SUSE Enterprise Storage™ Architectural Overview and Recommendations
Executive Summary

The case for scale-out storage continues to grow in the enterprise. According to IDC, storage will continue to grow by thirty percent every year, making scale-up strategies difficult and expensive to maintain.

Parallel to the continued storage growth, the shift to open source technology continues to accelerate as well. Combine the two, and the result is Ceph, an enterprise grade, scale-out storage solution.

The architecture described illustrates how to deploy SUSE Enterprise Storage™, a commercially supported Ceph distribution, as a block storage repository for systems connecting via iSCSI or native RBD. The design allows for performance characteristics similar to those of a mid-range array.

Some example use cases for a storage environment such as this include:

- Storage for Test/Dev
- Storage for low to moderate I/O virtual machines
- Disk-based backup target
- General archive storage

Architecture Overview

General Information

Ceph is a distributed object store and file system designed to provide excellent performance, reliability and scalability. It delivers object, block and file storage in one unified system. (Source: Ceph documentation)

Ceph leverages a back-end object store called “RADOS.” While this can be directly accessed by applications using librados, it is most common to use a gateway interface. The gateways currently available include S3 and Swift, provided by the RADOS gateway component RADOS Block Device (RBD) and, most recently, iSCSI.

The figure below illustrates a high-level, logical view of the Ceph solution.

As the diagram illustrates, Ceph is very modular. From the physical perspective, each storage device (spinning disk or SSD) is paired with a Linux daemon, making it an independent and movable storage unit. These storage units, or OSDs, are grouped together in multiple OSD nodes. The current state of the storage
system is tracked by the monitor daemons. It is generally recommended that these daemons reside on dedicated servers with one daemon per server. These are represented here by the three green monitor nodes to the right.

Data is administered in a logical construct called a “pool.” Pools have only a little more than a loose association with specific OSDs (device + daemon) and by default, a single OSD will host multiple pools. It is possible to have a tighter linkage if a CRUSH rule is defined to identify particular types of OSDs and that rule is applied to pool creation. In a general case, this allows a pool to be virtually spread across all the OSDs available or to be targeted to a specific set of OSDs, usually based on performance characteristics.

Each pool contains objects. It is these objects that are directly accessible by librados or are the back-end of the various gateways described above. RBDs and iSCSI LUNs are made up of multiple objects spread throughout the pool(s) containing them.

Data Protection
One attribute assigned to a pool is the data protection scheme. Ceph allows for administrative flexibility in implementing data protection. It is possible to configure a pool with everything from no protection all the way up to clearly impractical implementation with hundreds of replicas. It is also possible to make use of erasure coding, although this is suggested primarily for cold storage and archive data. For the purposes of this paper, we will utilize 3x replication unless noted otherwise, meaning there are three copies of data stored in the pools.

Because Ceph makes all storage available for use and expects to find recovery storage available in the event of a failure, it is important to account for this capacity during the initial sizing and architecture of a cluster. With a replica of 3, it is recommended to have at least 4 nodes; this is the minimum supported cluster size. This provides enough space to survive failure of a full node.

It is up to the administrator to monitor and ensure that this “spare” capacity is always available, to provide sufficient capacity for reconstruction. One way to do this would be to leverage Ceph’s quota system and set fixed maximum sizes for the pools, to prevent them from taking up some of the recovery capacity.

Gateway Services
Within the SUSE Enterprise Storage cluster, gateway daemons provide certain services. These services include the RADOS Gateway and iSCSI services. It is important for these services to be made highly available and not become a single point of failure. This is achieved in a different manner for each service.

For RADOS Gateway (rgw) Services, high availability is achieved through utilizing an HTTP load balancer. In general, the rgw services would be deployed on several nodes and then be presented via a cluster of network load balancers. It is also possible to use solutions such as IP virtual server (IPVS), haproxy and other load balancing solutions. Both haproxy and IPVS are provided by SUSE® Linux High Availability Extension.

For iSCSI services, availability is provided by deploying multiple iSCSI target services presenting the same LUNs. This is possible through the use of multi-pathing at the client. With multiple target nodes presenting the same target LUN, it is possible to achieve both increased aggregate performance and improved availability. For purposes of this paper, iSCSI services are handled by each OSD. For clusters that are more performance-sensitive or of larger size (in excess of 16 OSD nodes), it is advised to create standalone iSCSI gateway nodes.

Calamari
SUSE Enterprise Storage includes Romana (formerly calamari-clients) as a lightweight monitoring and basic management tool. This web-based tool can be deployed on a monitor node or on a dedicated admin node and provides a basic view of the cluster health, basic OSD management and very basic pool management.

Sizing
Properly sizing a storage cluster can be challenging. Factors ranging from drive connectivity to CPU speed to network design can and will affect the overall performance of the cluster. It is important to consider as much data as possible during the cluster design.
Key Questions
Ask key questions such as these to determine the storage cluster size:

- What is the throughput expectation for a single thread?
- What is the aggregate throughput expectation?
- What is the latency expectation (minimum, maximum, average)?
- How much performance degradation is acceptable during a failure/rebuild event?
- What is the read/write ratio?
- How big is the working data set?
- What is the access protocol or method?

Knowing the answers to these questions will help in determining which storage technologies are needed for various “pools,” how much bandwidth each node needs for client interaction and internal cluster functions, and the necessity of cache tier and other optimizations that can be accounted for in the design.

General Sizing Rules for Monitors
Monitor nodes lack the heavy resource requirements of an OSD node. In general, a monitor only needs a few relatively fast cores (4 @ 2.3 Ghz) and 4GB of RAM. The disks are recommended to be mirrored SSDs because they receive a significant amount of fsync activity. It is also recommended that the connectivity be in line with the rest of the cluster for the public network.

General Sizing Rules for Gateway Services
Sizing of gateway services can be a bit tricky, especially understanding the amount of work that will be performed by the gateway service and considering the impact to the hosting node. In general, these services are simply providing protocol translation. In the case of a RADOS gateway service (RGW), it may be able to co-locate on an iSCSI node that has extra cores and memory. Be aware that some anecdotal testing where 40Gbe interfaces were being utilized drove 8 cores and utilized slightly over 32GB of RAM for RGW services alone. iSCSI gateway services are better suited to dedicated nodes for a large installation. If in doubt, dedicate a few nodes that can be upgraded with additional RAM or processing horsepower if needed.

General Sizing Rules for OSD
There are general rules of thumb when sizing a cluster. This section addresses the OSD nodes. Failure to follow these guidelines can have a negative impact on the cluster at various points during normal or degraded modes of operation.

Nodes and Drives
There are many applications for the scale-out nature of Ceph storage and a matching number of chassis configurations. With the chassis it is important to balance the number of OSDs in a single chassis with the total number of chassis. The goal is to find the proper cluster size with acceptable impact during rebuild while providing sufficient overhead to account for a chassis-level failure. The recommended size is for a minimum of seven OSD nodes. This ensures that no single node exceeds 15% of the total cluster size.

To build an example, use the cluster nodes below:

A four-node cluster with 3x copies of the data would require the full capacity of a single node to be available for rebuild purposes, equaling twenty-five percent storage overhead. However, the same scenario in a 10-node cluster would impose only a ten percent storage overhead.

It also is important to think in terms of performance impact during rebuild. Using our previous example, in a node failure in the four-node cluster, a failure could result in as much as one-third of the cluster data moving around to rebalance, heavily impacting the performance of all of the nodes. However, in the second example with ten nodes, the impact is spread across the entire nine remaining nodes, resulting in less disk and network I/O on any given node.

To put it in a numbers perspective, if each node has 12 6TB drives, in the four-node cluster, each node might have to deal with moving around 24TB of data. Compare this to the 10-node cluster, where each node is only likely to need to deal with around 8TB of data. If you assume bonded 10 gigabit connection running at full throttle, that is a difference of about 4x in rebuild times.
Continuing to build on the same example, a ten-node cluster with 68 drives per node would not be advantageous in many circumstances, due to the sheer amount of data that would need to be moved around. In numbers, using the same 6TB drives, each node would hold about 408TB. Divide that across the nine remaining nodes, and each node could be dealing with around 45TB of data needing to be written. Take this back to the previously discussed four-node cluster, and it is not a desirable situation.

**DRIVE TECHNOLOGY**

Drives are the key element for each storage node in the cluster. It is a common practice to utilize solid state drives (SSDs) for the journals that front-end spinning media. This provides significant performance gains while keeping the per-GB cost at a reasonable level. The normal ratio for SSDs to spinning media is between 1:4 to 1:6.

For the primary storage medium, most clusters are built using 7200 RPM SATA or SAS drives. In more mission-critical environments, SAS provides slightly better error handling and a few additional benefits in regards to tuning, although generally at the cost of slightly higher power consumption. In most cases, SATA is the preferred medium for economic and power considerations.

When working with storage-dense chassis, it is important to know whether expanders are in use. This is important when considering the maximum throughput the system can generate, because the expander will reduce the throughput to the speed of the channel connecting to the SATA controller. The net effect being that it is possible that the link to the expander becomes the bottleneck when all drives are busy.

It is also possible to use only spinning media or to use only SSDs in a storage pool. Doing the latter is very common when building a higher performance pool. The use of a custom CRUSH map and ruleset allow the administrator to specify which pool goes to which devices.

It is generally recommended to place the operating system on mirrored drives. In the case of the monitor nodes, these should be SSD drives because the monitor daemons create a significant amount of fsync activity. Many systems now come available with M.2 for boot drives; this is a desirable location for the OS deployment.

**PROCESSOR**

It is highly recommended not to starve the processors on a storage node. Doing so can dramatically impact the ability to recover from failure and service requests and can, in general, create undesirable effects. To keep the processor in check, have 2Ghz per storage device in the system. In calculating the speed for processors that support Hyper-Threading Technology (HT) and have it enabled, the extra threads that show up as CPU cores in Linux count as a physical core would, so a 2Ghz 10 core CPU with HT would be calculated as:

\[
\text{10 cores } \times 2 \times 2\text{Ghz} = 40\text{Ghz of available CPU speed}
\]

This ensures that a reasonable amount of compute power is present to perform required processes. If nodes are to be loaded with extra compute-intensive items such as on-disk encryption, secondary services, and so forth, those items will need to be taken into account and provided for above and beyond the base allocation.

It is also important to note there is a preference for more cores rather than higher speed per core. This means that a 2Ghz 10-core processor is preferred over a 3.3Ghz 6-core due to the number of simultaneous threads that can be processed. There is also a slight preference towards a single socket versus a 2socket implementation due to slight penalties incurred for crossing the QPI links between the sockets.

**RAM**

Failure to place enough RAM in the system can result in swapping or even out-of-memory (OOM) conditions during critical events in the cluster. These events could include rebuilds, rebalancing and other actions where extra RAM is needed by the system. An OOM can result in any number of troublesome outcomes that degrade cluster performance. **The recommendation is to employ 1.5GB per TB of storage on a node.** This provides sufficient RAM for any number of back-end operations.
NETWORK
The network can be divided into two segments, the public network and the cluster network. The public network is used for communication with the monitors and clients, while the cluster network is only for OSD-to-OSD communication. In environments where possible, it is recommended to bond all ports using 802.3ad (LACP) and utilize VLANs to segment the traffic into cluster and public traffic. This has several advantages over dual physical network fabrics, including ease of administration.

When building a scale-out storage cluster, it is important to architect it in a way that allows for continual scalability. While Ceph takes care of this from the software perspective, it is important for the architect to take care in designing the node size, network layout and capacity, as well as which data protection schemes will be in place.

One important aspect of the network design is to ensure sufficient IP address space for the storage services. It is recommended to create a single storage subnet where all nodes, gateways and clients will connect. In many cases, this might entail using a range larger than a /24 if utilizing IPv4. While storage traffic can be routed, it is generally discouraged, to help ensure lower latency.

It is recommended to use redundant and scalable networking infrastructure such as the stacking switches depicted below. Utilizing LACP to create the bond provides the cluster with both availability and bandwidth aggregation. This configuration can survive failure of a cable, NIC or switch without stopping the flow of data.

The diagram in the top right shows a representative cabling with one node.

Another aspect of scalability is to consider whether higher speed network connections are warranted for the use cases being addressed by the SUSE Enterprise Storage cluster. While multiple 10Gb connections provide a strong and redundant network backbone, workloads with high aggregate throughput requirements measured in multiple gigabytes per second might warrant upgrading to multiple 40Gb/s or faster interfaces. It is also important to address the issue of scaling. As the cluster grows to more and more nodes, the network backbone becomes the bottleneck, making it necessary to ensure intercabinet links at least match or exceed the speed available to cluster nodes inside the cabinet.

General Infrastructure
Each node should also have a DNS A record that refers to the public interface. Consistent name resolution is important enough that it might be advisable to deploy a slave DNS as part of the storage infrastructure. This would allow for the cluster interfaces to be named and maintained local to the cluster, for example, node1.cluster.company.com for the cluster interface.

It is also recommended to make use of the Subscription Management Tool from SUSE. This service provides a local mirror of the SUSE repositories, allowing for rapid software deployment and updating. This software is part of the SUSE Linux Enterprise 12 SP1 distribution. For prior versions, more information can be found here: www.suse.com/documentation/sles-12/book_smt/data/book_smt.html
Physical Layout
It is important to understand the general concept of an availability zone. An availability zone represents a particular configuration in terms of ensuring availability. This can include power, network and cooling, and provides a logical manner in which to divide scale-out infrastructure components.

The diagram below shows a representative rack elevation for a single rack cluster.

The two switches are placed at the top of the rack. The three servers immediately below the switches are the monitor nodes. The four servers at the bottom of the rack are the OSD nodes. In a multi-rack configuration, the monitor nodes should be distributed across availability zones at the rack level.

Use Case Specifics
SUSE Enterprise Storage provides a flexible infrastructure that is able to service a number of use cases. While some use cases use common architectures, others benefit from specific configuration options. Specific use cases and related configuration guidance is outlined in this section.

Block or Object Store for Active Archive and Disk-to-Disk Backup
Active archive and disk to disk backup (D2D) are key uses of bulk storage in today’s data center. These share characteristics of not having stringent latency requirements and being oriented towards supporting a number of parallel readers and writers. The general recommendation for use is for large-item storage rather than billions of tiny files. This favors Ceph’s strength for massive horizontal scalability.

In these scenarios, it is suggested to architect using higher density chassis while still meeting the minimum recommended cluster size of seven storage nodes. These nodes should follow the hardware recommendations made in this document, with the possible exception of including SSD journals. The reason for this possible exception is simple: the workloads are not latency sensitive and this is precisely what the journals do, reduce latency. This being the case, if the iSCSI protocol is utilized, it is still recommended to deploy journals. An additional note for iSCSI is that multiple gateways should be deployed for redundancy.

Additionally, when utilizing the object storage protocols, greater storage efficiency can be gained through the use of erasure coding. A fairly common set of erasure coding settings would have $k=8$ and $m=3$, with a total of 11 or more nodes needed for this configuration in an ideal scenario. This provides for the failure of three devices without data loss, and when combined with a CRUSH map that matches the cluster topology, allows for the loss of three nodes without data loss.

OpenStack
The most commonly used storage for OpenStack is Ceph. OpenStack can interface via both block, cinder for deployed storage and object protocols, glance for the image repository. While the glance repository does not impose significant performance requirements, the block repository for virtual images does. This is most commonly handled by the RBD protocol, which maximizes the strengths of Ceph’s distributed architecture.

When architecting a solution for OpenStack, it is important to understand the workloads that will be utilizing the storage and their requirements. At the minimum, it is a firm recommendation to utilize SSD journals for the block storage component. This helps to ensure that the storage is responsive to the guest operating systems. It is also strongly recommended to ensure that there is sufficient bandwidth on the storage network for all of the demands that will be placed on it by the guest workloads.

Virtual Machines
Similar to OpenStack is the use of SUSE Enterprise Storage to host more traditional virtual system images. This entails supporting block storage protocols that provide adequate responsiveness for the systems deployed. In general, it is recommended to utilize less storage-dense options because the aggregated capability of the storage servers provides a more responsive infrastructure. For these workloads, SSD journals are highly recommended. In some cases, using an all-flash storage environment might be desirable. A key item to note is that SUSE does not recommend deploying Ceph as storage for latency-sensitive workloads such as high-performance databases at this time.
In order to support Microsoft or VMware virtualization workloads, it is necessary to utilize the iSCSI protocol. The best practice for deployments of these workloads includes having multiple iSCSI gateway systems that are all addressing the same LUNs. The multi-pathing policy in both cases should be round-robin, allowing the I/O to aggregate across all gateways in a fairly even distribution. KVM on Linux environments should leverage the native RBD interface for optimal performance.

**Video Surveillance**

Video surveillance and archive systems make use of a wide array of protocols. In most cases, the system is receiving many streams that have individually relatively low requirements, but aggregate to a much greater need. An example would be a site with 200 surveillance cameras recording at 1080p using H.264 encoding. Each stream at 30 frames per second is recommended to have 4MB/s of bandwidth. Aggregated out, that is a constant writing rate of 800MB/s. This is well within the capability of SUSE Enterprise Storage by itself, but if it is then combined with other workloads or even the video being analyzed, viewed, and so forth, it is necessary to ensure that there is sufficient throughput capability to the cluster to accommodate the needs.

It is generally recommended to deploy capacity-dense chassis using SSD journals in front of spinning media. These systems should be carefully sized from a networking perspective to provide three to four times the required frontend bandwidth. This will ensure that there is sufficient room for growth, backups, and so forth to occur without impacting the system’s ability to record the inbound video streams. It is also important to ensure that there are sufficient gateways if using object storage or iSCSI, and also deployed in such a way to provide for failure or unavailability of at least half the gateway nodes while still meeting the throughput requirements.

**Additional Information**

For further information on SUSE Enterprise Storage, please visit: [www.suse.com/products/suse-enterprise-storage/](http://www.suse.com/products/suse-enterprise-storage/)