SAMSUNG

All-Flash NVMe(PM1725a) Reference Architecture

Red Hat Ceph Storage 3.2 with BlueStore

Table of contents

Introduction and Ceph overview

Executive Summary

This document introduces Samsung's NVMe SSD Reference Architecture for providing optimal performance in Red Hat ® Ceph storage with Samsung PM1725a NVMe SSD on an x86 architecture-based storage cluster. It also provides an optimized configuration for Ceph clusters and their performance benchmark results.

Samsung configured five storage nodes based on all-flash with PM1725a NVMe SSD, resulting in 4 KB random read performance surpassing 2 million IOPS. The table below shows the results for the performance and latency evaluated in this Reference Architecture. In the sections that follow, we will cover the details of the Ceph and Samsung Reference Architecture.

4 KB Random Workload		
	Write	Read
Avg. Throughput (KIOPS)	493	2255
Avg. 99.99%th Latency (ms)	74.20	153.21
Avg. Latency (ms)	12.97	2.85

Table 1: Summary of 4 KB random workload

128 KB Sequential Workload		
	Write	Read
Avg. Throughput (GB/s)	18.8	51.6

Table 2: Summary of 128 KB sequential workload

Introduction

Enterprise storage infrastructure and related technologies continue to evolve year after year. In particular, as IoT, 5G, AI, and ML technologies are gaining attention, the demand for SDS (software-defined storage) solutions based on clustered storage servers is also increasing. Ceph has emerged as a leading SDS solution that takes on high performance intensive workloads. Therefore, high throughput and low latency features of storage devices are important factors that improve the overall performance of the Ceph cluster. Adoption of a Ceph cluster on a NVMe SSD will maximize performance improvement. So, Samsung designed Ceph clusters based on all-flash NVMe SSDs and conducted various tests to provide Ceph users with optimized Ceph configurations.

Ceph distributed architecture overview

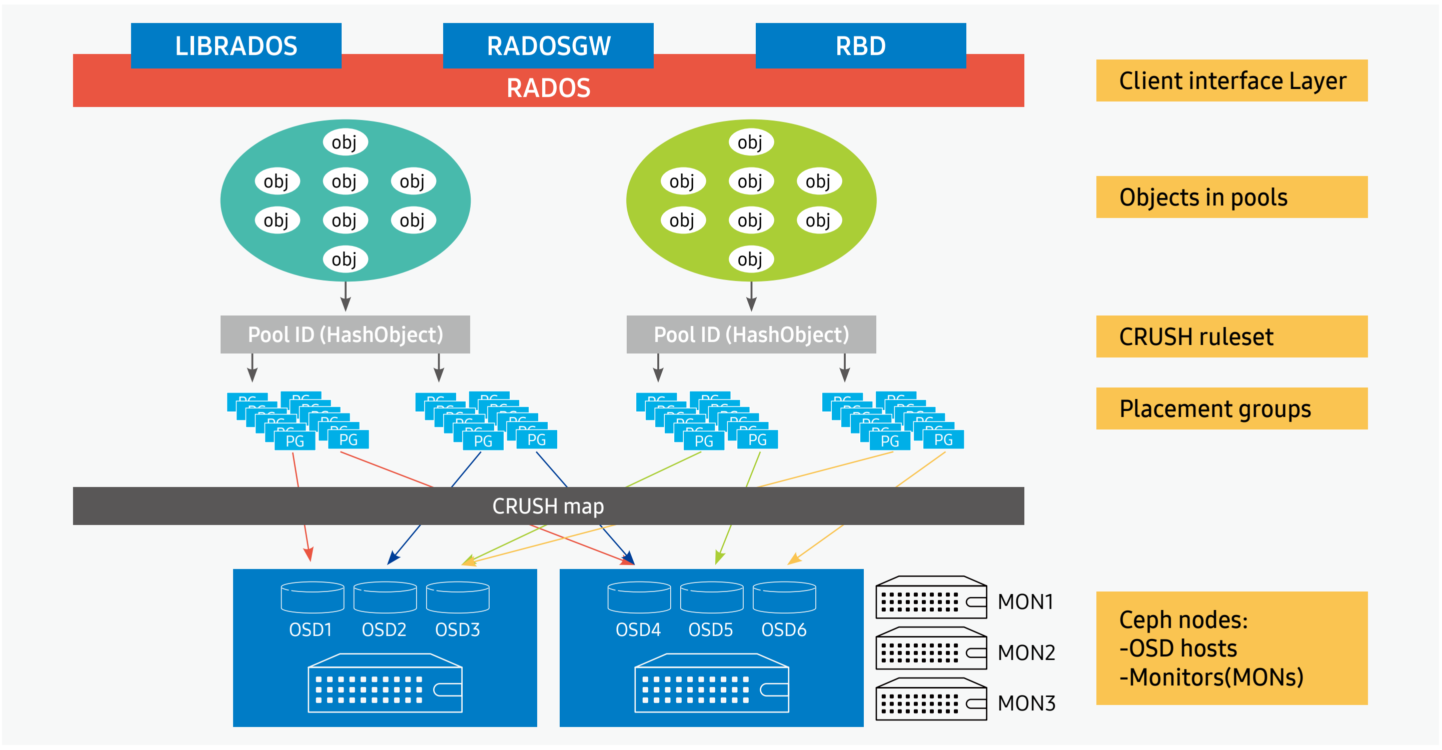
A Ceph storage cluster is built from large numbers of Ceph nodes for scalability, fault tolerance, and performance. Each node is based on commodity hardware and uses intelligent Ceph daemons that communicate with each other to:

- Store and retrieve data
- Replicate data
- Monitor and report on cluster health
- Redistribute data dynamically (remap and backfill)
- Ensure data integrity (scrubbing)
- Detect and recover from faults and failures

From the Ceph client interface where reads and writes operations are shown, a Ceph storage cluster looks like a simple pool where data is stored. However, the storage cluster performs many complex operations in a way that is completely transparent at the client interface level. Ceph clients and Ceph object storage daemons (Ceph OSD daemons or OSDs) both use the CRUSH (controlled replication under scalable hashing) algorithm for storage and retrieval of objects.

From the Ceph client standpoint, the storage cluster is very simple. When a Ceph client reads or writes data, it connects to a logical storage pool in the Ceph cluster. Figure 1 illustrates the overall Ceph architecture, featuring concepts that are described in the sections that follow.

Introduction and Ceph overview



Source : Red Hat Ceph Architecture Overview
Figure 1: Clients write to Ceph storage pools while the CRUSH ruleset determines how placement groups are distributed across OSDs(object storage daemons)

- Pools: A Ceph storage cluster stores data objects in logical dynamic partitions called pools. Pools can be created for particular data types, such as for block devices, object gateways, or simply to separate user groups. The Ceph pool configuration dictates the number of object replicas and the number of PGs (placement groups) in the pool. Ceph storage pools can be either replicated or erasure coded, according to the application and cost model. Additionally, pools can “take root” at any position in the CRUSH hierarchy, allowing placement on groups of servers with differing performance characteristics—allowing storage to be optimized for different workloads.
- Placement groups: Ceph maps objects to PGs (placement groups). PGs are shards or fragments of a logical object pool that are composed of a group of Ceph OSD daemons that are in a peering relationship. Placement groups provide a means of creating replication or erasure coding groups of coarser granularity than on a per object basis. A larger number of placement groups (e.g., 200 per OSD or more) leads to better balancing.
- CRUSH ruleset: The CRUSH algorithm provides controlled, scalable, and declustered placement of replicated or erasure-coded data within Ceph and determines how to store and retrieve data by computing data storage locations. CRUSH empowers Ceph clients to communicate with OSDs directly, rather than through a centralized server or broker. By determining the method for storing and retrieving data by an algorithm, Ceph avoids single points of failure, performance bottlenecks, and physical limits to scalability.
- Ceph monitors (MONs): Before Ceph clients can read or write data, they must contact a Ceph MON to obtain the current cluster map. A Ceph storage cluster can operate with a single monitor, but this introduces a single point of failure. For added reliability and fault tolerance, Ceph supports an odd number of monitors in a quorum (typically three or five for small to mid-sized clusters). Consensus among various monitor instances ensures consistent knowledge about the cluster’s state.
- Ceph OSD daemons: In a Ceph cluster, Ceph OSD daemons store data and handle data replication, recovery, backfilling, and rebalancing. They also provide some cluster state information to Ceph monitors by checking other Ceph OSD daemons with a heartbeat mechanism. A Ceph storage cluster that is configured to keep three replicas of every object requires a minimum of three Ceph OSD daemons, two of which need to be operational to successfully process write requests. A rough equivalent to Ceph OSD daemons is a file system on a physical disk drive.

- Red Hat Ceph architecture overview : https://www.redhat.com/en/resources/resources-how-configure-red-hat-ceph-storage-html

Ceph BlueStore

BlueStore is a new back end for the OSD daemons. Unlike the original FileStore back end, BlueStore stores objects directly on the block devices without any file system interface. This improves the performance of the cluster.

The following are some of the main features of using BlueStore:

- Direct management of storage devices: BlueStore consumes raw block devices or partitions. This prevents any intervening layers of abstraction, such as local file systems like XFS, that might limit performance or add complexity.
- Metadata management with RocksDB: BlueStore uses the RocksDB key-value database to manage internal metadata, such as the mapping from object names to block locations on a disk.
- Full data and metadata checksumming: By default, all data and metadata written to BlueStore is protected by one or more checksums. No data or metadata are read from disk or returned to the user without verification.
- Efficient copy-on-write: The Ceph Block Device and Ceph File System snapshots rely on a copy-on-write clone mechanism that is efficiently implemented in BlueStore. This results in efficient I/O both for regular snapshots and for erasure coded pools that rely on cloning to implement efficient two-phase commits.
- No large double-writes: BlueStore first writes any new data to unallocated space on a block device, and then commits a RocksDB transaction that updates the object metadata to reference the new region of the disk. It only falls back to a write-ahead journaling scheme when the write operation is below a configurable size threshold, similar to how FileStore operates.
- Multi-device support: BlueStore can use multiple block devices for storing different data. For example, HDD for the data, SSD for metadata, NVM (non-volatile memory), or NVRAM (non-volatile random-access memory) or persistent memory for the RocksDB WAL (write-ahead log).

- Bluestore : https://access.redhat.com/documentation/en-us/red_hat_ceph_storage/3/html/administration_guide/osd-bluestore

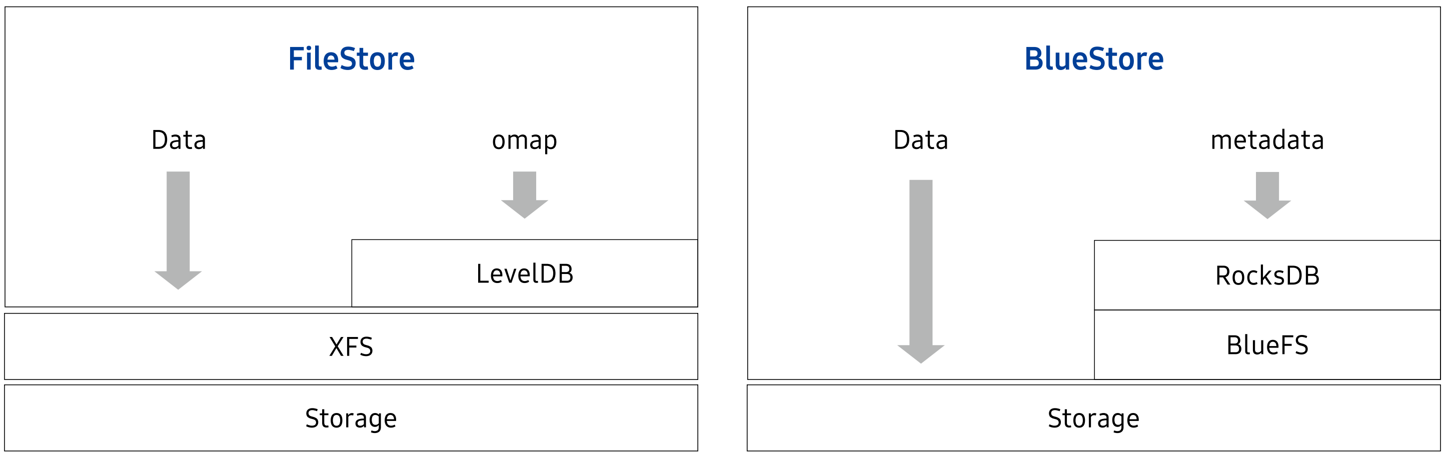


Figure 2: Comparing the structure of FileStore and BlueStore

Introducing the Samsung PM1725a NVMe SSD

The Samsung PM1725a SSD is optimized to excel in virtually any data center scenario. This enterprise-level, ultra-high performance SSD provides excellent random read performance and is particularly suitable for read-intensive data center applications. When compared with other standardized Samsung SSDs, the PM1725a SSD provides high random read IOPS performance. The PM1725a features Samsung 3rd-generation TLC V-NAND technology, which significantly reduces performance and reliability issues caused by the capacity limitations of planar NAND technologies. The Samsung V-NAND technology delivers reliable and consistent performance at lower costs for today's demanding data-centric world.

Samsung PM1725a NVMe SSD delivers:

- Exceptional value: The PM1725a utilizes Samsung V-NAND flash memory and employs cost-effective TLC (triple-level cell) flash memory, which delivers higher reliability over MLC (multi-level cell) planar NAND flash memory SSDs. With this architecture, the PM1725a can economically deliver enterprise-level performance.
- Consistently high performance: The PM1725a HHHL (half-height, half-length) card delivers a wide bandwidth of up to 6,200/2,600 MB/s sequential R/W speeds respectively, using under 23 W of power. It delivers up to 1,000K and 180K IOPS for random 4 KB read and write, respectively. The PM1725a 2.5-inch SSD delivers a bandwidth of up to 3,300/2,600 MB/s for sequential R/W and up to 800K/180K IOPS for random 4 KB R/W, respectively.
- Outstanding reliability: Although the PM1725a employs TLC V-NAND flash memory, it is capable of 5 DWPD (drive writes per day) for 5 years. This rate translates to writing a total of 32 TB each day during that time, which means users can write 6,400 files of 5 GB-equivalent data every day. This level of reliability is more than sufficient for enterprise storage systems that have to perform ultrafast transmissions of large amounts of data.
- High density: By fitting more memory into a V-NAND chip, it provides significantly more capacities of up to 6.4 TB in both the PM1725a 2.5-inch and HHHL card SSDs. Depending on your storage requirements and applications, 800 GB, 1.6 TB, 3.2 TB, and 6.4 TB capacities are available.

Samsung PM1725a NVMe SSD		
Form factor	2.5 inch	HHHL
Capacity	800 GB, 1.6 TB, 3.2 TB, 6.4 TB	1.6 TB, 3.2 TB, 6.4 TB
Host interface	PCI Express Gen3 x4, NVMe	PCI Express Gen3 x8, NVMe
Sequential read	Up to 3,300 MB/s	Up to 6,200 MB/s
Sequential write	Up to 2,600 MB/s	Up to 2,600 MB/s
Random read	Up to 800,000 IOPS	Up to 1,000,000 IOPS
Random write	Up to 180,000 IOPS	Up to 180,000 IOPS
NAND flash memory	Samsung V-NAND	
MTBF	2,000,000 hours	
Endurance	5 DWPD for 5 years	

Table 3: Samsung PM1725a NVMe SSD specifications

- Sequential performance measured using FIO with 128 KB block size, queue depth 32, number of jobs 1.
- Random performance measured using FIO with 4 KB block size, queue depth 32, number of jobs 8.
- Actual performance may vary depending on use conditions and environment

Samsung NVMe SSD Reference Architecture

Samsung designed this Reference Architecture to maximize and optimize the performance of all-flash based Ceph clusters. This Reference Architecture consists of five storage nodes and is based on all-flash NVMe SSDs with Samsung PM1725a SSDs, which provide ultra-high performance. Since Ceph clusters involve complex and various parameters, Samsung has applied optimal parameters to maximize its performance. In this Reference Architecture, a Ceph cluster was connected with 100 GbE two separated networks (public and cluster) to communicate with each other. Figure 3 provides an overall design of the Reference Architecture, and following sections cover the details.

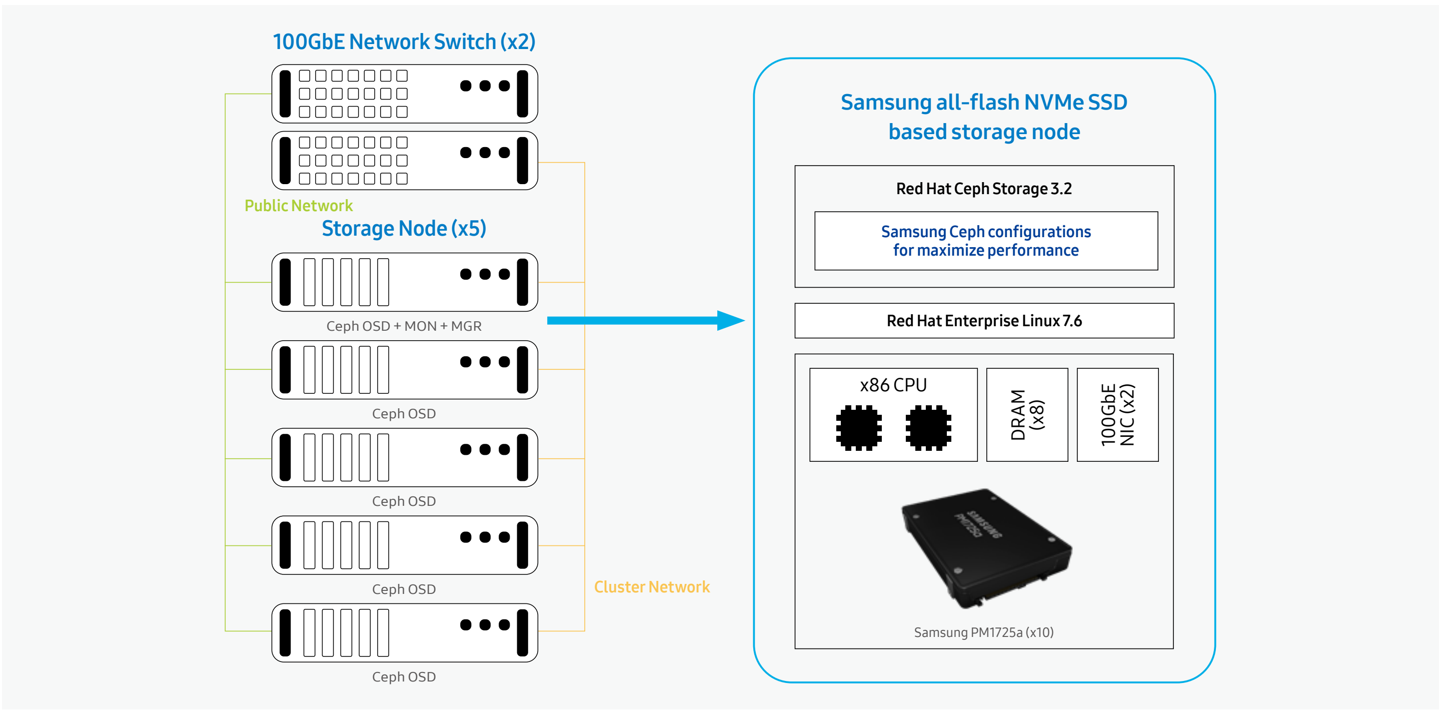


Figure 3: Samsung NVMe SSD Reference Architecture for Ceph cluster
Copyright © 2020 Samsung Electronics Co., Ltd. All Rights Reserved.

Software

Red Hat® Ceph Storage

Designed for the cloud, Red Hat Ceph Storage significantly reduces the cost of storing enterprise data and helps to manage exponential data growth—efficiently and automatically. Delivered in a self-healing, self-managing platform, Red Hat Ceph Storage handles data management so that administrators can focus on improving data availability for business. The key benefits are:

- Value
  Significantly lowers the cost of storing data. Lays a foundation for managing its exponential growth at a low cost per gigabyte.
- Enterprise readiness
  Integrates tightly with OpenStack® and provides advanced block storage capabilities that work just like a traditional block storage device but with hardware flexibility and massive scalability.
- Longevity
  Starts with block storage and grows into object storage, or vice versa. Integrates with existing storage infrastructure easily.
- Expert backed
  Takes advantage of the expertise of Ceph's creators and primary sponsors through professional services and training.

Red Hat Enterprise Linux

With Red Hat® Enterprise Linux®, a platform with unparalleled stability and flexibility, businesses can reallocate infrastructure resources toward meeting their next challenges instead of just maintaining the status quo.

The key benefits are:

- Freedom through stability
  Business applications require a tested, proven, and predictable platform. Red Hat Enterprise Linux frees IT personnel to deliver meaningful business results by providing exceptional reliability and security.
- An ecosystem of solutions and support
  With a Red Hat Enterprise Linux subscription, administrators are connected to the industry's largest ecosystem of partners, customers, and experts that support and accelerate success for organizations.
- Confidence through flexibility
  Red Hat Enterprise Linux gives you the flexibility to tailor your infrastructure for business needs now and in the future. As markets shift and technologies evolve, you'll have the agility, adaptability, and performance to succeed.

Ceph optimized configurations

This Reference Architecture is based on Red Hat Ceph Storage 3.2 (Luminous 12.2.8). It introduces a BlueStore for new backend storage and it is used by default when deploying new OSDs. In addition, Ceph has various parameters and performance can vary depending on how it is set up. All tests of Ceph cluster were conducted based on the below configurations. And the appendix at the end of this document is the various parameters that are tuned and used in Ceph clusters for this Reference Architecture.

Details	
Ceph version	Red Hat Ceph Storage 3.2 (Luminous 12.2.8)
OSD backend storage	BlueStore
# of OSD daemon	80
OSDs per NVMe SSD	2
Replication factor	2x
# of PG (Placement Group)	8192
Samsung Ceph parameters	ceph.conf, osds.yml, all.yml (Appendix)

Table 4: Ceph configurations in Samsung Reference Architecture

Hardware

Samsung PM1725a NVMe SSD

Samsung PM1725a NVMe SSD delivers exceptional value, consistently high performance, outstanding reliability, and high density, putting an end to bottlenecks in performance and efficiency. In this Reference Architecture, each storage node was based on an all-flash NVMe SSD using Samsung PM1725a.

Ceph storage nodes

Ceph storage nodes use x86 Intel Xeon architecture platforms with Samsung PM1725a and additional storage for local operating systems. These nodes must have a NIC (Network Interface Card) to connect with Ceph clients, monitors, and OSDs.

Storage nodes (x5)	
Processor (x2)	Intel® Xeon® Gold 6152 CPU 2.10 GHz
DRAM (x8)	Samsung 16 GB DDR4-2400 MT/s, (128 GB per node)
NVMe SSD (x10)	Samsung PM1725a NVMe SSD, 1.6 TB , 2.5 inch form factor
SATA SSD (x1)	Samsung 850 EVO SATA SSD
NIC (x1)	Mellanox ConnectX®-5 MCX516A-CCAT Dual-Port adapter (100 GbE)

Table 5: Ceph storage node details

Networks

Ceph storage nodes configure two separated networks. One is used as the public network (Monitors and clients), and other as the cluster network (Heart-beating, replication, peering, recovery). We use 100 GbE Mellanox ConnectX®-5 NICs with 100 GbE Mellanox MSN2700-CS2F Network switches for throughput-intensive workloads and performance-optimized Ceph clusters.

Network devices	
Network Switch (x2)	Mellanox MSN2700-CS2F 1U Ethernet switch (100 GbE)
NIC (x10)	Mellanox ConnectX®-5 MCX516A-CCAT Dual-Port adapter (100 GbE)

Table 6: Network device details

Operational planning considerations

This section presents general guidance on operational planning for deploying high-performance Ceph storage clusters using Samsung NVMe SSDs.

- Storage nodes: It is recommended that at least three storage nodes be deployed to a Ceph cluster for redundancy and high availability.
- Monitor nodes: It is recommended that an odd number of monitor nodes be deployed. Each Ceph cluster can run one monitor node, but three or more monitors are used in a production cluster. Monitor nodes do not need to have high performance CPUs; they would benefit from high performance SSDs to store monitor map data.
- Replication factor: Given the better MTBF and MTTR of flash-based media, many Ceph customers have chosen to run 2x replications in production when deploying OSDs on flash. This differs from magnetic media deployments, which typically use 3x replication.
- CPU sizing: Ceph OSDs intensively uses CPU to calculate data placement and dynamically redistributed their load. Therefore, a higher CPU core count results in higher performance in Ceph for IO intensive workloads.
- OSDs per SSD: An appropriate number of OSDs should be used, as the number of OSDs per SSD affects the performance. Too many OSDs per SSD may negatively affect performance due to increased context switching costs.
- Networking: At least one 10 GbE cluster is required to gain the performance benefits of NVMe SSD based Ceph cluster. For throughput-oriented workloads, 50 GbE to 100 GbE is recommended. In accordance with standard Ceph recommendations, it is also recommended that a Ceph storage cluster be run with two networks (a public network and a cluster network).
- OS Tunings: The Ceph IO path traverses through several kernel modules in the Linux stack. The default values of these respective modules will not be the best fit for a Ceph configuration optimized for performance.

Baseline test results

The purpose of this test is to measure the pure IO performance of the storage at each node where the Ceph package is not installed. Each node has a Samsung PM1725a NVMe SSD, and their performance was measured using the Fio (Flexible I/O tester) benchmark tool with libaio IO engine. IOPS performance was evaluated for random IO workloads of a small IO size (4 KB). Sequential performance was also evaluated for sequential IO workloads of a large IO size (128 KB). The test was performed three times, and the results were averaged. Tables 7 and 8 below show the baseline test results.

- Fio options for random workload: Number of jobs - 8, Queue depth - 32, IO engine - libaio
- Fio options for sequential workload: Number of jobs - 1, Queue depth - 32, IO engine - libaio

4 KB Random Write					
	Node1	Node2	Node3	Node4	Node5
Avg. Throughput (KIOPS)	4826	4760	4776	4772	4731
Avg. 99.99%th Latency (ms)	7.24	7.53	7.66	7.44	7.55
Avg. Latency (ms)	0.53	0.54	0.54	0.54	0.54
4 KB Random Read					
Avg. Throughput (KIOPS)	5590	5537	5465	5655	5462
Avg. 99.99%th Latency (ms)	2.98	2.59	1.94	3.13	1.53
Avg. Latency (ms)	0.46	0.46	0.47	0.45	0.47

Table 7: Baseline result of random tests

128 KB Sequential Write					
	Node1	Node2	Node3	Node4	Node5
Avg. Throughput (GB/s)	19.8	19.6	19.6	19.6	19.6
Avg. 99.99%th Latency (ms)	9.76	10.07	9.63	9.50	9.72
Avg. Latency (ms)	1.97	2.00	2.00	1.99	1.99
128 KB Sequential Read					
Avg. Throughput (GB/s)	29.8	29.6	29.8	29.9	29.9
Avg. 99.99%th Latency (ms)	1.88	1.93	1.97	1.86	1.88
Avg. Latency (ms)	1.31	1.32	1.31	1.31	1.31

Table 8: Baseline result of sequential tests

Benchmark configurations and results

The following sections provide the result of synthetic benchmark performance for all-flash based Ceph clusters using PM1725a NVMe SSD. The test was conducted in the RBD-based (RADOS Block Device) storage pool, which is the block storage component for Ceph. Workloads were generated using the Fio benchmark with 10 client servers. Before starting the test, we created 200 RBD images that generated a total of 15 TB of data. We then applied a 2x replication, resulting in the entire size of the data stored in the cluster being 30 TB.

- 10 Clients x 20 RBD images per client x 75 GB RBD image size = 15 TB (2x Replication: 15 TB x 2 = 30 TB)

A Random test was evaluated for a 4 KB small IO workload with number of jobs 8 and queue depth 32 per Fio instance. A sequential test was evaluated for 128 KB large IO workload with number of jobs 1 and queue depth 32 per Fio instance. We also measured latency variation across each test. The test was performed three times, and the results were averaged.

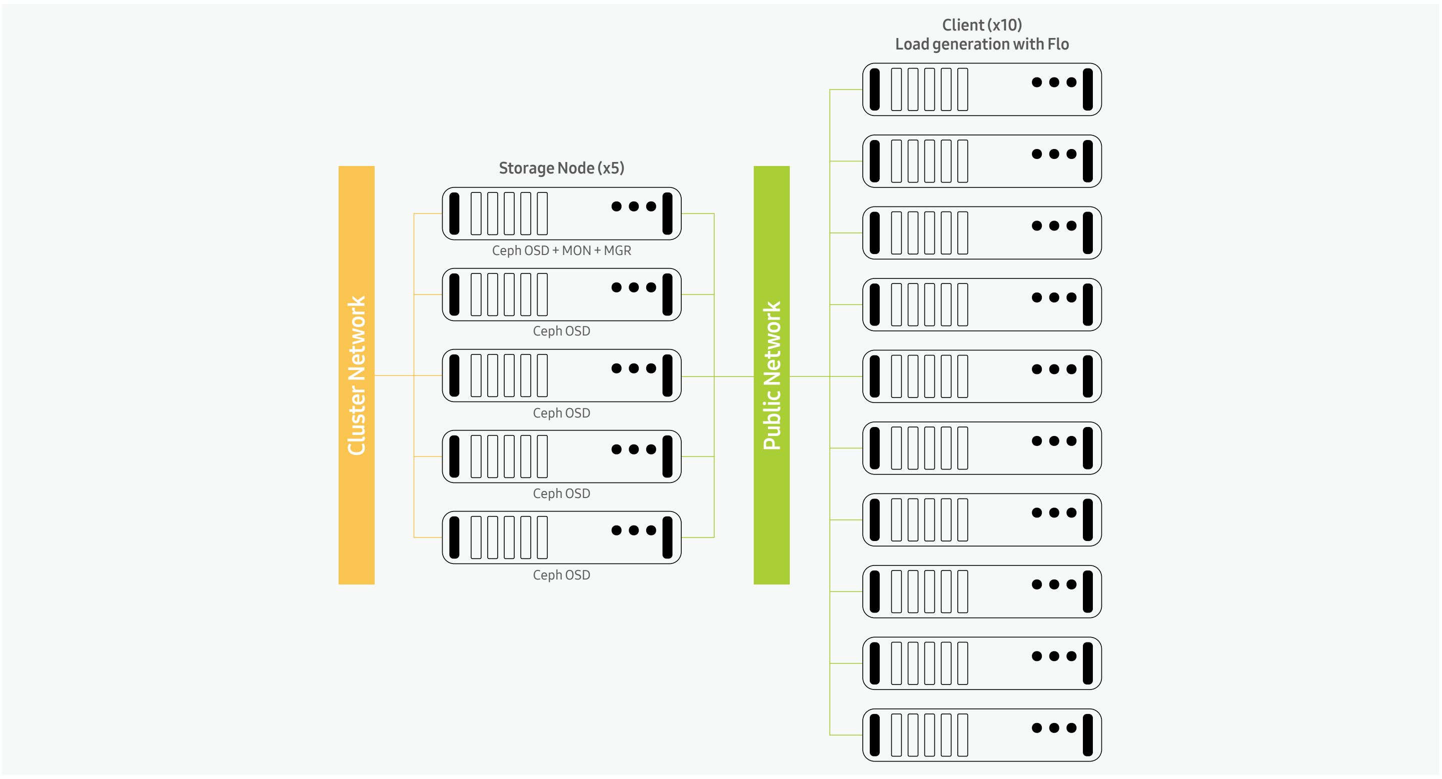


Figure 4: Test environment topology

Configurations and Benchmark results

Details	
Ceph storage type	RBD (RADOS Block Device)
Load generation tool	Fio 3.14
Replication factor	2x
# of client node	10
# of client RBD image	200 (20 RBD images per client)
RBD image size	75 GB
# of monitor daemon	1
# of manager daemon	1

Table 9: Benchmark configurations

4 KB Random write workload

We measured the performance and latency of 4 KB random writes with increasing queue depths on 200 clients. At a queue depth of 32, 4 KB random write performance was measured at an average of 493K IOPS, with an average latency of 12.97ms and an average tail latency (99.99%th latency) of 74.20ms. As queue depth increased, performance and latency tended to increase. Tail latency (99.99%th latency) increased significantly at queue depths of 16 and higher.

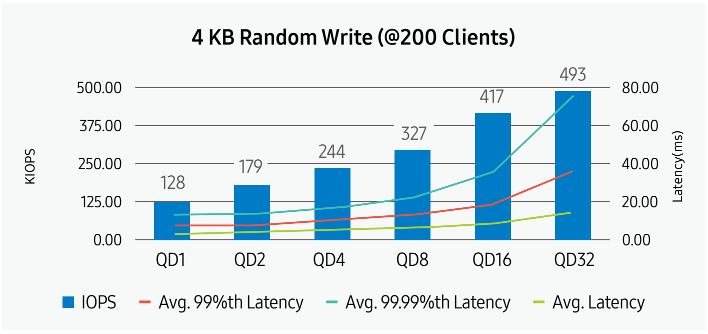


Figure 5: Result of 4 KB random write performance and latency

	QD1	QD2	QD4	QD8	QD16	QD32
Avg. Throughput (KIOPS)	128	179	244	327	417	493
Avg. 99%th Latency (ms)	6.21	6.93	9.49	12.26	17.44	34.59
Avg. 99.99%th Latency (ms)	11.87	12.22	15.44	21.10	34.16	74.20
Avg. Latency (ms)	1.57	2.23	3.28	4.89	7.67	12.97

Table 10: Result of random write test

4 KB Random read workload

We measured the performance and latency of 4 KB random reads with increasing queue depths on 200 clients. At a queue depth of 32, 4 KB random read performance was measured at an average of 2255K IOPS, with an average latency of 2.85ms and a tail latency (99.99%th latency) of 153.21ms. As the queue depths increased, performance and latency tended to increase. Tail latency (99.99%th latency) increased significantly at queue depths of 8 and higher.

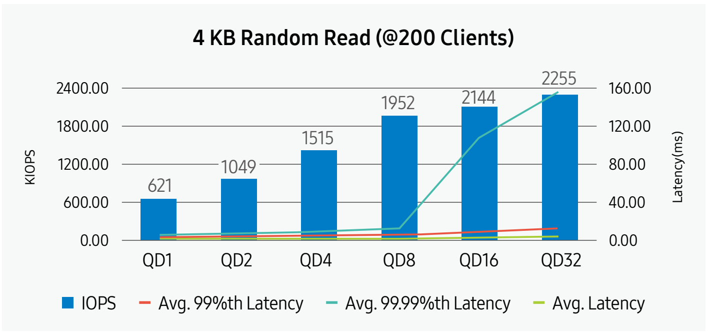


Figure 6: Result of 4 KB random read performance and latency

	QD1	QD2	QD4	QD8	QD16	QD32
Avg. Throughput (KIOPS)	621	1049	1515	1952	2144	2255
Avg. 99%th Latency (ms)	0.48	0.66	1.16	2.60	6.01	10.36
Avg. 99.99%th Latency (ms)	4.36	5.40	7.44	11.42	103.59	153.21
Avg. Latency (ms)	0.32	0.38	0.53	0.82	1.50	2.85

Table 11: Result of random read test

Configurations and Benchmark results

We also measured the variation in random read latency as the total number of clients increased. (Fio benchmark option is set to number of jobs 1, queue depth 1). As shown in Figure 7 below, the tail latency (99.99%th latency) remained within a certain fixed range, even though the total number of clients gradually increased.

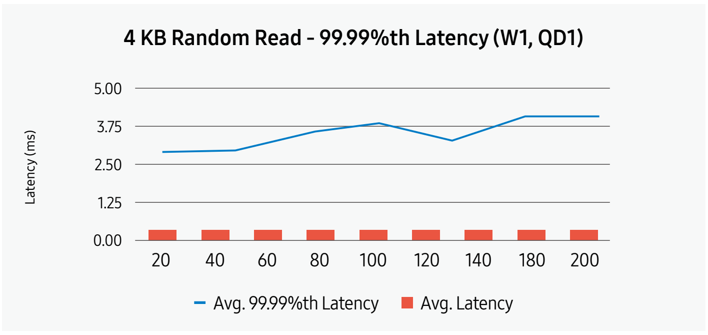


Figure 7: Result of 4 KB Random read latency variation

	20	40	60	80	100	120	140	180	200
Avg. 99.99%th Latency (ms)	3.02	3.11	3.62	3.92	3.41	4.17	4.14	4.30	4.45
Avg. Latency (ms)	0.31	0.28	0.28	0.28	0.27	0.29	0.30	0.31	0.31

Table 12: Latency variation of random read test

128 KB Sequential write workload

Average throughput for 128 KB sequential writes was 18.8 GB/s in 200 clients. Latency increased steadily as the number of clients increased, while throughput remained relatively constant once the number of clients reached 80.

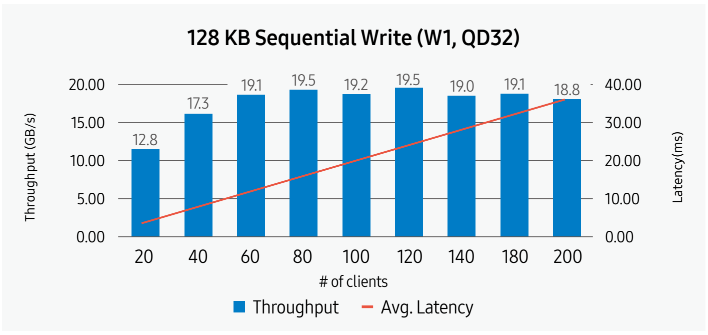


Figure 8: Result of 128 KB Sequential write performance and latency

	20	40	60	80	100	120	140	180	200
Avg. Throughput (GB/s)	12.8	17.3	19.1	19.5	19.2	19.5	19.0	19.1	18.8
Avg. Latency (ms)	6.10	9.05	12.30	16.04	20.31	24.03	28.81	32.76	37.50

Table 13: Result of sequential write test

128 KB Sequential read workload

The average throughput for 128 KB sequential reads was 51.6 GB/s in 200 clients. Latency increased steadily as the number of clients increased, while throughput remained relatively constant once the number of clients reached 100.

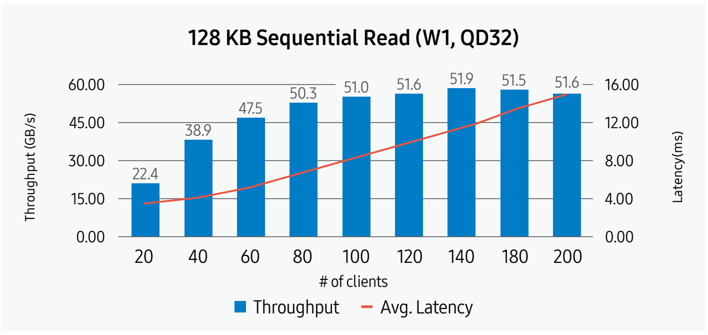


Figure 9: Result of 128 KB Sequential read performance and latency

	20	40	60	80	100	120	140	180	200
Avg. throughput (GB/s)	22.4	38.9	47.5	50.3	51.0	51.6	51.9	51.5	51.6
Avg. Latency (ms)	3.50	4.04	5.02	6.39	7.83	9.39	10.90	12.80	14.22

Table 14: Result of sequential read test

Conclusion

Samsung PM1725a NVMe SSD is optimized for enterprise environments and delivers consistently high performance, making it a perfect solution for software-defined storage such as Red Hat Ceph Storage.
Samsung has designed a performance-optimizing, all-flash based Ceph cluster using PM1725a and Red Hat Ceph Storage, and was able to achieve over 2.2M 4 KB random read performance and excellent sequential performance.

Equipped with its expertise and experience in developing cutting-edge SSD technology, Samsung's Memory Division is committed to supporting your data centers to run at their highest levels of performance.

Appendix

ceph.conf

```
[client]
rbd_cache = False
rbd_cache_writethrough_until_flush = False

# Please do not change this file directly since it is
managed by Ansible and will be overwritten
[global]
auth client required = none
auth cluster required = none
auth service required = none
cluster network = 192.168.44.0/24
debug asok = 0/0
debug auth = 0/0
debug bluefs = 0/0
debug bluestore = 0/0
debug buffer = 0/0
debug client = 0/0
debug context = 0/0
debug crush = 0/0
debug filer = 0/0
debug filestore = 0/0
debug finisher = 0/0
debug hadoop = 0/0
debug heartbeatmap = 0/0
debug journal = 0/0
debug journaler = 0/0
debug lockdep = 0/0
debug log = 0/0
debug mds = 0/0
debug mds_balancer = 0/0
debug mds_locker = 0/0
debug mds_log = 0/0
debug mds_log_expire = 0/0
debug mds_migrator = 0/0
debug mon = 0/0
debug monc = 0/0
debug ms = 0/0
debug objclass = 0/0
debug objectcacher = 0/0
debug objecter = 0/0
debug optracker = 0/0
debug osd = 0/0
debug paxos = 0/0
debug perfcounter = 0/0
debug rados = 0/0
debug rbd = 0/0
debug rgw = 0/0
debug rocksdb = 0/0
debug throttle = 0/0
debug timer = 0/0
debug tp = 0/0
debug zs = 0/0
fsid = b229189e-ef00-493e-97ce-80f9106ef653
mon pg warn max per osd = 1600
mon_allow_pool_delete = True
mon_host = 192.168.43.8
mon_initial_members = ceph-node8
mon_max_pg_per_osd = 1600
ms_crc_data = False
ms_crc_header = True
ms_type = async
osd objectstore = bluestore
osd_pool_default_size = 2
osd_pool_min_size = 1
perf = True
public network = 192.168.43.0/24
rocksdb_perf = True

[mon]
mon_allow_pool_delete = True
mon_health_preluminous_compat = True
mon_max_pool_pg_num = 166496
mon_osd_down_out_interval = 300

[osd]
bluestore_cache_autotune = 0
bluestore_cache_kv_max = 200G
bluestore_cache_kv_ratio = 0.2
bluestore_cache_meta_ratio = 0.8
bluestore_cache_size_ssd = 32G
bluestore_csum_type = none
bluestore_extent_map_shard_max_size = 200
bluestore_extent_map_shard_min_size = 50
bluestore_extent_map_shard_target_size = 100
bluestore_prefer_deferred_size = 0
bluestore_rocksdb_options =
compression=kNoCompression,max_write_
buffer_number=64,min_write_buffer_number_to_
merge=32,recycle_log_file_num=64,compaction_
style=kCompactionStyleLevel,write_buffer_
size=4MB,target_file_size_base=4MB,max_
background_compactions=64,level0_file_num_
compaction_trigger=64,level0_slowdown_writes_
trigger=128,level0_stop_writes_trigger=256,max_
bytes_for_level_base=6GB,compaction_
threads=32,flusher_threads=8,compaction_
readahead_size=2MB
osd memory target = 5887072665
```

SAMSUNG

Appendix

```
osd_map_share_max_epochs = 100
osd_max_backfills = 5
osd_max_pg_log_entries = 10
osd_memory_target = 10737418240
osd_min_pg_log_entries = 10
osd_op_num_shards = 8
osd_op_num_threads_per_shard = 2
osd_pg_log_dups_tracked = 10
osd_pg_log_trim_min = 10
```

osds.yml

```
dummy:
osd_scenario: lvm
lvm_volumes:
- data : lvm0
data_vg : vg_data0
wal :  lvm0
wal_vg : vg_wal
db : lvm0
db_vg : vg_db
- data : lvm0
data_vg : vg_data1
wal :  lvm1
wal_vg : vg_wal
db : lvm1
db_vg : vg_db
- data : lvm0
data_vg : vg_data2
wal :  lvm2
wal_vg : vg_wal
db : lvm2
db_vg : vg_db
- data : lvm0
data_vg : vg_data3
wal :  lvm3
wal_vg : vg_wal
db : lvm3
db_vg : vg_db
- data : lvm0
data_vg : vg_data4
wal :  lvm4
wal_vg : vg_wal
db : lvm4
db_vg : vg_db
- data : lvm0
data_vg : vg_data5
wal :  lvm5
wal_vg : vg_wal
db : lvm5
db_vg : vg_db
- data : lvm0
data_vg : vg_data6
wal :  lvm6
wal_vg : vg_wal
db : lvm6
db_vg : vg_db
- data : lvm0
data_vg : vg_data7
wal :  lvm7
wal_vg : vg_wal
db : lvm7
db_vg : vg_db
- data : lvm1
data_vg : vg_data0
wal :  lvm8
wal_vg : vg_wal
db : lvm8
db_vg : vg_db
- data : lvm1
data_vg : vg_data1
wal :  lvm9
wal_vg : vg_wal
db : lvm9
db_vg : vg_db
- data : lvm1
data_vg : vg_data2
wal :  lvm10
wal_vg : vg_wal
db : lvm10
db_vg : vg_db
- data : lvm1
data_vg : vg_data3
wal :  lvm11
wal_vg : vg_wal
db : lvm11
db_vg : vg_db
- data : lvm1
data_vg : vg_data4
wal :  lvm12
wal_vg : vg_wal
db : lvm12
db_vg : vg_db
- data : lvm1
data_vg : vg_data5
wal :  lvm13
wal_vg : vg_wal
db : lvm13
db_vg : vg_db
- data : lvm1
data_vg : vg_data6
wal :  lvm14
wal_vg : vg_wal
db : lvm14
db_vg : vg_db
- data : lvm1
data_vg : vg_data7
wal :  lvm15
wal_vg : vg_wal
db : lvm15
db_vg : vg_db
```

all.yml

```
dummy:
fetch_directory: ~/ceph-ansible-
keys
mon_group_name: mons
osd_group_name: osds
client_group_name: clients
mgr_group_name: mgrs
ceph_repository_type: iso
ceph_repository: rhcs
ceph_rhcs_version: 3
ceph_rhcs_iso_path: "{{ ceph_
stable_rh_storage_iso_path |
default('/home/cephtest/rhceph-
3.2-rhel-7-x86_64.iso') }}"
ceph_rhcs_mount_path: "{{ ceph_
stable_rh_storage_mount_path |
default('/tmp/rh-storage-mount') }}"
ceph_rhcs_repository_path:
"{{ ceph_stable_rh_storage_
repository_path | default('/tmp/rh-
storage-repo') }}" # where to copy
iso's content
fsid: "{{ cluster_uuid.stdout }}"
generate_fsid: true
monitor_interface: p1p2
ceph_conf_overrides:
global:
mon_initial_members: ceph-node8
mon_host: 192.168.43.8
auth client required: none
auth cluster required: none
auth service required: none
cluster network: 192.168.44.0/24
public network: 192.168.43.0/24
ms_type: async
ms_crc_data: False
ms_crc_header: True
debug asok: 0/0
debug auth: 0/0
debug bluefs: 0/0
debug bluestore: 0/0
debug buffer: 0/0
debug client: 0/0
debug context: 0/0
debug crush: 0/0
debug filer: 0/0
debug filestore: 0/0
debug finisher: 0/0
debug hadoop: 0/0
debug heartbeatmap: 0/0
debug journal: 0/0
debug journaler: 0/0
debug lockdep: 0/0
debug log: 0/0
debug mds: 0/0
debug mds_balancer: 0/0
debug mds_locker: 0/0
debug mds_log: 0/0
debug mds_log_expire: 0/0
debug mds_migrator: 0/0
debug mon: 0/0
debug monc: 0/0
debug ms: 0/0
debug objclass: 0/0
debug objectcacher: 0/0
debug objecter: 0/0
debug optracker: 0/0
debug osd: 0/0
debug paxos: 0/0
debug perfcounter: 0/0
debug rados: 0/0
debug rbd: 0/0
debug rgw: 0/0
debug rocksdb: 0/0
debug throttle: 0/0
debug timer: 0/0
```

Appendix

```
debug tp: 0/0
debug zs: 0/0
mon pg warn max per osd: 1600
mon_allow_pool_delete: True
mon_max_pg_per_osd: 1600
osd objectstore: bluestore
osd_pool_default_size: 2
osd_pool_min_size: 1
perf: True
rocksdb_perf: True

mon:
mon_allow_pool_delete: True
mon_health_preluminous_compat: True
mon_osd_down_out_interval: 300
mon_max_pool_pg_num: 166496

osd:
osd_min_pg_log_entries: 10
osd_max_pg_log_entries: 10
osd_memory_target: 10737418240
osd_pg_log_dups_tracked: 10
osd_pg_log_trim_min: 10
bluestore_cache_kv_max: 200G
bluestore_cache_kv_ratio: 0.2
bluestore_cache_meta_ratio: 0.8
bluestore_cache_size_ssd: 32G
bluestore_csum_type: none
bluestore_extent_map_shard_max_size: 200
bluestore_extent_map_shard_min_size: 50
bluestore_extent_map_shard_target_size: 100
bluestore_prefer_deferred_size: 0
bluestore_cache_autotune: 0
osd_map_share_max_epochs: 100
osd_max_backfills: 5
osd_op_num_shards: 8
osd_op_num_threads_per_shard: 2
bluestore_rocksdb_options:
compression=kNoCompression,max_write_
buffer_number=64,min_write_buffer_number_to_
merge=32,recycle_log_file_num=64,compaction_
style=kCompactionStyleLevel,write_buffer_
size=4MB,target_file_size_base=4MB,max_
background_compactions=64,level0_file_num_
compaction_trigger=64,level0_slowdown_writes_
trigger=128,level0_stop_writes_trigger=256,max_
bytes_for_level_base=6GB,compaction_
threads=32,flusher_threads=8,compaction_
readahead_size=2MB
client:
rbd_cache: false
rbd_cache_writethrough_until_flush: false

os_tunning_params:
- { name: fs.aio-max-nr, value: 1048576 }
- { name: kernel.sched_min_granularity_ns, value:
10000000 }
- { name: kernel.sched_wakeup_granularity_ns,
value: 15000000 }
- { name: kernel.pid_max, value: 4194303 }
- { name: fs.file-max, value: 26234859 }
- { name: vm.zone_reclaim_mode, value: 0 }
- { name: vm.swappiness, value: 1 }
- { name: vm.dirty_ratio, value: 10 }
- { name: vm.dirty_background_ratio, value: 5 }
- { name: net.ipv4.tcp_rmem, value: "4096 87380
134217728" }
- { name: net.ipv4.tcp_wmem, value: "4096 65536
134217728" }
- { name: net.core.rmem_max, value: 268435456 }
- { name: net.core.wmem_max, value: 268435456 }
- { name: net.ipv4.tcp_tw_reuse, value: 1 }
- { name: net.ipv4.tcp_fin_timeout, value: 10 }
- { name: net.ipv4.tcp_slow_start_after_idle, value: 0 }
- { name: net.ipv4.conf.all.send_redirects, value: 0 }
- { name: net.ipv4.conf.all.accept_redirects, value: 0 }
- { name: net.ipv4.conf.all.accept_source_route,
value: 0 }
- { name: net.ipv4.tcp_mtu_probing, value: 1 }
- { name: net.ipv4.tcp_timestamps, value: 0 }
- { name: net.core.netdev_max_backlog, value:
50000 }
- { name: net.ipv4.tcp_max_syn_backlog, value:
30000 }
- { name: net.ipv4.tcp_max_tw_buckets, value:
2000000 }
```

Legal and Disclaimer

※ Evaluation results may vary depending on the server environment and settings.

About Samsung Electronics Co., Ltd.

Samsung inspires the world and shapes the future with transformative ideas and technologies. The company is redefining the worlds of TVs, smartphones, wearable devices, tablets, digital appliances, network systems, and memory, system LSI, foundry, and LED solutions. For the latest news, please visit the Samsung Newsroom at news.samsung.com.

Samsung Electronics Co., Ltd.
129 Samsung-ro, Yeongtong-gu, Suwon-si, Gyeonggi-do 16677, Korea **www.samsung.com** 2020-03