SUSE Enterprise Storage v5 & Intel® Cache Acceleration Software Implementation Guide

Supermicro SuperServer Platforms

Contents

Introduction

The objective of this document is to provide guidance for implementing Intel® Cache Acceleration Software (Intel® CAS) and Supermicro NVMe devices with SUSE Enterprise Storage.

Upon completion of the steps in this document, a working SUSE Enterprise Storage (v5) will be operational as described in the SUSE Enterprise Storage 5 Deployment Guide with the addition of Intel® CAS software working to provide performance acceleration for certain workloads.

Target Audience

The target for the content of this document are those architects and administrators who have need of improving certain aspects of performance using Intel's enterprise caching software.

Super Micro Computer, Inc.
980 Rock Avenue
San Jose, CA 95131 USA
www.supermicro.com

Business problem and business value

Business Problem

While SUSE Enterprise Storage delivers a highly scalable, resilient, self-healing storage system designed for large scale environments ranging from hundreds of Terabytes to Petabytes, there is often a need to provide a mechanism to deal with burst activity to the storage cluster. There are several options for meeting this need, however, not all of them are economically viable.

Business value

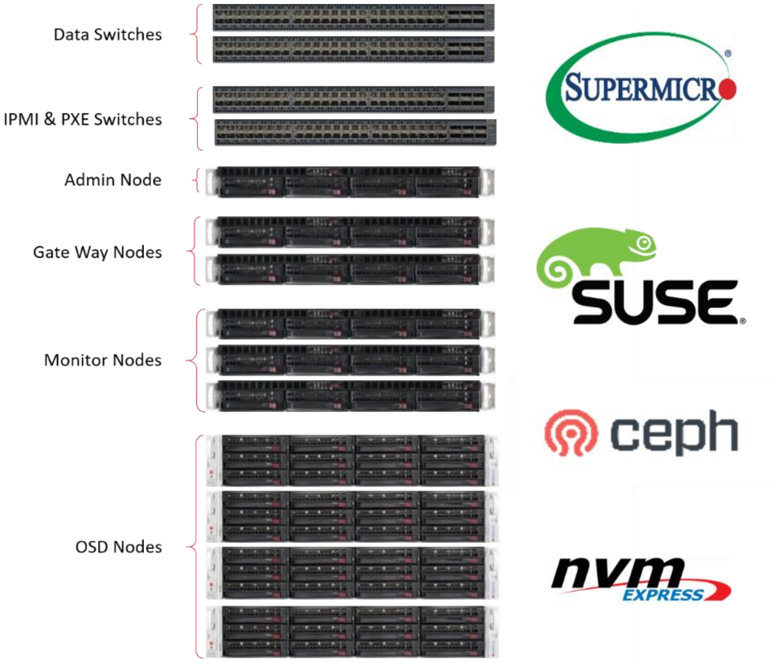
By taking Intel® CAS software and combining with SUSE Enterprise Storage, customers are able to offer caching services with multiple tunable parameters to improve performance in specific environments. While generally applicable to most use cases, the addition of Intel® CAS can provide performance gains with certain workloads to help the customer meet performance needs with a minimal investment and only minor change to the environment.

Requirements

The requirements are not only about performance, but also about maintaining availability and reliability of the environment. It is also important that the caching layer be easily modified to operate in write-back, write-around, write-through, or disabled modes. Intel® CAS offers all of these caching modes, plus the capability to change caching mode on the fly with immediate results in workload performance. It is also ideal that the caching layer can be removed in a non-destructive manner.

Configuration

Intel® CAS was tested with an existing Supermicro cluster. The cluster leveraged three models of Supermicro servers. The role/functionality of each SUSE Enterprise Storage component will be explained in more detail in the architectural overview section. There were two unique server types tested for this reference architecture that can provide the admin, monitor, and protocol gateway functions.



***Admin, Monitor, and Gateway Nodes*:**
2 Supermicro 1U SYS-6019U-TN4RT

- 128GB RAM
- 2 2TB SATA using RAID-1
- 2x Intel(R) Xeon(R) 2630V4 CPU @ 2.20GHz
- Intel Ethernet Controller XL710 for 40GbE QSFP+[1]

***Admin, Monitor, and Gateway Nodes*:**
1 Supermicro 2U SYS-6029TP-HC0R with 4 nodes

- Per Node Configuration
- 128GB RAM
- 2 2TB SATA using RAID-1
- 2x Intel(R) Xeon(R) Gold 5120 CPU @ 2.20GHz
- Intel Ethernet Controller XL710 for 40GbE QSFP+[1]

Storage Nodes:
4 Supermicro **SYS-6029U-E1CR4**

- 128GB RAM
- 2x Intel(R) Xeon(R) CPU E5-2630L v4 @ 1.80GHz
- 2x 2TB SATA in RAID-1
- 10x 8TB SATA
- 2x INTEL Non-Volatile Memory Enterprise (NVMe) Solid State Drive SSDPEDKE020T7 AIC Intel® SSD DC P4600 Series (2.0TB, 1/2 Height PCIe 3.1 x4, 3D1, TLC)[1]
- Intel Ethernet Controller XL710 for 40GbE QSFP+[1]

Switching infrastructure:

1 Supermicro 40GbE/100GbE SDN SuperSwitch[1]

Software:

SUSE Enterprise Storage 5

Intel® CAS software

Please note: The SUSE Enterprise Storage subscription includes a limited use [for SUSE Enterprise Storage] entitlement for SUSE Linux Enterprise Server.

Key Benefits

1. An SSD optimized for cloud storage architectures
2. Optimized for caching across a range of workloads
3. Manageability to maximize IT efficiency
4. Industry-leading reliability and security
5. Designed for today's modern data centers

Performance

- Sequential Read (up to) 3200 MB/s
- Sequential Write (up to) 1575 MB/s
- Random Read (100% Span) 610000 IOPS
- Random Write (100% Span) 196650 IOPS
- Latency - Read 85 µs
- Latency - Write 15 µs
- Power - Active Sequential Avg. 17W (Write), 9.4W (Read)
- Power - Idle <5 W

Reliability

- Vibration - Operating 2.17 GRMS
- Vibration - Non-Operating 3.13 GRMS
- Shock (Operating and Non-Operating) 50 G Trapezoidal, 170 in/s
- Operating Temperature Range 0°C to 35°C
- Endurance Rating (Lifetime Writes) 11.08 PBW
- Mean Time Between Failures (MTBF) 2 million hours
- Uncorrectable Bit Error Rate (UBER) <1 sector per 10^17 bits read
- Warranty Period 5 yrs

Intel® P4600 NVMe Drives

Optimized for Caching Across a Range of Workloads

This cloud-inspired SSD is built with an entirely new NVMe controller that is optimized for mixed workloads commonly found in data caching and is architected to maximize CPU utilization.

With controller support for up to 128 queues, the DC P4600 helps minimize the risk of idle CPU cores and performs most effectively on Intel platforms with Intel® Xeon® processors. The queue pair-to-CPU core mapping supports high drive count and also supports multiple SSDs scaling on Intel platforms.

With the DC P4600, data centers can accelerate caching to enable more users, add more services, and perform more workloads per server. Now you can cache faster and respond faster.

Intel has built industry-leading end-to-end data protection into the DC P4600. This includes protection from silent data corruption, which can cause catastrophic downtime and errors in major businesses.

Power Loss Imminent (PLI) provides protection from unplanned power loss, and is obtained through a propriety combination of power management chips, capacitors, firmware algorithms, and a built-in PLI self-test. Intel's PLI feature provides data centers with high confidence of preventing data loss during unplanned power interruptions.

Designed for Today's Modern Data Centers

The DC P4600 is Intel's new 3D NAND SSD for mixed workloads that are common to the data caching needs of cloud-driven data centers. The mix of performance, capacity, endurance, manageability, and reliability make it the ideal solution for data caching in software-defined and converged infrastructures.

Architectural overview

This architecture overview section complements the SUSE Enterprise Storage Technical Overview document available online which presents the concepts behind software defined storage and Ceph as well as a quick start guide (non-platform specific).

Solution architecture

SUSE Enterprise Storage provides unified block, file, and object access based on Ceph. Ceph is a distributed storage solution designed for scalability, reliability and performance. A critical component of Ceph is the RADOS object storage. RADOS enables a number of storage nodes to function together to store and retrieve data from the cluster using object storage techniques. The result is a storage solution that is abstracted from the hardware.

Ceph supports both native and traditional client access. The native clients are aware of the storage topology and communicate directly with the storage daemons, resulting in horizontally scaling performance. Non-native protocols, such as ISCSI, S3, and NFS require the use of gateways. While these gateways may be thought of as a limiting factor, the ISCSI and S3 gateways can scale horizontally using load balancing techniques.

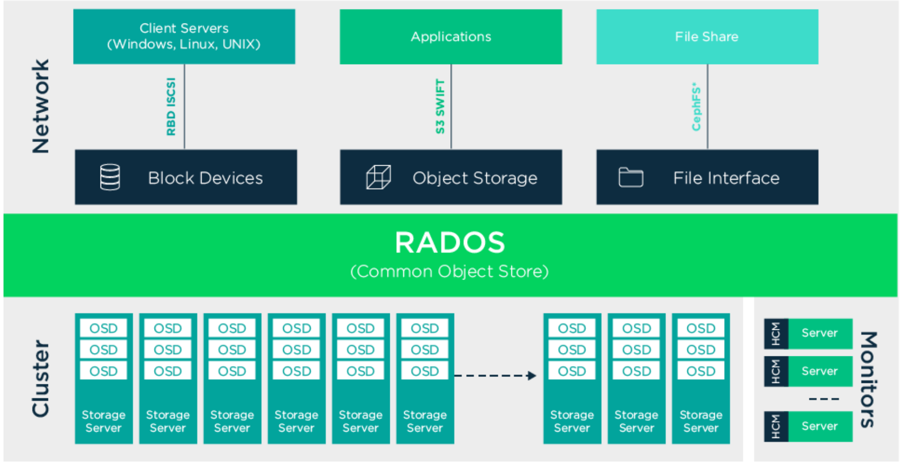


Figure 1. Ceph architecture diagram

In addition to the required network switch ports, the minimum SUSE Enterprise Storage cluster comprises of a minimum of one administration server (physical or virtual), four object storage device nodes (OSDs), and three monitor nodes (MONs). Specific to this implementation:

- One TwinPro node is deployed as the administrative host server. The administration host is the salt-master and hosts openATTIC, the central management system which supports the cluster.

- Three additional TwinPro nodes are deployed as (MONs). Monitor nodes maintain information about the cluster health state, a map of the other monitor nodes and a CRUSH map. They also keep a history of changes performed to the cluster.

- Additional 1U or TwinPro nodes may be deployed as iSCSI gateway nodes. iSCSI is a storage area network (SAN) protocol that allows clients (called initiators) to send SCSI command to SCSI storage devices (targets) on remote servers. This protocol is utilized for block-based connectivity to environments such as Microsoft Windows, VMware, and traditional UNIX. These systems may be scaled horizontally through client usage of multi-path technology.

- The RADOS gateway may also be deployed on TwinPro or 1U nodes. The RADOS gateway provides S3 and Swift based access methods to the cluster. These nodes are generally situated behind a load balancer infrastructure to provide redundancy and scalability. It is important to note that the load generated by the RADOS gateway can consume a significant amount of compute and memory resources making the minimum recommended configuration contain 6-8 CPU cores and 32GB of RAM.

- This configuration uses Supermicro SYS-6029U-E1CR4 systems as storage nodes. The storage nodes contain individual storage devices that are each assigned an Object Storage Daemon (OSD). The OSD daemon assigned to the OSD stores data and manages the data replication and rebalancing processes. OSD daemons also communicate with the MONs and provide them with the state of the other OSD daemons.

  - One particular focus of the OSD node design was to accelerate Write Ahead Log (WAL) and RocksDB performance by placing them on an Intel DC P4600 2TB NVMe PCIe Solid State Drive. This provides a dedicated high throughput, low latency storage location for these critical services. Using the NVMe helps reduce long tail latencies and ensures more consistent performance for the solution.

  - An additional Intel DC P4600 was added to the system for use by Intel® CAS. The intent of the Intel® CAS is to inject a tunable caching layer between the OSD and the physical device. By tuning the size of cached I/Os and the caching behavior itself, it is possible to accelerate very specific I/O sizes.

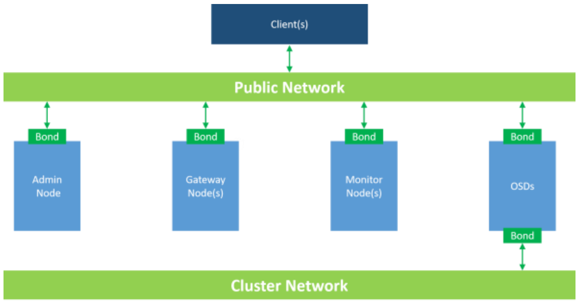


Figure 2. Sample networking diagram for Ceph cluster

Networking architecture

A software-defined solution is as reliable as its slowest and least redundant component. This makes it important to design and implement a robust, high performance storage network infrastructure. From a network perspective for Ceph, this translates into:

Separation of cluster (backend) and client-facing network traffic and isolate Ceph OSD daemon replication activities from Ceph client to storage cluster access.

Redundancy and capacity in the form of bonded network interfaces connected to switches.

Figure 2 shows the logical layout of the traditional Ceph cluster implementation.

In this particular configuration, two VLANs were utilized to segment the traffic. The default VLAN of 1 was untagged and association with the 192.168.101.0/24 network. The cluster network of 192.168.100.0/24 was tagged to VLAN 100 and only configured on OSD nodes. Separate GbE interfaces provided admin network interfaces.

Network/IP address scheme

Specific to this implementation, the following naming and addressing scheme were utilized.

Function	**Hostname**	**Public Network**	**Cluster Network**	**Admin Network**
Admin (Host)	salt.supermicro.lab	192.168.101.90	N/A	192.168.124.90
Monitor	mon1.supermicro.lab	192.168.101.101	N/A	192.168.124.101
Monitor	mon2.supermicro.lab	192.168.101.102	N/A	192.168.124.102
Monitor	mon3.supermicro.lab	192.168.101.103	N/A	192.168.124.103
Object/ISCSI Gateway	gw1.supermicro.lab	192.168.101.104	N/A	192.168.124.104
Object/ISCSI Gateway	gw2.supermicro.lab	192.168.101.105	N/A	192.168.124.105
OSD Node	osd1.supermicro.lab	192.168.101.111	192.168.100.111	192.168.124.111
OSD Node	osd2.supermicro.lab	192.168.101.112	192.168.100.112	192.168.124.112
OSD Node	osd3.supermicro.lab	192.168.101.113	192.168.100.113	192.168.124.113
OSD Node	osd4.supermicro.lab	192.168.101.114	192.168.100.114	192.168.124.114

Component model

The preceding sections provided significant details on the both the overall Supermicro hardware as well as an introduction to the Ceph software architecture. In this section, the focus is on the SUSE components: SUSE Linux Enterprise Server (SLES), SUSE Enterprise Storage (SES), and the Subscription Management Tool (SMT).

Component overview (SUSE)

SUSE Linux Enterprise Server – A world class secure, open source server operating system, equally adept at powering physical, virtual, or cloud-based mission-critical workloads. Service Pack 3 further raises the bar in helping organizations to accelerate innovation, enhance system reliability, meet tough security requirements and adapt to new technologies.

Subscription Management Tool for SLES12 SP3 – allows enterprise customers to optimize the management of SUSE Linux Enterprise (and extensions such as SUSE Enterprise Storage) software updates and subscription entitlements. It establishes a proxy system for SUSE Customer Center with repository and registration targets.

SUSE Enterprise Storage – Provided as an extension on top of SUSE Linux Enterprise Server, this intelligent software-defined storage solution, powered by Ceph technology, with enterprise engineering and support from SUSE enables customers to transform enterprise infrastructure to reduce costs while providing unlimited scalability. The most recent release brings a new underlying storage technology, Bluestore, to the product. Bluestore significantly reduces long tail latencies and significantly improves performance of some use cases. SUSE Enterprise Storage 5 also brings the distributed file system, CephFS to production with multiple meta-data server support, allowing for broad usage of this highly-performant, scale-out technology across many use cases.

Intel® CAS

Intel® Cache Acceleration Software (Intel® CAS) increases storage performance via intelligent caching and is optimized for use with Intel SSDs.

Intel® CAS is a drop-in solution to accelerate applications; it requires no modification to the existing application or backend storage media. An Intel® SSD with Intel® CAS enables the software to utilize the SSD to cache the hottest data from your data node primary storage. It is a cost-effective solution that quickly and easily provides a high-impact boost to the read and write performance of your application and its data workloads.

Intel® CAS accelerates Linux applications by caching active (hot) data to a local block storage device inside servers. Intel® CAS implements caching at the server level, using local high-performance flash media as the cache drive media within the application server, thus reducing storage latency.

Intel® CAS installs into the Linux operating system as a kernel module. The nature of the integration provides a cache solution that is transparent to users, applications, and your existing storage infrastructure. No storage migration or application changes are required. Since CAS installs as a kernel driver, it is important that the module be appropriately signed so as to not taint the kernel and void supportability. It was with this in mind that Intel is making CAS available through the SUSE Solid Driver Program. This program helps ensure supportability by signing the module and having joint support agreements in place, should an issue arise.

As shown in **Figure 1**, initial read data is retrieved from backend storage and copied to the Intel® CAS cache. A second read promotes data to system memory. Subsequent reads are returned at high-performance RAM or flash speed. In Write-through mode, all data is written synchronously to both the backend storage and the cache. In Write-back mode, all data is written to the cache then eventually flushed to the backend storage. When the cache is full, newly identified active data evicts stale data from the cache, utilizing the Intel® CAS proprietary eviction algorithm.

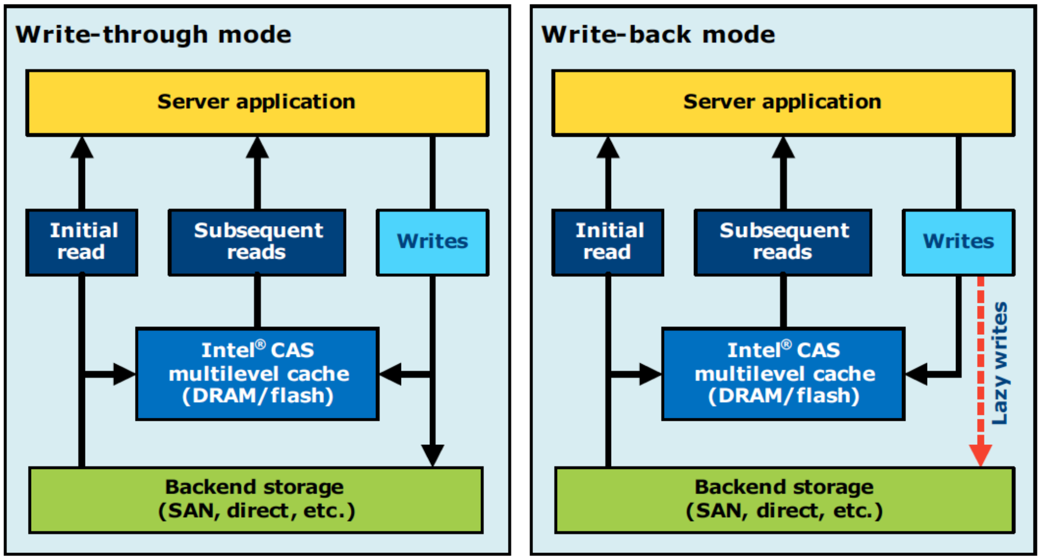

Write-through mode
Server application
Initial read
Subsequent reads
Writes
Intel® CAS multilevel cache (DRAM/flash)
Backend storage (SAN, direct, etc.)
Write-back mode
Server application
Initial read
Subsequent reads
Writes
Lazy writes
Intel® CAS multilevel cache (DRAM/flash)
Backend storage (SAN, direct, etc.)

Figure 3: Block Diagram (write-through and write-back modes)

Deployment

This deployment section should be seen as a supplement online documentation - specifically, the SUSE Enterprise Storage 5 Deployment Guide, the SUSE Linux Enterprise Server Administration Guide and the Intel® CAS documentation. It is assumed that a Subscription Management Tool server exists within the environment. If not, please follow the information in Subscription Management Tool (SMT) for SLES 12 SP3 to make one available. The emphasis is on specific design and configuration choices.

Network Deployment overview/outline

The following considerations for the network configuration should be attended to:

- Ensure that all network switches are updated with consistent firmware versions.
- Configure 802.3ad for system port bonding and IRF between the switches, plus enable jumbo frames.
- Specific configuration for this deployment can be found in Appendix D: Network Switch Configuration
- Network IP addressing and IP ranges need proper planning. In optimal environments, a single storage subnet should be used for all SUSE Enterprise Storage nodes on the primary network, with a separate, single subnet for the cluster network. Depending on the size of the installation, ranges larger than /24 may be required. When planning the network, current as well as future growth should be taken into consideration.
- Setup DNS A records for all nodes. Decide on subnets and VLANs and configure the switch ports accordingly.
- Ensure that you have access to a valid, reliable NTP service, as this is a critical requirement for all nodes. If not, it is recommended to use the admin node.

HW Deployment configuration (suggested)

The following considerations for the hardware platforms should be attended to:

- Ensure Boot Mode is set to ‘UEFI’ for all the physical nodes that comprise the SUSE Enterprise Storage Cluster.
- Verify BIOS/UEFI level on the physical servers correspond to those on the SUSE YES certification for the Supermicro platforms.
- Configure a mirrored pair of drives for the operating system
- Configure all data and journal devices as individual RAID-0

Operating System Deployment and Configuration

When deploying the operating system, be sure to utilize only the correct device. This is the RAID-1 created during hardware configuration.

- Properly configure the network devices during installation. This is illustrated in Appendix E: OS Networking Configuration.
- Register the system against the SMT server.
- When prompted for extensions, select SUSE Enterprise Storage
- On the Suggested Partitioning selection, select Edit Proposal Settings and uncheck Propose Separate Home Partition
- After installation is complete, run **`zypper up`** to ensure all current updates are applied.

SW Deployment configuration (DeepSea and Salt)

Salt along with DeepSea is a stack of components that help deploy and manage server infrastructure. It is very scalable, fast, and relatively easy to get running.

There are three key Salt imperatives that need to be followed and are described in detail in section 4 (Deploying with DeepSea and Salt):

1. The Salt master is the host that controls the entire cluster deployment. Ceph itself should NOT be running on the master as all resources should be dedicated to Salt master services. In our scenario, we used the Admin host as the Salt master.
2. Salt minions are nodes controlled by Salt master. OSD, monitor, and gateway nodes are all Salt minions in this installation.

   Salt minions need to correctly resolve the Salt master's host name over the network. This can be achieved through configuring unique host names per interface (osd1-cluster.supermicro.lab and osd1-public.supermicro.lab) in DNS and/or local `/etc/hosts` files.
3. DeepSea consists of series of Salt files to automate the deployment and management of a Ceph cluster. It consolidates the administrator's decision making in a single location around cluster assignment, role assignment and profile assignment. DeepSea collects each set of tasks into a goal or stage.

The following steps, performed in order will be used for this reference implementation:

- Install the salt-master packages on the admin node:
  - **`zypper in salt-master`**
- Start the *salt-master* service and enable:
  - **`systemctl start salt-master.service`**
  - **`systemctl enable salt-master.service`**
- Install the *salt-minion* on all cluster nodes (including the Admin):
  - **`zypper in salt-minion`**
- Configure all minions to connect to the Salt master: Modify the entry for *master* in the `/etc/salt/minion`
  - In this case: master: sesadmin.domain.com
- Start the *salt-minion* service and enable:
  - **`systemctl start salt-minion.service`**

  -
    ```
    systemctl enable salt-minion.service
    ```

- Clear all non-OS drives on the OSD nodes, reset the labels, and reboot the nodes:

  -
    ```
    dd if=/dev/zero of=/dev/sda bs=1M count=1024 oflag=direct
    ```

  -
    ```
    sgdisk -Z --clear -g /dev/sda
    ```

  -
    ```
    reboot
    ```
- List and accept all salt keys on the Salt master: *salt-key --accept-all* and verify their acceptance

  -
    ```
    salt-key --list-all
    ```

  -
    ```
    salt-key --accept-all
    ```
- Install DeepSea on the Salt master which is the Admin node:

  -
    ```
    zypper in deepsea
    ```
- At this point, you can deploy and configure the cluster:
  - Prepare the cluster: `deepsea stage run ceph.stage.prep`
  - Run the discover stage to collect data from all minions and create configuration fragments:

    -
      ```
      deepsea stage run ceph.stage.discovery
      ```
  - A default proposal is generated by the previous stage, however, we desire to use a custom proposal generated by:

    -
      ```
      salt-run proposal.populate name=nvme ratio=9 wal=1700-2000 db=1700-2000 target='osd*' db-size=55g wal-size=2g
      ```
  - A `/srv/pillar/ceph/proposals/policy.cfg` file needs to be created to instruct Salt on the location and configuration files to use for the different components that make up the Ceph cluster (Salt master, admin, monitor, and OSDs).
    - See Appendix C for the `policy.cfg` file used in the installation.
  - Next, proceed with the configuration stage to parse the `policy.cfg` file and merge the included files into the final form

    -
      ```
      deepsea stage run ceph.stage.configure
      ```

- The last two steps manage the actual deployment. Deploy monitors and ODS daemons first:
    - **`deepsea stage run ceph.stage.deploy`** (Note: The command can take some time to complete, depending on the size of the cluster).
    - Check for successful completion via: **`ceph –s`**
    - Finally, deploy the services(gateways [iSCSI, RADOS], and openATTIC to name a few): **`deepsea stage run ceph.stage.services`**

Post-deployment quick test

The steps below can be used (regardless of the deployment method) to validate the overall cluster health:

```
ceph status

ceph osd pool create test 1024

rados bench –p test 300 write --no-cleanup

rados bench –p test 300 seq
```

Once the tests are complete, you can remove the test pool via:

```
ceph tell mon.* injectargs --mon-allow-pool-delete=true

ceph osd pool delete test test --yes-i-really-really-mean-it

ceph tell mon.* injectargs --mon-allow-pool-delete=false
```

Intel® CAS Implementation

For purposes of this implementation, Intel® CAS was added from the SUSE Solid Driver repository. On each OSD node, the software needs to be installed as per the instructions noted in the SUSE section of the installation guide found here: https://drivers.suse.com/intel/Intel-CAS/sle-12-sp3-x86_64/1.0/install-readme.html

Intel® CAS features which should be considered when optimizing for the storage environment include, but are not limited to:

Multiple Cache Instances: for dense storage nodes, e.g., greater than six core devices, multiple cache instances provide greater robustness for handling intensive IO and preventing cache drive saturation.

Cache Management and Statistics: Intel® CAS provides a user-based admin utility (**casadm**) that allows for monitoring and configuration of the caching environment. Statistics such as cache occupancy, hits and misses (reads & writes), amount of dirty data, etc. are able to be viewed during workload runtime.

Administrator's Guide: For further details on setup, configuration, and operations, Intel provides a comprehensive administrator's guide for instructional guidance.

The deployment process with SES is the same as that called out for Ceph in the Intel® CAS Admin Guide in chapter 14 with the variation that we use a single cache for all devices connected to it. For completeness, the procedure utilized is reflected here:

After installation is complete, an included configuration file (/etc/intelcas/intelcas.conf) must be edited to specify the caching device, the caching type, and the devices (cores which represent the backend storage) to be cached. A discussion of the configuration file may be found in section 4.1 of the Admin Guide included in the SUSE Solid Driver repository for Intel® CAS https://drivers.suse.com/intel/Intel-CAS/doc. In the example below, there is a single NVMe device supporting 4 OSD devices in write-through mode.

```
[caches]

1        /dev/disk/by-id/nvme-INTEL_SSDPE2MD016T4_PHFT5523000N1P6JGN-part9
WT



 [cores]

1  /dev/disk/by-id/scsi-36001ec90ee0b140022e4d8a80842f4e3

1   /dev/disk/by-id/scsi-36001ec90ee0b140022e4d8ba095c73d7

1   /dev/disk/by-id/scsi-36001ec90ee0b140022e4d8c90a3add3b

1   /dev/disk/by-id/scsi-36001ec90ee0b140022e4d8d50af3ec6c
```

The next step is to set the OSDs where they won't be detected as "out" during Intel® CAS enablement.

```
ceph osd set noout

ceph osd set norebalance
```

Next it is necessary to stop the OSD services and unmount the Ceph OSD directories:

```
systemctl stop ceph-osd@*
```

Alternatively:

```
systemctl stop ceph-osd.target

umount /var/lib/ceph/osd/ceph-*
```

Once Ceph has been stopped and no longer has attachment to the backend storage drives, and the /etc/intelcas/intelcas.conf file has been configured, CAS can be initialized and started via:

```
intelcas init
```

This step may take a few minutes to process while Intel® CAS reads the configuration file, starts the cache, and pairs the core devices to the caching device. After initialization, Ceph services are auto-restarted, OSD’s mounted, and the cluster returned to its prior status. Once complete, cache status can be checked with:

```
casadm -L
```

The output should show attached and active caching device(s) and cores as is seen below

```
type   id  disk          status    write policy  device
cache  1   /dev/nvme0n1  Running   wa            -
├core  1   /dev/sda      Active    -             /dev/intelcas1-1
├core  2   /dev/sdb      Active    -             /dev/intelcas1-2
├core  3   /dev/sdc      Active    -             /dev/intelcas1-3
```

Repeat the process for each OSD node. Intel® CAS configuration and initialization processes may be scripted for ease of installation in large environments.

When complete, clear the options that prevent OSDs being out or rebalanced:

```
ceph osd unset noout

ceph osd unset norebalance
```

Deployment Considerations

Some final considerations before deploying your own version of a SUSE Enterprise Storage cluster, based on Ceph. As previously stated, please refer to the Deployment Guide.

With the default replication setting of 3, remember that the client-facing network will have about half or less of the traffic of the backend network. This is especially true when component failures occur or rebalancing happens on the OSD nodes. For this reason, it is important not to under provision this critical cluster and service resource.

It is important to maintain the minimum number of monitor nodes at three. As the cluster increases in size, it is best to increment in pairs, keeping the total number of MONs as an odd number. However, only very large or very distributed clusters would likely need beyond the three monitor nodes cited in this reference implementation. For performance reasons, it is recommended to use distinct nodes for the MON roles, so that the OSD nodes can be scaled as capacity requirements dictate.

As described in this implementation guide as well as the SUSE Enterprise Storage documentation, a minimum of four OSD nodes is recommended, with the default replication setting of 3. This will ensure cluster operation, even with the loss of a complete OSD node. Generally speaking, performance of the overall cluster increases as more properly configured OSD nodes are added.

It is important to note that caching will not resolve a poorly sized cluster or a cluster with insufficient networking resources. Write-back caching is designed to deal with bursts of write activity, if the backing storage lacks the I/O capacity to deal with the performance requirements for nominal operation, the cache will not provide any measurable benefit. Write-around and write-through modes are designed to improve repetitive reads, meaning that if the working set is substantially larger than the cache, the percentage of cache hits (and thus performance improvement from the cache) will be reduced or nullified entirely.

Appendix A: Bill of Materials

Component / System

Role	**Qty**	**Component**	**Notes**
Admin/MON/Gateway servers	1	SYS-6019U-TN4RT	Each node consists of: 2x Intel(R) Xeon(R) Gold 5120 CPU @ 2.20GHz 128GB RAM 2x 2TB SATA in RAID-1 1 Intel XL710 w/40GbE QSFP+
Admin/MON/Gateway servers	2	SYS-6029TP-HC0R	Each node consists of: 2 2TB SATA using RAID-1 128GB RAM 2x Intel(R) Xeon(R) 2630V4 CPU @ 2.20GHz Intel Ethernet Controller XL710 for 40GbE QSFP+
OSD Hosts	4	SYS-6029U-E1CR4	Each node consists of: 2x Intel(R) Xeon(R) CPU E5-2630L v4 @ 1.80GHz 128GB RAM 2x 2TB SATA drives in RAID-1 10x SATA drives 1 Intel XL710 w/40GbE QSFP+
Software	1	SUSE Enterprise Storage Subscription Base configuration	Allows for 4 storage nodes and 6 infrastructure nodes
Software	1	Intel® Cache Acceleration Software	Caching software

Appendix B: OSD Drive and Journal Proposal Changes

The proposal generated by salt-run proposal.populate name=nvme ratio=9 wal=1700-2000 db=1700-2000 target='osd*' db-size=55g wal-size=2g and selected for use is named: `profile-nvme`. The contents of the proposal are below.

```
ceph:
  storage:
    osds:
      /dev/disk/by-id/ata-HGST_HUS724040ALE640_PK2331PAG66ART:
        db: /dev/disk/by-id/nvme-INTEL_SSDPEDKE020T7_PHLE725100252P0IGN
        db_size: 50g
        format: bluestore
        wal: /dev/disk/by-id/nvme-INTEL_SSDPEDKE020T7_PHLE725100252P0IGN
        wal_size: 2g
      /dev/disk/by-id/ata-HGST_HUS724040ALE640_PK2331PAJ93B6T:
        db: /dev/disk/by-id/nvme-INTEL_SSDPEDKE020T7_PHLE725100252P0IGN
        db_size: 50g
        format: bluestore
        wal: /dev/disk/by-id/nvme-INTEL_SSDPEDKE020T7_PHLE725100252P0IGN
        wal_size: 2g
      /dev/disk/by-id/ata-HGST_HUS724040ALE640_PK2331PAJ94G0T:
        db: /dev/disk/by-id/nvme-INTEL_SSDPEDKE020T7_PHLE725100252P0IGN
        db_size: 50g
        format: bluestore
        wal: /dev/disk/by-id/nvme-INTEL_SSDPEDKE020T7_PHLE725100252P0IGN
        wal_size: 2g
      /dev/disk/by-id/ata-ST4000NM115-1YZ107_ZC11VNZ1:
        db: /dev/disk/by-id/nvme-INTEL_SSDPEDKE020T7_PHLE725100252P0IGN
        db_size: 50g
        format: bluestore
        wal: /dev/disk/by-id/nvme-INTEL_SSDPEDKE020T7_PHLE725100252P0IGN
        wal_size: 2g
      /dev/disk/by-id/ata-ST4000NM115-1YZ107_ZC11VP04:
        db: /dev/disk/by-id/nvme-INTEL_SSDPEDKE020T7_PHLE725100252P0IGN
        db_size: 50g
        format: bluestore
        wal: /dev/disk/by-id/nvme-INTEL_SSDPEDKE020T7_PHLE725100252P0IGN
        wal_size: 2g
      /dev/disk/by-id/ata-WDC_WD4000FYYZ-01UL1B0_WD-WCC130368354:
        db: /dev/disk/by-id/nvme-INTEL_SSDPEDKE020T7_PHLE725100252P0IGN
        db_size: 50g
        format: bluestore
        wal: /dev/disk/by-id/nvme-INTEL_SSDPEDKE020T7_PHLE725100252P0IGN
        wal_size: 2g
      /dev/disk/by-id/scsi-35000cca22bddefac:
        db: /dev/disk/by-id/nvme-INTEL_SSDPEDKE020T7_PHLE725100252P0IGN
        db_size: 50g
        format: bluestore
        wal: /dev/disk/by-id/nvme-INTEL_SSDPEDKE020T7_PHLE725100252P0IGN
        wal_size: 2g
      /dev/disk/by-id/scsi-35000cca22be08431:
        db: /dev/disk/by-id/nvme-INTEL_SSDPEDKE020T7_PHLE725100252P0IGN
        db_size: 50g
        format: bluestore
        wal: /dev/disk/by-id/nvme-INTEL_SSDPEDKE020T7_PHLE725100252P0IGN
        wal_size: 2g
```

NOTE: There is ONE space between the colon separating the OSD and journal entries. Accurate spacing is important with *Salt*.

Appendix C: Policy.cfg

```
## Cluster Assignment
cluster-ceph/cluster/*.sls

## Roles
# ADMIN
role-master/cluster/salt*.sls
role-admin/cluster/salt*.sls

# MON
role-mon/cluster/mon*.sls

# MGR (mgrs are usually colocated with mons)
role-mgr/cluster/mon*.sls

# MDS
#role-mds/cluster/mds*.sls

# IGW
role-igw/cluster/gw*.sls

# RGW
role-rgw/cluster/gw*.sls

# NFS
#role-ganesha/cluster/ganesha*.sls

# openATTIC
role-openattic/cluster/salt*.sls

# COMMON
config/stack/default/global.yml
config/stack/default/ceph/cluster.yml

## Profiles
profile-nvme/cluster/*.sls
profile-nvme/stack/default/ceph/minions/*.yml
```

Appendix D: Network Switch Configuration

First, properly cable and configure each node on the switches. Ensuring proper switch configuration at the outset will prevent networking issues later. The key configuration items include ensuring the switches are properly stacked with an MLAG, creating LACP bonds, VLANS, and enabling jumbo frames. Each aggregation group needs a unique number, planned ahead of time. It is also recommended that you disable the spanning tree on the ports utilized for storage.

Configure the MLAG for stacking the switches as found here:

https://docs.cumulusnetworks.com/display/DOCS/Multi-Chassis+Link+Aggregation+-+MLAG

If enabling jumbo frame packets, follow the directions under MTU found here:

https://docs.cumulusnetworks.com/display/DOCS/Layer+1+and+Switch+Port+Attributes

The next step is to create the LACP bonds. Information on this step is found here:

https://docs.cumulusnetworks.com/display/DOCS/Bonding+-+Link+Aggregation

There are special requirements for using LACP over the stacked switches as described in the MLAG section under LACP and Dual-Connectedness

The final step is to add the interfaces to a bridge and enable VLANs. This can be found here:

https://docs.cumulusnetworks.com/display/DOCS/VLAN-aware+Bridge+Mode+for+Large-scale+Layer+2+Environments

SPECIAL NOTE:

In some situations where the switch is operating at less than 100Gb, it may be necessary to force the ports on the switch to the appropriate speed. In the testing performed for this document, 40GbE adapters were used and thus the following command was issued for each interface in use:

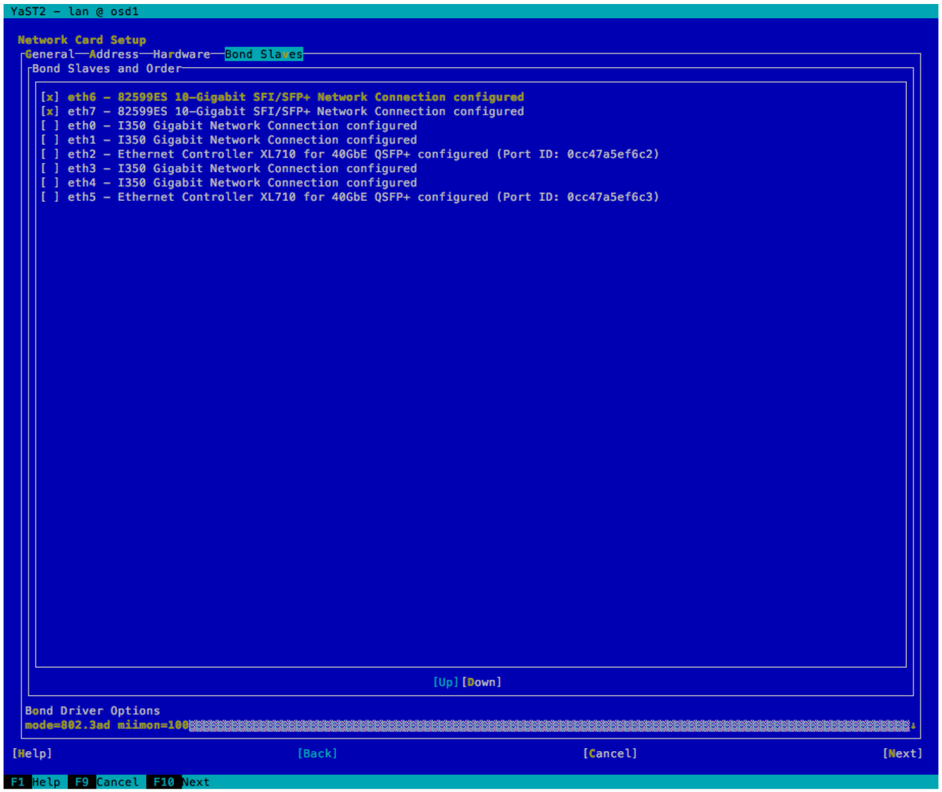
```
link-speed 40000
```

Appendix E: OS Networking Configuration

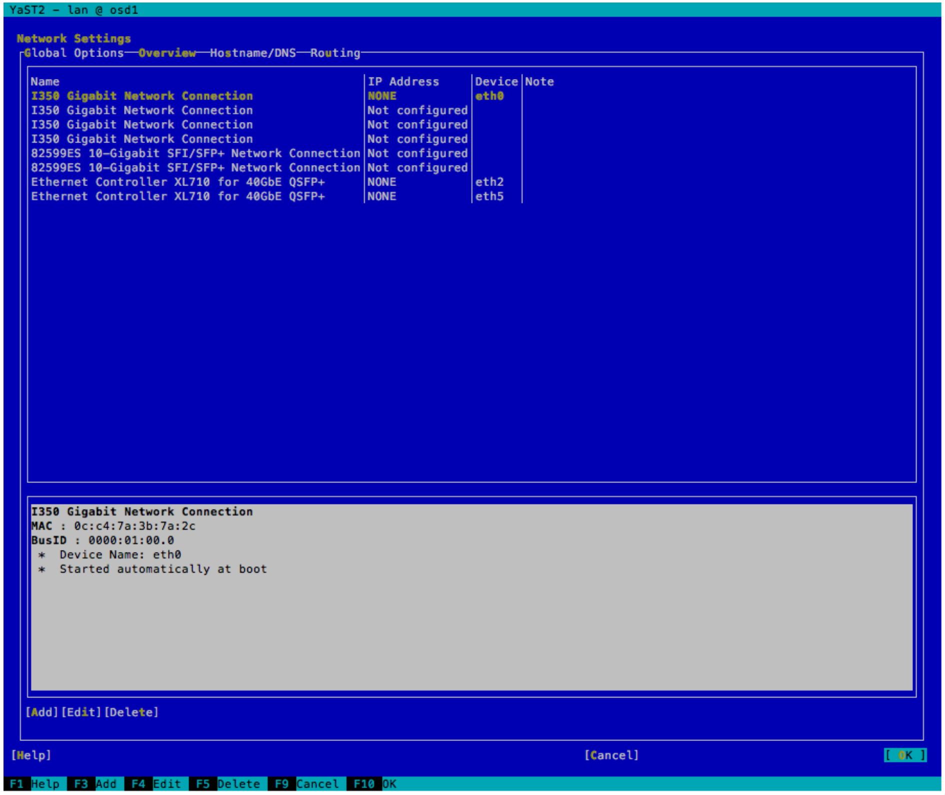
Perform the network configuration during installation.

`Set the Intel XL710 to No Link`

Add an interface of type bond, set it to the proper IP address for the untagged VLAN, and proceed to the Bond slaves page where the Intel XL710 interfaces should be selected and the mode set to 802.3ad



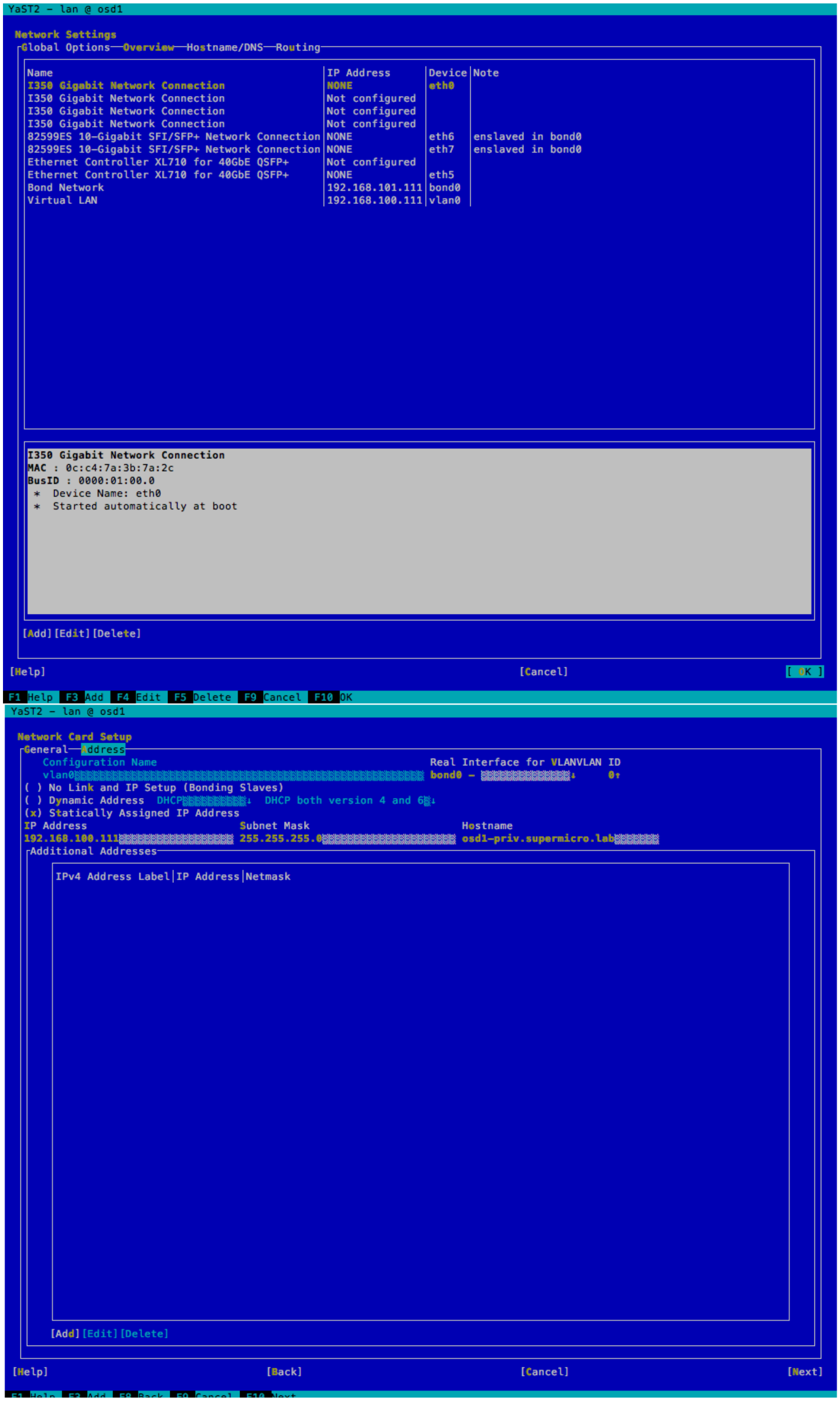
```
YaST2 - lan @ osd1

Network Card Setup
┌General──Address──Hardware──Bond Slaves─────────────────────────
│┌Bond Slaves and Order──────────────────────────────────────────

  [x] eth6 - 82599ES 10-Gigabit SFI/SFP+ Network Connection configured
  [x] eth7 - 82599ES 10-Gigabit SFI/SFP+ Network Connection configured
  [ ] eth0 - I350 Gigabit Network Connection configured
  [ ] eth1 - I350 Gigabit Network Connection configured
  [ ] eth2 - Ethernet Controller XL710 for 40GbE QSFP+ configured (Port ID: 0cc47a5ef6c2)
  [ ] eth3 - I350 Gigabit Network Connection configured
  [ ] eth4 - I350 Gigabit Network Connection configured
  [ ] eth5 - Ethernet Controller XL710 for 40GbE QSFP+ configured (Port ID: 0cc47a5ef6c3)

                                  [Up] [Down]

 Bond Driver Options
 mode=802.3ad miimon=100

[Help]                     [Back]                     [Cancel]                     [Next]

 F1 Help  F9 Cancel  F10 Next
```



```
YaST2 - lan @ osd1

Network Settings
┌Global Options──Overview──Hostname/DNS──Routing──────────────────

 Name                                        │IP Address    │Device│Note
 I350 Gigabit Network Connection             │NONE          │eth0  │
 I350 Gigabit Network Connection             │Not configured│      │
 I350 Gigabit Network Connection             │Not configured│      │
 I350 Gigabit Network Connection             │Not configured│      │
 82599ES 10-Gigabit SFI/SFP+ Network Connection│Not configured│      │
 82599ES 10-Gigabit SFI/SFP+ Network Connection│Not configured│      │
 Ethernet Controller XL710 for 40GbE QSFP+   │NONE          │eth2  │
 Ethernet Controller XL710 for 40GbE QSFP+   │NONE          │eth5  │

 I350 Gigabit Network Connection
 MAC : 0c:c4:7a:3b:7a:2c
 BusID : 0000:01:00.0
  *  Device Name: eth0
  *  Started automatically at boot

 [Add] [Edit] [Delete]

[Help]                                        [Cancel]                          [ OK ]

 F1 Help  F3 Add  F4 Edit  F5 Delete  F9 Cancel  F10 OK
```

Add the VLAN interfaces, making sure to select the correct VLAN ID and setting the IP and host information.

This figure represents the proper network configuration for osdnode1 as configured in this paper.



```
YaST2 - lan @ osd1

Network Settings
┌Global Options──Overview──Hostname/DNS──Routing
 Name                                        │IP Address     │Device│Note
 I350 Gigabit Network Connection             │NONE           │eth0  │
 I350 Gigabit Network Connection             │Not configured │      │
 I350 Gigabit Network Connection             │Not configured │      │
 I350 Gigabit Network Connection             │Not configured │      │
 82599ES 10-Gigabit SFI/SFP+ Network Connection│NONE         │eth6  │enslaved in bond0
 82599ES 10-Gigabit SFI/SFP+ Network Connection│NONE         │eth7  │enslaved in bond0
 Ethernet Controller XL710 for 40GbE QSFP+   │Not configured │      │
 Ethernet Controller XL710 for 40GbE QSFP+   │NONE           │eth5  │
 Bond Network                                │192.168.101.111│bond0 │
 Virtual LAN                                 │192.168.100.111│vlan0 │

 I350 Gigabit Network Connection
 MAC : 0c:c4:7a:3b:7a:2c
 BusID : 0000:01:00.0
  *  Device Name: eth0
  *  Started automatically at boot

[Add][Edit][Delete]

[Help]                                              [Cancel]                   [ OK ]
F1 Help  F3 Add  F4 Edit  F5 Delete  F9 Cancel  F10 OK
YaST2 - lan @ osd1

Network Card Setup
┌General──Address
    Configuration Name                                       Real Interface for VLANVLAN ID
    vlan0                                                    bond0 -          ↓      0↑
( ) No Link and IP Setup (Bonding Slaves)
( ) Dynamic Address  DHCP       ↓  DHCP both version 4 and 6 ↓
(x) Statically Assigned IP Address
IP Address                     Subnet Mask                  Hostname
192.168.100.111                255.255.255.0                osd1-priv.supermicro.lab
┌Additional Addresses
      IPv4 Address Label│IP Address│Netmask

      [Add][Edit][Delete]

[Help]                    [Back]                     [Cancel]                  [Next]
F1 Help  F3 Add  F8 Back  F9 Cancel  F10 Next
```

Appendix F: Performance Data

Comprehensive performance baselines are run as part of a reference build activity. This activity yields a vast amount of information that may be used to approximate the size and performance of the solution. The only tuning applied is documented in the implementation portion of this document.

The tests are comprised of a number of Flexible I/O (fio) job files run against multiple worker nodes. The job files and testing scripts may be found for review at: https://github.com/dmbyte/benchmaster. This is a personal repository and no warranties are made in regard to the fitness and safety of the scripts found there.

The testing methodology involves two different types of long running tests. The types and duration of the tests have very specific purposes. There are both i/o simulation jobs and single metric jobs.

The length of the test run, in combination with the ramp-up time specified in the job file, is intended to allow the system to overrun caches. This is a worst-case scenario for a system and would indicate that it is running at or near capacity. Given that few applications can tolerate significant amounts of long tail latencies, the job files have specific latency targets assigned. These targets are intended to be in-line with expectations for the type of I/O operation being performed and set realistic expectations for the application environment.

The latency target, when combined with the latency window and latency window percentage set the minimum number of I/Os that must be within the latency target during the latency window time period. For most of the tests, the latency target is 20ms of less. The latency window is five seconds and the latency target is 99.99999%. The way that fio uses this is to ramp up the queue depth at each new latency window boundary until more than .00001% of all I/O's during a five second window are higher than 20ms. At that point, fio backs the queue depth down where the latency target is sustainable.

These settings, along with block size, max queue depth, jobs per node, etc, are all visible in the job files found at the repository link above.

Caching

The performance testing was performed with both write-through and write-back testing. Given that the benchmark tests generate a worst-case scenario for caching due to very large working set size and sustained maximum throughput/iops, benefits were not realized for write-back caching in any cases. In the tested scenarios, the cache is consumed with evictions of dirty data to backend storage to make room for new data. Data contention will occur which causes significant amounts of incoming I/O to be processed directly to the backend store in pass-through mode. Subsequent testing with smaller working sets and/or shorter test runs indicate substantial improvements (up to about 30% on 4k random writes) for cases where the cache is not being constantly over-run thus allowing the cache to de-stage, or flush, to the backing store effectively.

In a real-world deployment, it is expected that the back-end would be appropriately sized to handle the I/O, thus allowing the cache to deal with activity bursts and in general work to improve overall latencies. In the real world deployment, performance can be significantly adjusted based on right-sizing of the cache. Close awareness of caching statistics such as cache occupancy (how full the cache device is) and percentage of dirty data associated with write-back mode are vital to gaining the most advantage from Intel® CAS.

Write-through results proved more interesting with performance gains across the board with random reads on cephFS. The results were most dramatic with erasure coded backing pools where there is significantly more read activity to re-assemble the data during a read vs the replicated pools. Some similar results were observed with RBD but were not deemed conclusive enough for inclusion in this publication with the exception of the VM simulation test where a performance gain of 25% was realized using Intel® CAS.

It is also interesting to note that all recorded gains occurred with write sizes of 64KB and below. This is indicative of overrunning the cache and maxing out the performance of the OSD devices, thus negating the effect of the CAS software.

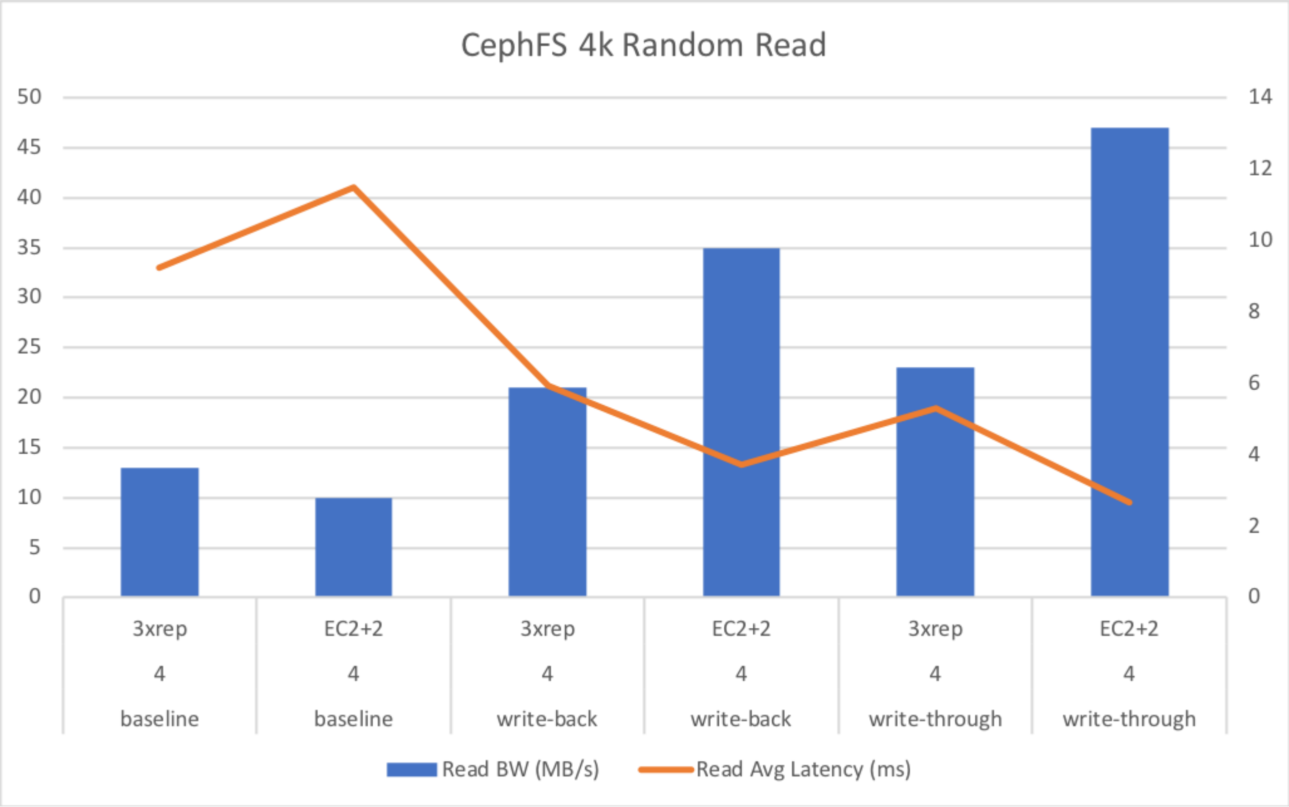
Selected results where Intel® CAS showed performance gains are included in the following pages for comparison with baseline results.

CephFS Random Reads

The random read tests were conducted on both 4K and 64K I/O sizes with latency targets of 10ms and 20ms respectively. The 64k random read shows write-back being of slightly higher performance, which is anomalous to the rest of the data. The hardware availability window did not permit repeating the test to attempt reproduction of the anomaly. The top performance mark for each protection type is highlighted in green.

4k CephFS Random Read

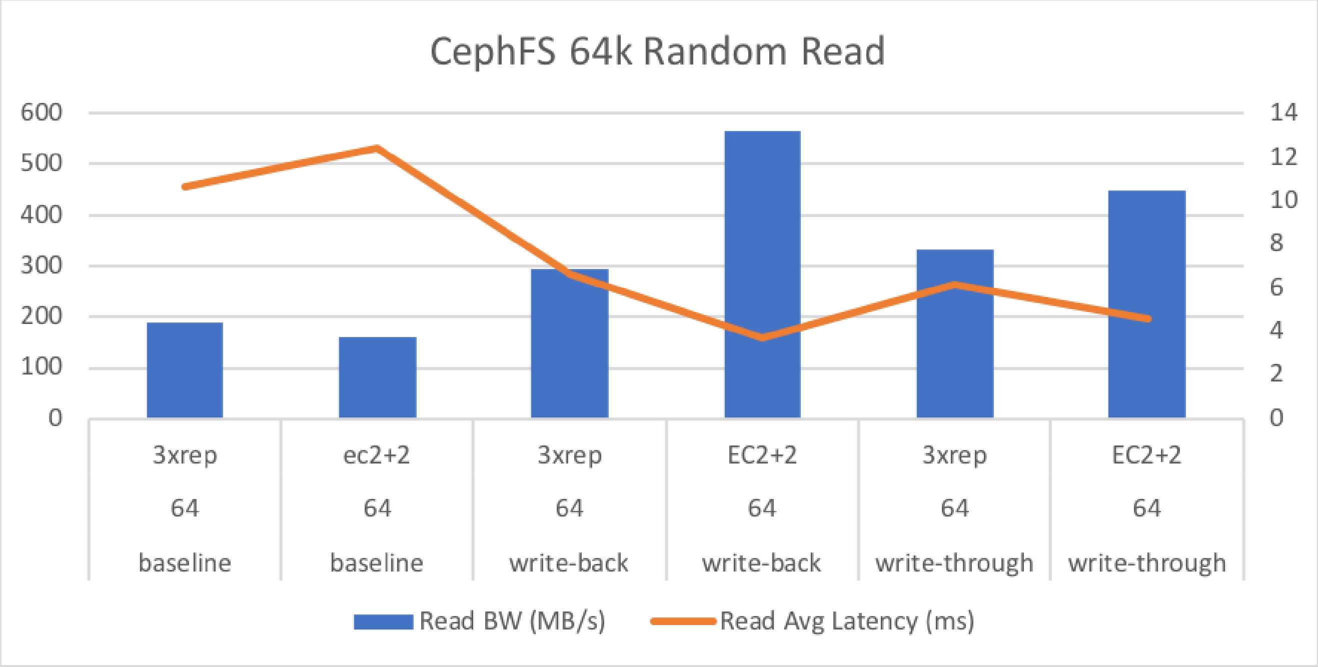
CAS Status	IO Size	Protection	Read BW (MB/s)	Read IOPS	Read Avg Latency (ms)
baseline	4	3xrep	13	3471	9
baseline	4	EC2+2	10	2796	11
write-back	4	3xrep	21	5404	6
write-back	4	EC2+2	35	9042	4
write-through	4	3xrep	23	6065	5
write-through	4	EC2+2	47	12096	3



In the results below, both write-back and write-through indicated a positive improvement for EC pools. During normal cluster operation, it is expected that this would likely not be the case due to writeback caching causing a faster cache ejection rate.

64k CephFS Random Read

CAS Status	IO Size	Protection	Read BW (MB/s)	Read IOPS	Read Avg Latency (ms)
baseline	64	3xrep	189	3037	11
baseline	64	ec2+2	161	2590	12
write-back	64	3xrep	295	4732	7
write-back	64	EC2+2	567	9083	4
write-through	64	3xrep	332	5321	6
write-through	64	EC2+2	449	7193	5



Workload Simulation

In a simulated workload environment for KVM virtualized guests, gains were indicated when using write-through caching. This is expected as the write-through cache lowers read activity to the backing store, allowing writes to complete with less contention.

Test Name	CAS Status	Protection	Write BW (MB/s)	Write IOPS	Write Avg Latency (ms)	Read BW (MB/s)	Read IOPS	Read Avg Latency (ms)
rbd-kvm-krbd	baseline	3xrep	4	1167	17	18	4655	2
rbd-kvm-krbd	write-back	3xrep	4	1053	23	16	4198	2
rbd-kvm-krbd	write-through	3xrep	6	1609	15	25	6427	1

Simulated KVM Workload

Resources:

SUSE Enterprise Storage Technical Overview

https://www.suse.com/docrep/documents/1mdg7eq2kz/suse_enterprise_storage_technical_overview_wp.pdf

SUSE Enterprise Storage v5 - Administration Guide
https://www.suse.com/documentation/suse-enterprise-storage-5/book_storage_admin/data/book_storage_admin.html

SUSE Linux Enterprise Server 12 SP3 - Administration Guide
https://www.suse.com/documentation/sles-12/book_sle_admin/data/book_sle_admin.html

Subscription Management Tool for SLES 12 SP3
https://www.suse.com/documentation/sles-12/book_smt/data/book_smt.html

About Super Micro Computer, Inc.

Supermicro® (NASDAQ: SMCI), the leading innovator in high-performance, high-efficiency server technology is a premier provider of advanced server Building Block Solutions® for Data Center, Cloud Computing, Enterprise IT, Hadoop/Big Data, HPC and Embedded Systems worldwide. Supermicro is committed to protecting the environment through its “We Keep IT Green®” initiative and provides customers with the most energy-efficient, environmentally-friendly solutions available on the market.

Learn more at www.supermicro.com