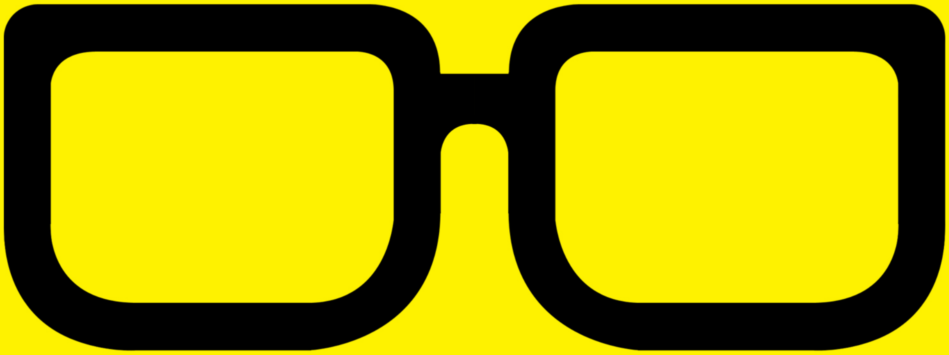
SPONSORED BY

SUSE®

GEEK GUIDE



Ceph: Open-Source SDS

LINUX™ JOURNAL

Table of Contents

TED SCHMIDT is the Senior Project Manager and Product Owner of Digital Products for a consumer products development company. Ted has worked in Project and Product Management since before the agile movement began in 2001. He has managed project and product delivery for consumer goods, medical devices, electronics and telecommunication manufacturers for more than 20 years. When he is not immersed in product development, Ted writes novels and runs a small graphic design practice at http://floatingOrange.com. Ted has spoken at PMI conferences, and he blogs at http://floatingOrangeDesign.Tumblr.com.

GEEK GUIDES:
Mission-critical information for the most technical people on the planet.

Copyright Statement

THE MATERIALS ARE PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT. The Materials are subject to change without notice and do not represent a commitment on the part of *Linux Journal* or its Web site sponsors. In no event shall *Linux Journal* or its sponsors be held liable for technical or editorial errors or omissions contained in the Materials, including without limitation, for any direct, indirect, incidental, special, exemplary or consequential damages whatsoever resulting from the use of any information contained in the Materials.

About the Sponsor

SUSE, a pioneer in open-source software, provides reliable, interoperable Linux, cloud and storage infrastructure solutions that give enterprises greater control and flexibility. More than 20 years of engineering excellence, exceptional service and an unrivaled partner ecosystem power the products and support that help our customers manage complexity, reduce cost and confidently deliver mission-critical services. The lasting relationships we build allow us to adapt and deliver the smarter innovation they need to succeed—today and tomorrow.

Ceph: Open-Source SDS

TED SCHMIDT

Introduction

Despite the recent trend of businesses moving to cloud-based storage and SaaS applications, businesses of all sizes will see significant benefits from pursuing a strategy that mixes cloud-based information services for standard or back-office functions with in-house management of mission- and strategy-critical data. Data has become more important than ever, and bigger than ever, but growing data comes with increased costs and performance issues. Enterprises are looking for data management solutions that are scalable, resilient and that can be built on commodity hardware.

The current trend is toward commodity hardware and solid-state drives rather than disk, and away from legacy NAS and SAN solutions, where proprietary software and hardware keep you from realizing the cost benefits of using commodity hardware. In short, businesses are looking for a way to separate storage hardware from the software that manages it, so they can use a single, efficient software solution that will manage any vendor's hardware.

Enter software-defined storage, or SDS.

But, what is SDS? There are a lot of opinions on what exactly SDS is and how it benefits the enterprise. In this ebook, I explore the background of SDS and define what it really means. I look at some of the characteristics of SDS, examine Ceph, an open-source SDS solution, and discuss why an open-source solution that leverages commodity hardware might be your best answer.

Overview: What Is Software-Defined Storage?

Software-defined storage is a relatively new category of software storage products. Although many consider SDS to be a natural evolution of virtualization and software-defined networking, SDS is, put simply, a virtualization technique aimed at reducing the costs of managing growing data stores by decoupling storage management software from its hardware to allow centralized management of cheaper commodity

So, to qualify as SDS, a solution must run on generic, industry-standard hardware, without any proprietary hooks that ultimately lead to limitations.

hardware. Beyond this over-simplified definition are nuances that create big differences in the solution you are ultimately getting. So, it's important to understand those nuances.

Gartner puts it clearly by stating that an SDS solution will use software to separate and abstract storage capabilities that are pulled from industry-standard, commodity hardware, with the aim of delivering higher quality of service while reducing costs. IDC adds to this idea of hardware agnosticism by defining SDS as "any storage software stack that can be installed on commodity resources (x86 hardware, hypervisors, or cloud) and/or off-the-shelf computing hardware". So, to qualify as SDS, a solution must run on generic, industry-standard hardware, without any proprietary hooks that ultimately lead to limitations. Gartner agrees with IDC's take on SDS by stating that SDS works regardless of class of storage. Use of commodity hardware is key to the ROI benefits offered by true SDS.

Now, let's look at some of the other characteristics of an SDS solution. Although SDS does provide for pooling of storage, to be true SDS, the solution also has to provide the following additional features:

- Establishment of policies for managing data services as well as storage.
- Metadata tagging for managing data services and storage.
- Dis-aggregation of data services and storage.
- Automated management of storage.
- UI that provides self-service.

Additional features that can be part of SDS but are not required:

- Use of non-proprietary hardware including industry-standard hardware.
- Enhances existing functions of specialized hardware.
- Scales out storage.
- Incremental build out of data services and storage solution.

Based on this list, it's easy to see that SDS can take a number of different forms depending on your budget, requirements or other factors. However, the separation of management software and services from hardware creates a solution that becomes scalable. Additionally, simplifying the indexing of unstructured data using object services based on representational state transfer (REST) is also key, as are filesystems that improve data protection and ease of capacity optimization, and free interaction of data services to allow the separation of data and scalability.

Benefits of SDS

The promise of SDS is that it will enable enterprise IT to provide a more on-demand, scalable and agile experience for business users with no single point of failure, while simplifying their storage management and reducing CAPEX and OPEX.

Single Management Interface: SDS brings great flexibility to the IT organization because it provides a single software interface to potentially all storage hardware, regardless of vendor. This means functions such as creating a volume, establishing RAID protection, implementing thin provisioning and tiering of data all can be done through a single interface. IT administrators don't need to be retrained on each storage system. This flexibility allows IT organizations to purchase storage systems that are specific to a task without adding to infrastructure management.

Plenty of storage vendors make excellent storage hardware, but they have not invested in the accompanying storage software. These vendors often are classified as Tier-2 vendors, but when coupled with SDS, they can match Tier-1 hardware vendors feature for feature.

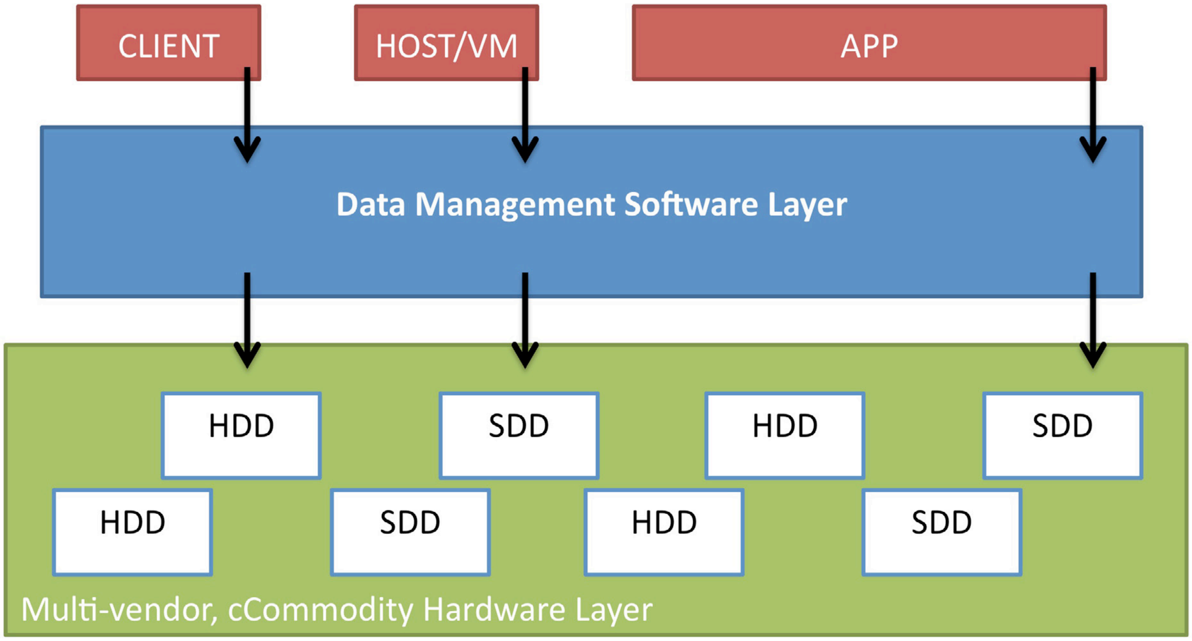

CLIENT
HOST/VM
APP
Data Management Software Layer
HDD
SDD
HDD
SDD
HDD
SDD
HDD
SDD
Multi-vendor, cCommodity Hardware Layer

FIGURE 1. Basic SDS Architecture

Reduced CAPEX: One of the benefits offered by SDS is that it separates the purchase of management software from storage hardware, so you effectively spend less capital when you need to add or upgrade storage hardware, because you don't need to worry about the added cost of the management software that comes with proprietary solutions.

With proprietary solutions, any time you need to upgrade your storage hardware, you have to buy new software with the hardware upgrade. This leads to increased training costs and even poses potential procedural risks when there are significant differences in the releases of management software. On the other hand, if the storage software is unchanged over generations of hardware, you're paying for something you don't need. In other words, you're paying for something you already own!

Scalable: As big data continues to grow, finding cost-effective ways to gain value from all that information will be a critical deciding factor for companies that want to remain viable in the future. Because you can use commodity hardware to grow your data architecture as your data grows, SDS provides a much more flexible and lower cost solution, no matter how massive your data storage needs become.

No Single Point of Failure: The no-single-point-of-failure design principle asserts simply that no single part can stop the entire system from working. With traditional dedicated storage solutions, a storage array can't borrow capacity from another when demand for storage increases, which leads to data bottlenecks and a single point of failure.

With SDS, however, that risk is avoided. Remember, SDS uses commodity storage devices and provides shared storage capabilities, such as mirroring and replication. SDS also eliminates the need for dedicated storage arrays and storage area networks. Because SDS distributes the workload across multiple devices, if any single device or node fails, it doesn't bring down the entire system.

I've defined SDS as a solution that separates management software from the commodity hardware it manages, and I've described the benefits of moving to an SDS solution, including cost reduction, simplification, scalability and avoidance of a single point of failure through distributed workload. Now, let's take a look at the industry leader in SDS: Ceph.

Ceph

Ceph is an open-source, distributed object store and filesystem originally designed by Sage Weil for his doctoral dissertation at UC, Santa Cruz. In 2012, Weil started Inktank Storage, which Red Hat acquired in 2014. In 2015, to assist the Ceph community of developers in creating and promoting a unified vision for open-source SDS technology, individuals from organizations including Canonical, CERN, Cisco, Fujitsu, Intel, Red Hat, SanDisk and SUSE formed the Ceph Community Advisory Board.

Designed to deliver extraordinary performance, reliability and scalability, Ceph provides interfaces for object, block and filesystem storage to store data on a single, unified system. It provides unified distribution of storage without a single point of failure, is scalable to the exabyte level, and because it's open source, it's available to anyone for free.

Ceph deployment is fairly straightforward. You start by setting up your network, every machine or server that will be part of the environment as a Ceph Node, and the Ceph Storage Cluster that requires at least one Ceph Monitor (for added fault tolerance and reliability, Ceph supports clustering of monitors), which maintains maps of the cluster state. You also need at least two Ceph Object Storage Device (OSD) Daemons to store data, which provide the Ceph Monitor with data and handle replication, recovery balancing and backfilling. If you're going to run any Ceph filesystem clients, you also should plan on setting up the Ceph Metadata Server

Part of what makes Ceph work is that the Daemons and Clients have knowledge of the topology of the cluster.

(MDS). MDS does just what it sounds like—it stores metadata for the Ceph filesystem.

Ceph then uses storage pools to store data. It calculates which placement group gets the data and which OSD should store the placement group using something called the CRUSH (Controlled, Scalable, Decentralized Placement of Replicated Data) algorithm, which enables the Ceph Storage Cluster to scale, rebalance and recover dynamically as needed.

Architecture: Let's look at the basic components of Ceph a little more closely, starting with the storage cluster. The RADOS (Reliable Autonomic Distributed Object Store)-based storage cluster in Ceph is composed of a Monitor and at least two OSD Daemons. The Ceph Monitor maintains maps of the state of the cluster to ensure high availability in case a Monitor Daemon fails, and the Daemons monitor their own state, as well as each other's state, and report back to the Monitor. Instead of having a central lookup table to reference, the Daemons and storage cluster Clients use CRUSH to compute information about data. CRUSH basically distributes the workload by managing the data objects across the Clients and Daemons in the cluster.

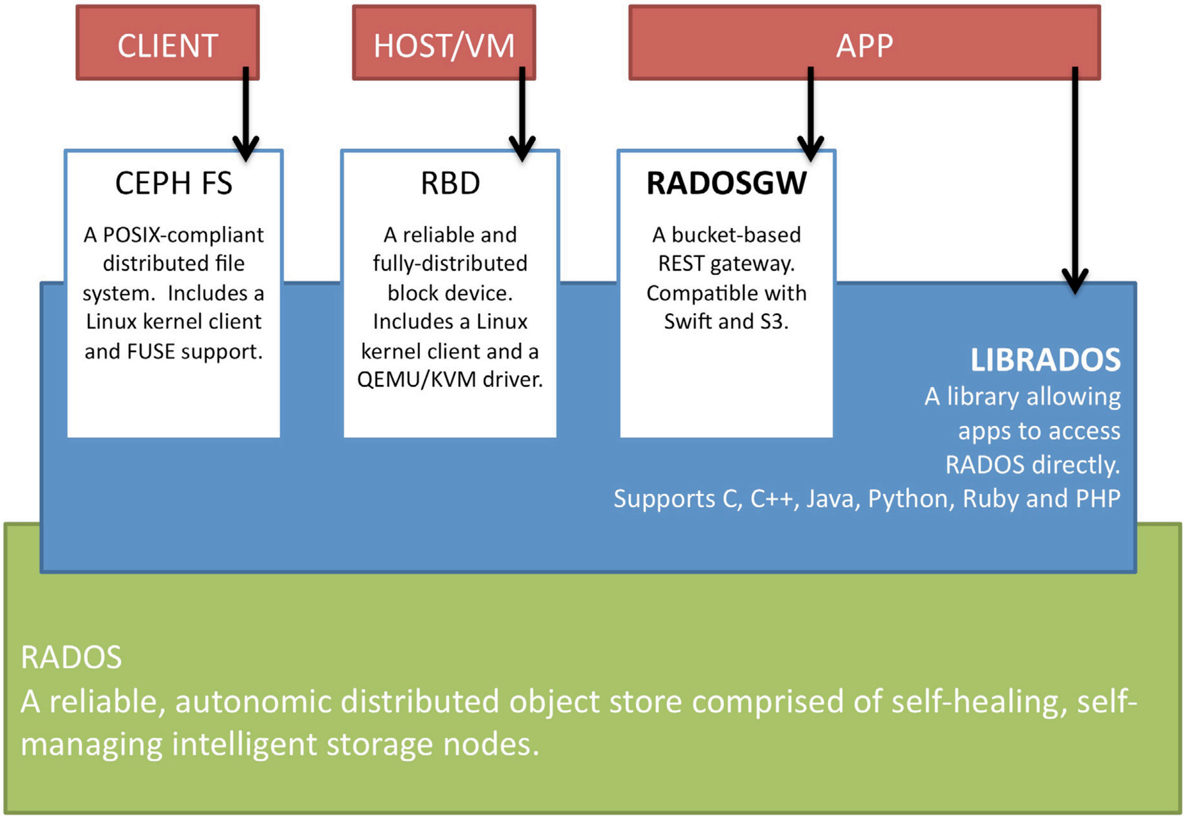

CLIENT
HOST/VM
APP
CEPH FS
A POSIX-compliant distributed file system. Includes a Linux kernel client and FUSE support.
RBD
A reliable and fully-distributed block device. Includes a Linux kernel client and a QEMU/KVM driver.
RADOSGW
A bucket-based REST gateway. Compatible with Swift and S3.
LIBRADOS
A library allowing apps to access RADOS directly.
Supports C, C++, Java, Python, Ruby and PHP
RADOS
A reliable, autonomic distributed object store comprised of self-healing, self-managing intelligent storage nodes.

FIGURE 2. High-Level CEPH Architecture

Part of what makes Ceph work is that the Daemons and Clients have knowledge of the topology of the cluster. This topology is contained in the Cluster Map. The Cluster Map is composed of five maps: Monitor Map, OSD Map, PG Map, CRUSH Map and the MDS Map. The Monitor Map contains the location of each monitor. The OSD Map contains a list of OSDs and their status in addition to a list of pools, replica sizes and PG numbers. The PG Map contains details on each placement group (PG). The CRUSH Map contains a list of storage devices, the failure domain hierarchy and hierarchy rules. The MDS Map contains the metadata pool and a list of metadata servers and their status. Every map contains its current epoch, when it was created and when it last changed. Whenever Ceph Clients want access to data, they first must obtain a copy of the Cluster Map from a Ceph Monitor.

As mentioned earlier, the OSD Daemons are aware of each other—something referred to as being "cluster aware". Because of this, OSD Daemons can interact with each other and Ceph Monitors, while Ceph Clients can interact directly with OSD Daemons. This architectural design feature allows OSD Daemons to use CPU and RAM of the cluster nodes to perform tasks at the exabyte scale that normally would cause bottlenecks. OSD Daemons now can service Clients directly, which increases performance and system capacity at the same time; Clients no longer wait on a centralized server. It also means that Ceph Monitors are lightweight processes, because someone else is always checking on them. Data scrubbing also becomes more thorough because the OSD Daemons can compare objects in the placement groups of other OSD Daemons. Finally, OSD Daemons relieve Ceph clients from the need to perform any data replication, because OSD Daemons can replicate data to however many other OSDs exist.

Hardware Considerations

Ceph was designed to run on commodity hardware, and that makes building and maintaining massive data clusters economically feasible. But, there are a few things to consider before you get started building your new cluster—things like whether to include failure domains, potential performance issues and matching hosts to OSD Daemons to gain the most efficiency (for instance, it's typically a good idea to run an OSD Daemon on hardware configured for that type of dæmon). A great source of information when planning out your Ceph implementation is http://ceph.com or, at the time of this writing, the most current recommendations by SUSE at https://www.suse.com/docrep/documents/w3lcomzhs7/suse_enterprise_storage_architectural_overview_with_recommendations_guide.pdf.

SUSE recommends understanding a few basic considerations before sizing your hardware, including:

- Single and aggregate thread throughput expectations.
- Minimum, maximum and average latency expectations.
- Acceptable performance degradation during failure/rebuild events.
- Read/write ratios.
- Working data set size.
- Access protocols or methods.

With this data in hand, let's explore a few basic, but more current, hardware configuration recommendations for implementing Ceph.

Let's begin with the trickiest, sizing the gateway services. Depending on what they are doing, they could co-locate on a monitor node, or require dedicated nodes, depending on cluster size. Ceph OSD Daemons need some processing power as well, and you can really damage performance during normal and degraded operations. The best thing here is to consult http://www.suse.com for the latest recommendations. Because Ceph Monitors primarily are concerned only with maintenance of the Cluster Map, they can run on just a few, quick cores, maybe two @ 2.3GHz with about 8GB of RAM.

All this being said, keep in mind any other processes you may have running on your hardware that could compete with Ceph processes. Make sure any other VM leaves enough resources for your Ceph processes, and run your metadata servers and OSDs on a separate host if at all possible.

RAM: More RAM is always a good thing. You should reserve at least 2GB per TB of node storage for each Monitor and Metadata server, because you want them to serve data as quickly as possible. OSD Daemons, on the other hand, really require 8GB for normal operations. Again, be careful! Their demands during recovery can spike, so plan accordingly.

Drives: Since SSDs have no moving parts and can show access speeds of more than 100 times that of most hard disk drives, despite the increased cost, they are worth considering for journals, where Ceph uses less storage. But, be careful; SSDs do have some limitations. For instance, OSD journaling means write-intensive semantics and handling multiple write requests at the same time. This results in a need for speed. Differences in sequential write throughput can have a serious impact when you're storing multiple journals for multiple OSD Daemons. So, even though they have no moving parts, cheaper SSDs that are slower actually can perform worse than a faster, high-end hard drive. That being said, you'll still want to take advantage of the lower cost of spinning drives. SUSE recommends a solid-state to spinning ratio of between 1:4 and 1:8.

If you decide to go with SSDs, make sure that any partitions are aligned to avoid any degradation in data transfer speeds. Although it may be a best practice to partition your drives, bad partition alignment can seriously slow things down. Always check the performance metrics of SSDs before you buy to make sure they provide the speeds you're looking for, and even test SSDs for performance before making the purchase.

Networks: Recently updated recommendations from SUSE for network configuration now include stacking top-of-rack network switches and creating LACP (802.3ad) bonding groups across the switches to protect against failure. They also recommend you create a single bonded interface on the OSD node, using mode 4 (also 802.3ad) and VLANs to segment traffic logically.

You'll want to be sure that the cluster has enough bandwidth for replication, recovery and other back-end actions, so it's important to design more bandwidth for the cluster than the outbound client links. To give the back-end VLAN performance a higher priority than the client-facing VLAN, it's also a good idea to use QoS, assuming your switches will support it.

You'll see a significant boost in data payload by using jumbo frames for the back end, and both private and public networks, along with better throughput to the storage.

You'll see a significant boost in data payload by using jumbo frames for the back end, and both private and public networks, along with better throughput to the storage. But don't mix frame sizes on the same VLAN or try to convert routing devices from standard to jumbo MTUs, or you'll likely overrun the capability of your processors.

Other Considerations

Here are a few more things to consider when evaluating and planning hardware for CEPH:

- Don't be afraid to spend some money to isolate failure domains. There are a lot of failures, from a crashed OS to a failed NIC or power supply that can prevent access to an OSD Daemon. Avoid failure domains as much as your budget can afford.

- Make sure that the total throughput of your OSD hard disks (or SSDs) doesn't exceed the network bandwidth needed to service a client's read/write requirements.

- Make sure each host is not overloaded with data. If a host fails, and you exceed the full ratio, Ceph can halt operations in an effort to prevent data loss.

- Make sure that the kernel is up to date any time you run multiple OSD Daemons on the same host. Since Linux kernels typically default to a smaller number of maximum threads, make sure the pid max is set to a higher number of threads when you want to run multiple OSD Daemons on a single host. Ceph.com provides performance specs for running multiple OSDs on a single host.

- If you can't run the OS and volume storage on separate disks, create a partition for your volume storage and a separate partition for the OS.

- If you expect to reach petabyte scale for your data storage, you still can use commodity hardware for production clusters—just be sure to plan for the heavy load by increasing RAM, CPU and storage.

- Always deploy Ceph on newer releases of Linux that provide long-term support. If you use the B-tree filesystem, you should go with Linux kernel 3.14 or later. To be sure, check Ceph.com for current release notes.

- Because multiple versions of Ceph are available, and because Ceph is tested to varying degrees for each of those versions on different OS versions, it's best to check Ceph.com to learn more about that as well.

Minimum Hardware Recommendations

If you haven't figured it out by now, Ceph can call for a lot of redundancy. But, because Ceph can run on inexpensive commodity hardware, you can build small development and production clusters that can run very well with inexpensive hardware. Before implementing Ceph in production, you should become familiar with the detailed hardware recommendations and operating system recommendations found at https://www.suse.com/documentation/ses2/book_storage_admin/data/cha_ceph_sysreq.html.

How to Get Started

There is no cookie-cutter approach to implementing Ceph. It's typically an iterative process that begins by understanding the needs of your business and moving less-critical applications to your Ceph cluster first. This allows you to learn as you go and develop a standardized migration methodology so you can maximize the benefit when you get to your mission-critical applications.

Start by determining where your pain points are. What is the business demanding, and how does that translate into future needs for access to data? What is your current mix of data, and how is it structured? If your data is unstructured, you can quickly realize real benefits from Ceph. Are you talking about a single, monolithic application, or does your business use a mix of smaller, discrete applications? Shops with a broader mix of business applications and big workloads tend to be better candidates for Ceph. How are you deploying your systems now and in the future? Again, Are you going to be using big data? Do you rely heavily on data analytics? Ceph was designed to manage massive amounts of data quickly.

Ceph can help if you have a mixed model, because it brings everything together. Are you going to be using big data? Do you rely heavily on data analytics? Ceph was designed to manage massive amounts of data quickly. It also was designed to protect and secure massive amounts of data.

Here are some application and data characteristics that will benefit greatly from a Ceph solution:

- Broad application mix.
- Big workloads.
- Unstructured data.
- Multiple deployment models (physical, virtual, cloud).
- Big data or analytics.
- Need to keep data secure and protected.
- Need to keep data around a long time.

SUSE Enterprise Storage

Earlier, I discussed some of the characteristics of software-defined storage. One of the key characteristics, and one that Ceph is based on, is the idea of device agnosticism. That is, since a benefit of implementing an SDS solution, particularly Ceph, comes from being able to employ low-cost commodity hardware, and most commodity hardware vendors do not provide a proprietary storage management solution, you want to find an SDS solution that will manage commodity hardware. Although Ceph is a great open-source solution, many IT organizations require enterprise-ready software that they know will meet their reliability, availability and scalability

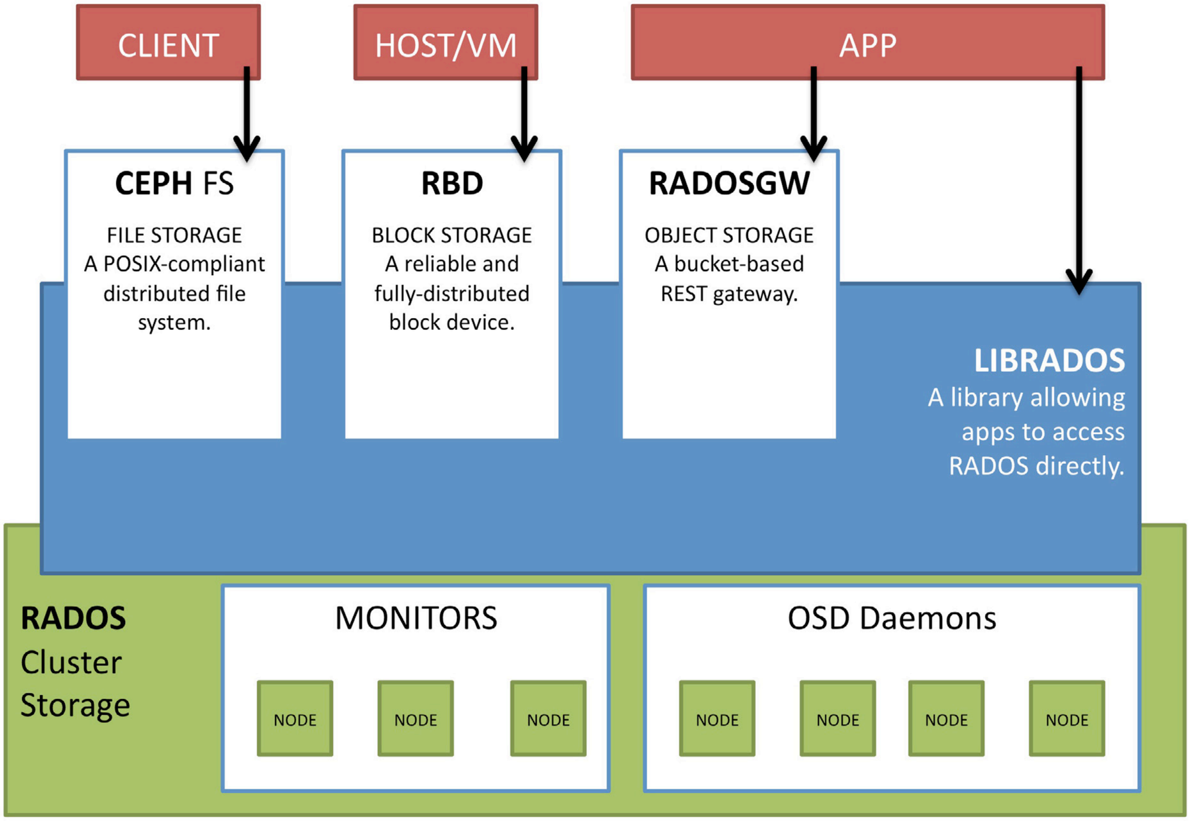

CLIENT
HOST/VM
APP
CEPH FS
FILE STORAGE
A POSIX-compliant distributed file system.
RBD
BLOCK STORAGE
A reliable and fully-distributed block device.
RADOSGW
OBJECT STORAGE
A bucket-based REST gateway.
LIBRADOS
A library allowing apps to access RADOS directly.
RADOS
Cluster
Storage
MONITORS
NODE
NODE
NODE
OSD Daemons
NODE
NODE
NODE
NODE

FIGURE 3. SUSE Enterprise Storage Architecture

True to Ceph design, SUSE Enterprise Storage algorithms automatically distribute data storage, monitor your system's data utilization and optimize data placement without manual intervention.

needs and is supported by a trusted vendor.

SUSE provides exactly such a solution with its SUSE Enterprise Storage software. It's a Ceph-based, data storage management solution that allows you to build cost-effective and highly scalable storage using commodity servers and disk drives. Working with independent hardware vendors, SUSE delivers quality, scalable, supported, enterprise-ready storage solutions.

Scalable to the petabyte level, SUSE Enterprise Storage is designed to increase your capacity and improve system performance while avoiding any level of system disruption.

True to Ceph design, SUSE Enterprise Storage algorithms automatically distribute data storage, monitor your system's data utilization and optimize data placement without manual intervention. Because it is powered by Ceph, SUSE Enterprise Storage is designed without any single points of failure.

SUSE Enterprise Storage also provides unified block and object access and is tightly integrated into the OpenStack virtualization infrastructure to connect block devices to virtual machines. With SUSE Enterprise Storage, you can boot new virtual machines rapidly with fault-tolerant, highly available, enterprise disk storage and integrate with Kernel Virtual Machines (KVMs).

SUSE Enterprise Storage allows for performance gains while reducing capital expenditures by enabling the use of commodity hardware. With unlimited scalability in a self-managed environment, it also helps customers keep operating expenses in check. Implementing an SDS solution with SUSE Enterprise Storage enables you to improve your business agility, providing the adaptable and fast storage access your business applications need, with a cost-effective, easily manageable and highly scalable storage model.

You can learn more about SUSE Enterprise Storage at https://www.suse.com/products/suse-enterprise-storage.

Conclusion

In this ebook, I discussed the emerging need for enterprises to be able to separate storage management software from the hardware that it manages, a concept referred to as hardware agnosticism. I also described software-defined storage in some detail and talked about why hardware agnosticism is key to its definition. I introduced Ceph, an open-source SDS, and pointed out how Ceph is key if your organization wants an SDS solution that provides scalability, performance, redundancy, self-management and, of course, device agnosticism.■