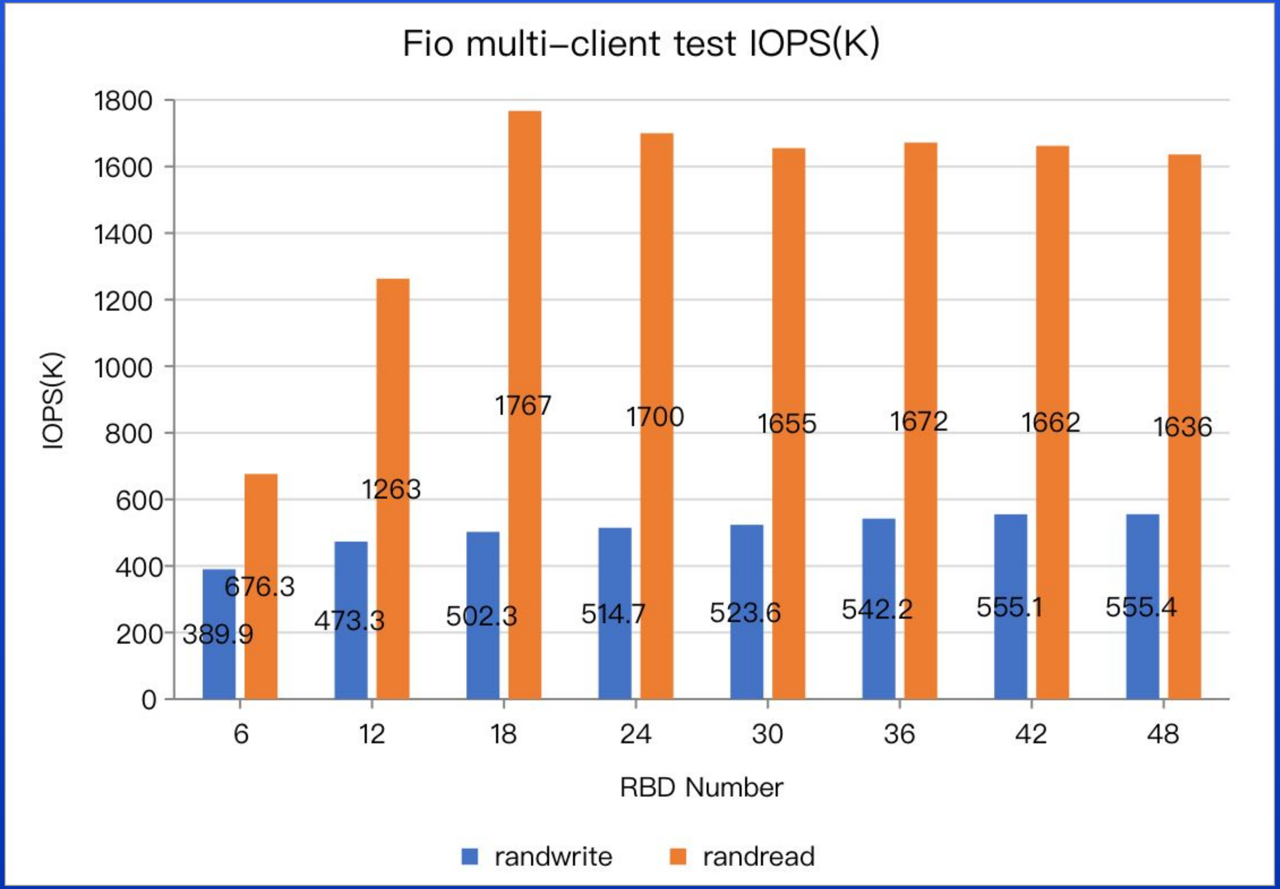




Ceph SPDK NVMe-oF Gateway性能测试 - multi-rbd test

Case 1: fio rbd multi-client test for ceph cluster capability (numjob=1, client_num=6)

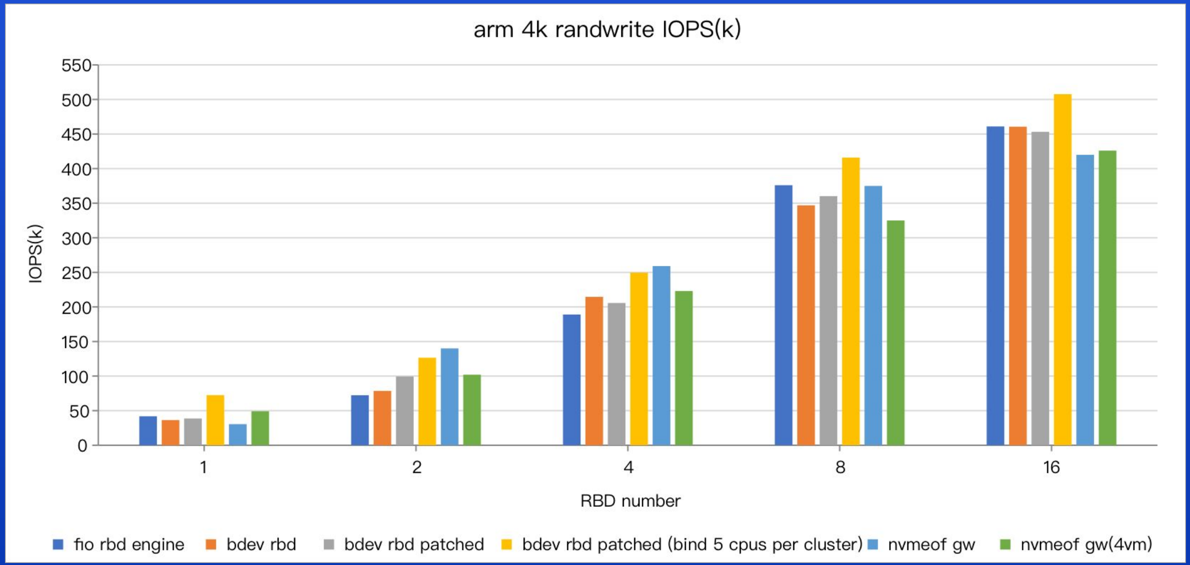
- ./fio –server& # run in the clients nodes
- ./run-n-fio-rbd-test_multi-client.sh 48 # run in one client node

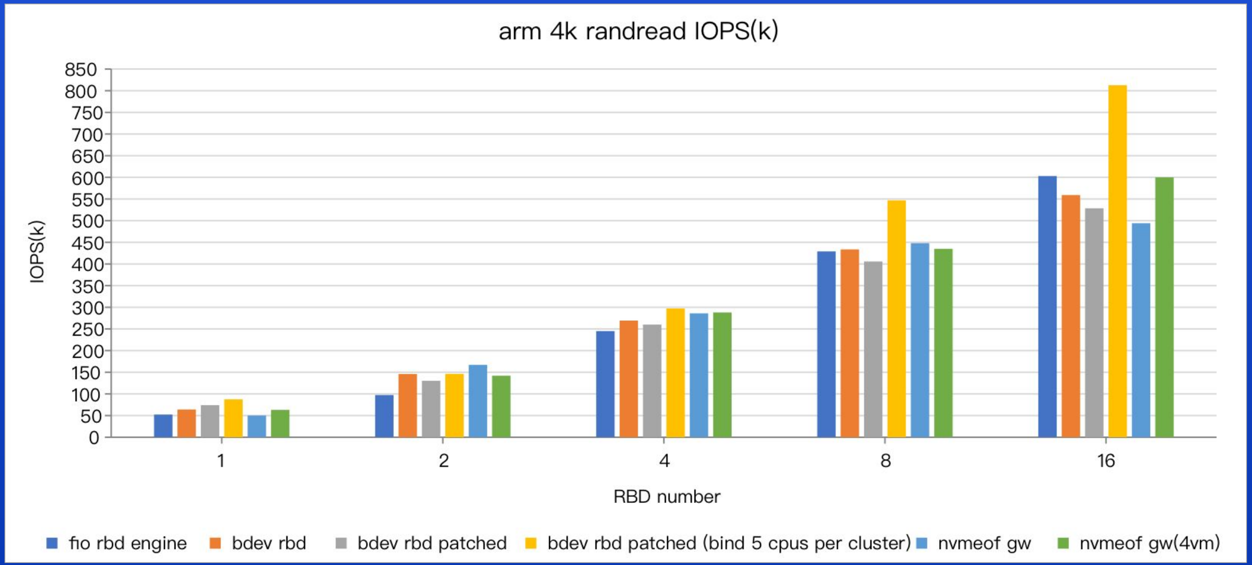


Ceph SPDK NVMe-oF Gateway性能测试 - multi-rbd test

Case 2: fio, bdev, gw rbd test (numjob=1, 16 spdk cores, bdevs_per_cluster = 1, iommu.passthrough=1）

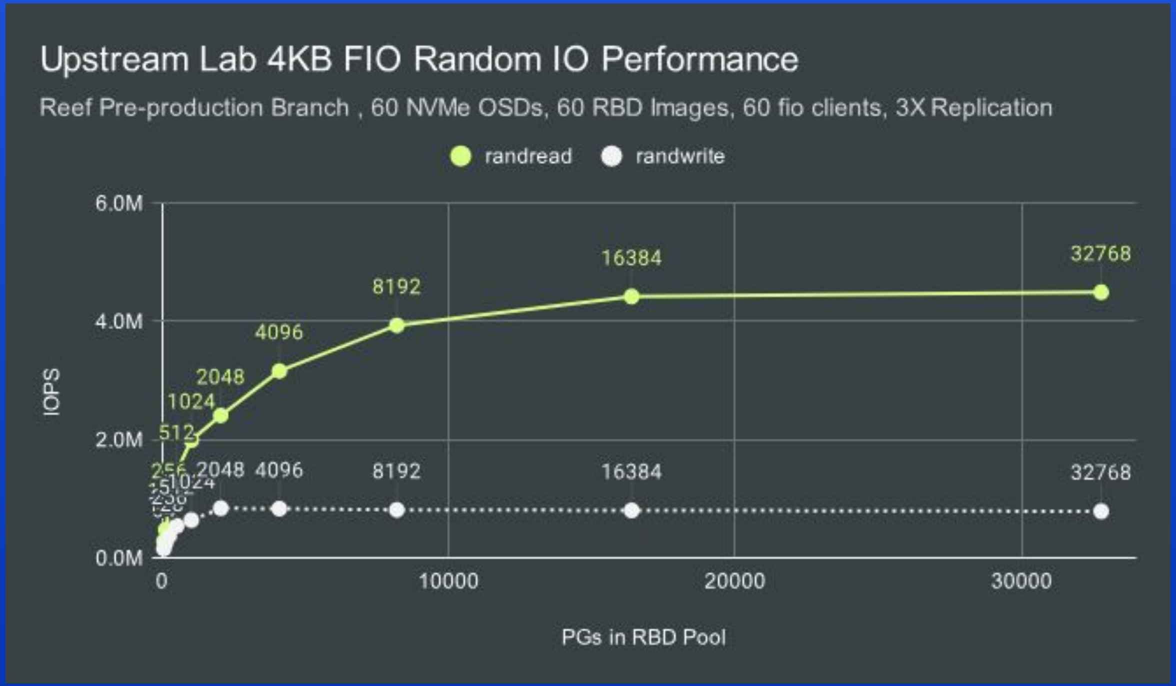
- bdevperf -q 128 -o 4096 -w randwrite -t 60 -m [0-15] -z # run in one terminal
- ./run-n-bdev-rbd-test1.sh N, ./run-n-fio-nvmeof-test1.sh N, run-n-fio-rbd-test1.sh N,
  - ./run-bdev-rbd-test2.sh N 16-95 # each ceph client threads binding to 5 cpu cores
- Test scripts: https://github.com/xin3liang/home-bin/tree/master/spdk-fio





Ceph SPDK NVMe-oF Gateway性能测试 - multi-rbd test tips

- Tip1: Ceph pool pg_num impacts performance much
  - pg_autoscale_mode scales pg_num according pool space utilization
  - Setting pg_num can get the maximum performance at the pool creation beginning
  - pool=nvmeof; ceph osd pool create $pool --size 1 --pg_num 16384 --pgp_num 16384 --bulk
  - #ceph osd pool set $pool target_size_ratio 0.9
  - ceph osd pool set $pool pg_autoscale_mode off
  - rbd pool init $pool
  - References
    - Ceph: A Journey to 1 TiB/s
    - PG Calc — Ceph Documentation
    - Placement Groups — Ceph Documentation



Upstream Lab 4KB FIO Random IO Performance

Reef Pre-production Branch , 60 NVMe OSDs, 60 RBD Images, 60 fio clients, 3X Replication

Ceph SPDK NVMe-oF Gateway性能测试 - multi-rbd test tips

- Tip2: Nvmeof-gateway should build release type
  - Upstream tacking card: https://github.com/ceph/ceph-nvmeof/issues/938
  - SPDK_CONFIGURE_ARGS="--with-rbd --disable-tests --disable-unit-tests --disable-examples **--enable-debug**"
  - Debug build type performance is pool.

- Tip3: Build LSE atomic instructions for Arm64
  - Check LSE: perf top see __aarch64_xxx_xxx_rel, e.g. __aarch64_ldadd8_acq_rel
  - Check LSE: objdump and egrep -i

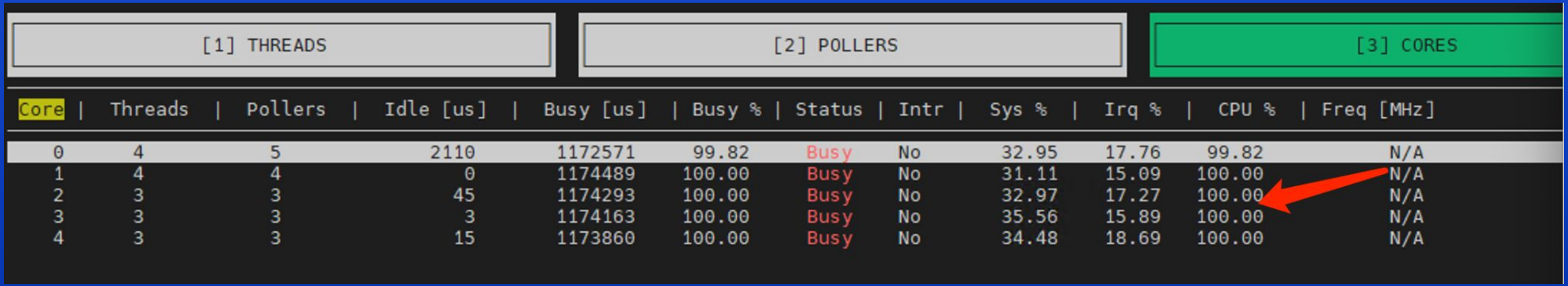
    'cas|casp|swp|ldadd|stadd|ldclr|stclr|ldeor|steor|ldset|stset|ldsmax|stsmax|ldsmin|stsmin|ldumax|stumax|ldumin|stumin'
  - Check load exclusives and store exclusives: objdump and egrep -i 'ldxr|ldaxr|stxr|stlxr'
  - Refer to: https://learn.arm.com/learning-paths/servers-and-cloud-computing/lse/example/

- Tip4: Passthrough iommu
  - iommu overhead is nonnegligible
  - Disable or Passthrogh it
    - kernel cmdline: iommu.passthrough=1 (arm64), iommu=pt(Intel)

Ceph SPDK NVMe-oF Gateway性能测试 - multi-rbd test tips

- Tip5: Nvmeof controller IO queue number >= SPDK CPU cores number (but not too larger than)
    - ceph-nvmeof.conf: "max_io_qpairs_per_ctrlr": 7
    - To utlize all the spdk cores and get best performance.

- Tip6: Scale spdk cores
    - Add spdk cores when spdk_top cpu% is full
    - ceph-nvmeof.conf: tgt_cmd_extra_args = “-m 0xFF”
    - Might need to increase iobuf pool
        - RPC=/usr/libexec/spdk/scripts/rpc.py
        - bdevperf -q 128 -o 4096 -w randwrite -t 60 -m [0-15] -z --wait-for-rpc
        - $RPC iobuf_set_options --small-pool-count 32767 --large-pool-count 8191 && $RPC framework_start_init



```
        [1] THREADS                    [2] POLLERS                        [3] CORES

Core |  Threads  |  Pollers  |  Idle [us]  |  Busy [us]  | Busy % | Status | Intr |  Sys %  |  Irq %  |  CPU %  | Freq [MHz]

 0        4           5           2110        1172571      99.82    Busy     No      32.95     17.76     99.82          N/A
 1        4           4              0        1174489     100.00    Busy     No      31.11     15.09    100.00          N/A
 2        3           3             45        1174293     100.00    Busy     No      32.97     17.27    100.00          N/A
 3        3           3              3        1174163     100.00    Busy     No      35.56     15.89    100.00          N/A
 4        3           3             15        1173860     100.00    Busy     No      34.48     18.69    100.00          N/A
```

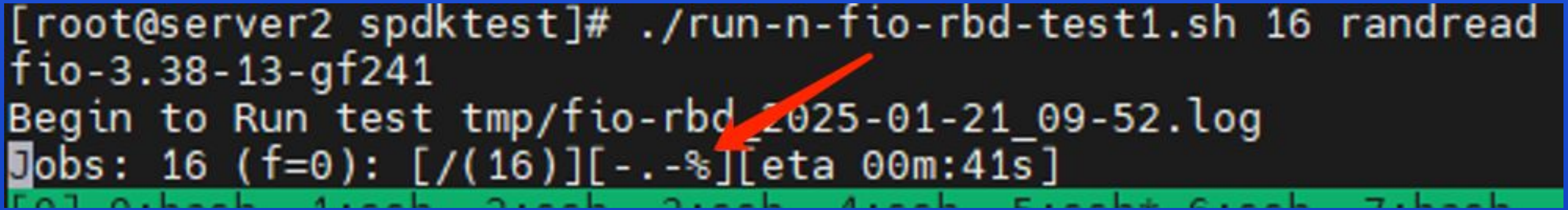
Ceph SPDK NVMe-oF Gateway性能测试 - multi-rbd test tips

- Tip7: Tune Ceph client thread number
  - ceph-nvmeof.conf: : bdevs_per_cluster = 1 as fio rbd engine
    - https://github.com/ceph/ceph-nvmeof/issues/939
  - ceph.conf
    - ms_async_op_threads 3 is default, msgr-worker-x thread number
    - librados_thread_count 2 is default, io_context_pool thread number
    - Tune if top cpu% is near 100%
- Tip8: Bind Ceph client threads
  - Binding cpu could get better performance
  - ceph-nvmeof.conf: librbd_core_mask = 0xFFFF0000
  - bdevperf test script run-n-bdev-rbd-test2.sh
    - $RPC bdev_rbd_register_cluster $cluster --core-mask "$cpu_list"
    - cpus_per_cluster=5

Ceph SPDK NVMe-oF Gateway性能测试 - multi-rbd test issue

Issue1: Fio rbd engine can only run small number of jobs

- Upstream tracking issue: https://github.com/axboe/fio/issues/1778
  - When running more jobs it will get stuck and no any fio test is running
  - Workaround: running in a container environment without this issue
    - podman run -it --pids-limit 4096 --name bdevtest centos-stream9

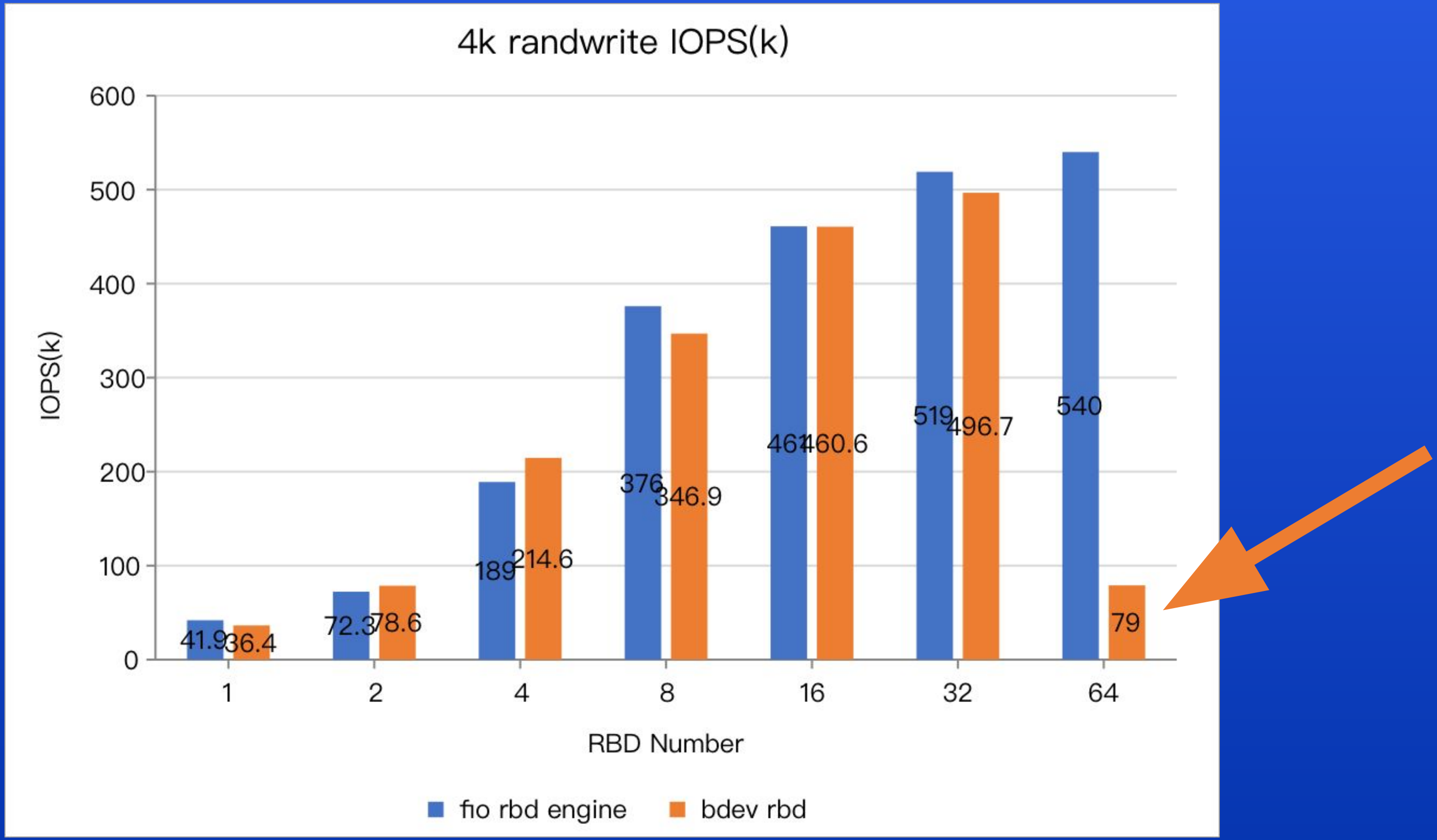


```
[root@server2 spdktest]# ./run-n-fio-rbd-test1.sh 16 randread
fio-3.38-13-gf241
Begin to Run test tmp/fio-rbd_2025-01-21_09-52.log
Jobs: 16 (f=0): [/(16)][-.-%][eta 00m:41s]
[0] 0:bash  1:ssh  2:ssh  3:ssh  4:ssh  5:ssh* 6:ssh  7:bash
```

Ceph SPDK NVMe-oF Gateway性能测试 - multi-rbd test issue

Issue2: 4k randwrite/randread IOPS drops rapidly as the RBD number grows large

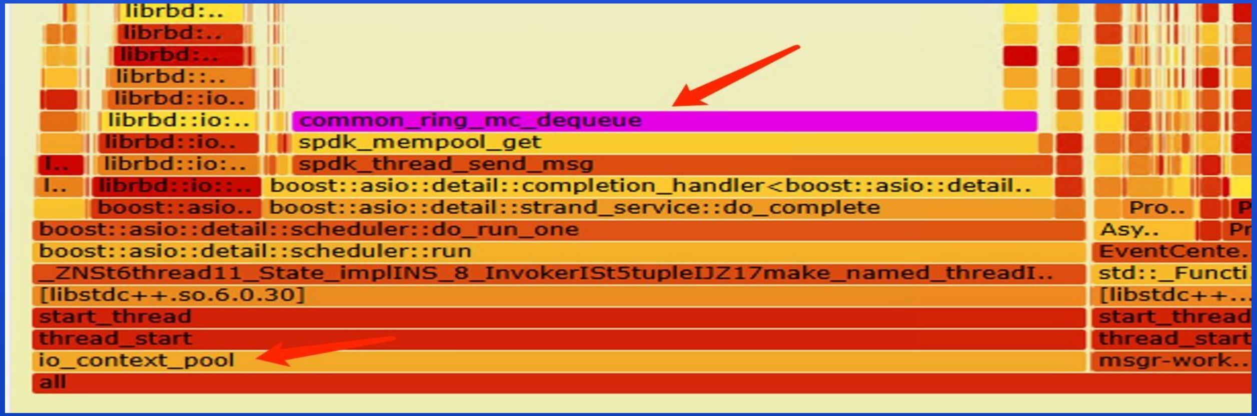
- Upstream tracking issue: https://github.com/spdk/spdk/issues/3547

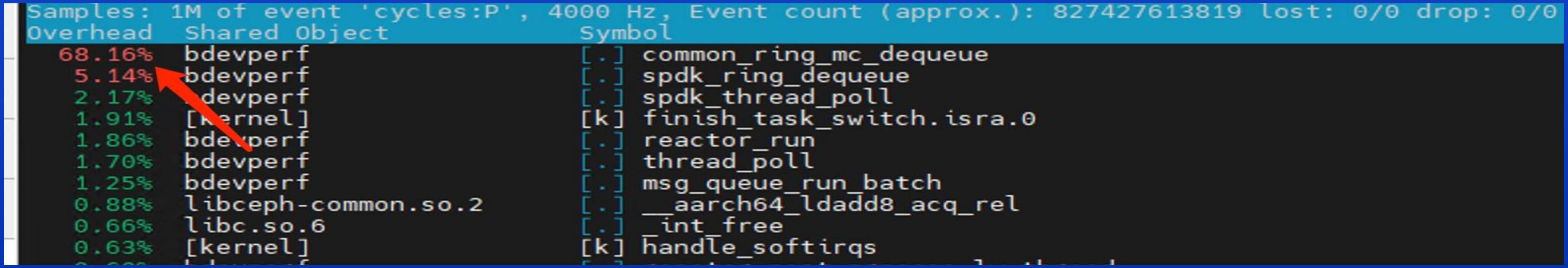


Ceph SPDK NVMe-oF Gateway性能测试 - multi-rbd test issue

Issue2: 4k randwrite/randread IOPS drops rapidly as the RBD number grows large

- Bottleneck analysis
  - The perf top and flame graph show that common_ring_mc_dequeue() calls spend too much time.
  - io_context_pool threads are non-EAL spdk threads.
  - Due to the lack of per-lcore caching, rte_mempool_get() performance will suffer when called by unregistered non-EAL threads.





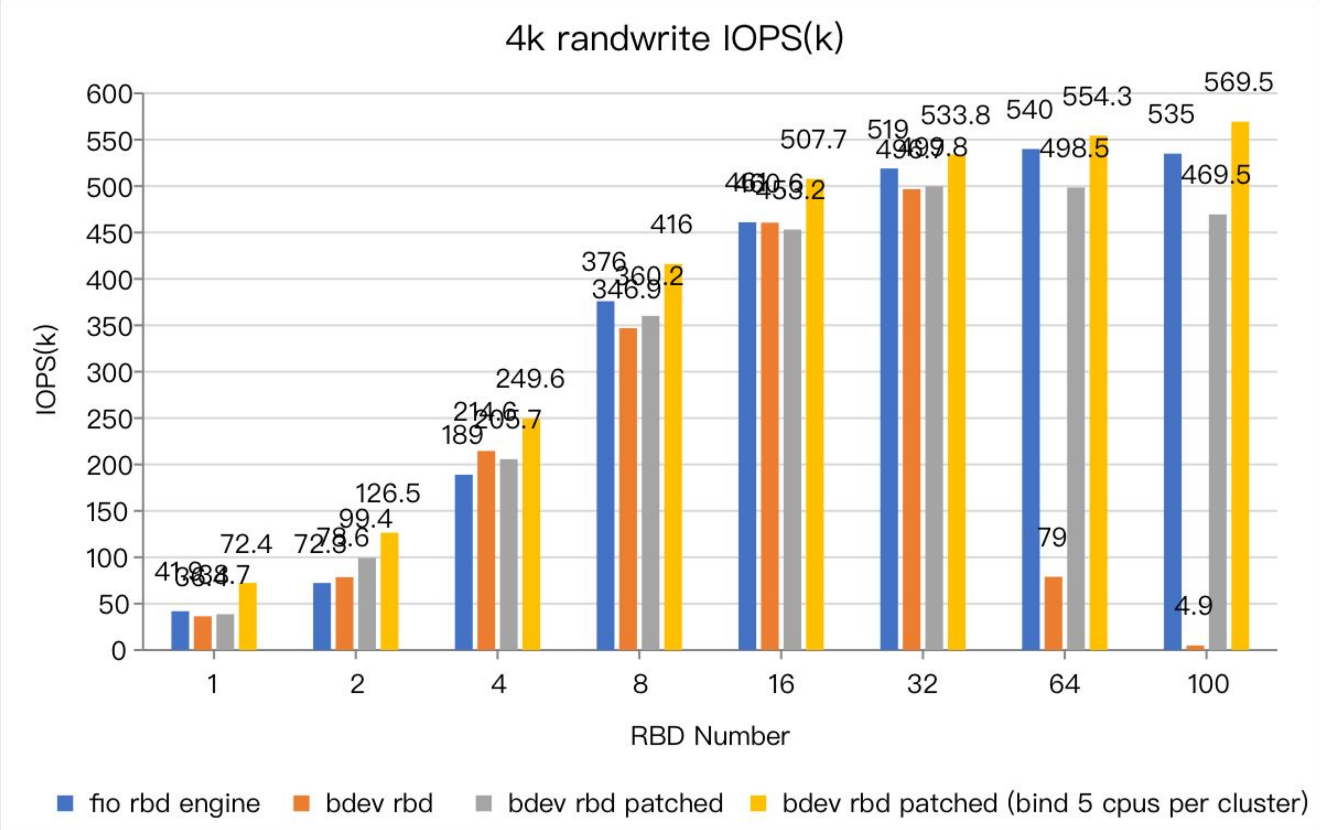
```
Samples: 1M of event 'cycles:P', 4000 Hz, Event count (approx.): 827427613819 lost: 0/0 drop: 0/0
Overhead  Shared Object                 Symbol
  68.16%  bdevperf                      [.] common_ring_mc_dequeue
   5.14%  bdevperf                      [.] spdk_ring_dequeue
   2.17%  bdevperf                      [.] spdk_thread_poll
   1.91%  [kernel]                      [k] finish_task_switch.isra.0
   1.86%  bdevperf                      [.] reactor_run
   1.70%  bdevperf                      [.] thread_poll
   1.25%  bdevperf                      [.] msg_queue_run_batch
   0.88%  libceph-common.so.2           [.] __aarch64_ldadd8_acq_rel
   0.66%  libc.so.6                     [.] _int_free
   0.63%  [kernel]                      [k] handle_softirqs
```

Ceph SPDK NVMe-oF Gateway性能测试 - multi-rbd test issue

Issue2: 4k randwrite/randread IOPS drops rapidly as the RBD number grows large

Possible fix ways:

1. Revert commit “62210eff55bd bdev/rbd: Remove epoll based group polling mechanism.”
2. Or, don’t use msg mempool here, just use calloc() to allocate msg
   - ○ rbd-bdev-drop-rapidly-fix.patch seems work, still under disscussing
   - ○ Need to handle msg release properly in msg_queue_run_batch()



Ceph SPDK NVMe-oF Gateway性能测试 - Summary

Summary

- Ceph main requires revert commit "nvmeof gw monitor: disable by default".
- Ceph nvmeof gateway requires rebuilding to non-debug release.
- Build LSE atomic instructions for Arm64.
- RBD Preconditioning: sequential write 2X+ size of rbd to get SSD steady state.
- Add spdk cores when spdk_top cpu% is full
- Bind and tune Ceph client, spdk tgt threads, bdevs_per_cluster=1.
- DPDK mempool is not working well for many non-EAL threads.
- Useful Links
  - https://spdk.io/cn/articles/
  - https://spdk.io/doc/
  - https://www.youtube.com/@storageperformancedevelopm301

THANKS

开放原子开源基金会
OPENATOM FOUNDATION | openEuler