A Saga of Smart Storage Devices

An Overview of Object Storage

MATTHEW W. BENJAMIN, CASEY BODLEY, ADAM C. EMERSON, AND MARCUS WATTS

Matt Benjamin is chief architect of CohortFS, and a founder of CohortFS, LLC. Matt is a contributor to numerous open source software packages and tools, including the NFS Ganesha and OpenAFS. Matt holds a master's degree from the University of Michigan, and a bachelor's degree (summa cum laude and Phi Beta Kappa) from the University of Missouri. matt@cohortfs.com

Casey Bodley studied computer science at Eastern Michigan University. He then worked for the Center for Information Technology Integration at the University of Michigan to develop a Windows client for NFSv4.1. He joined the CohortFS team in 2012, and has been working on parallel metadata enhancements to Ceph. casey@cohortfs.com

Adam C. Emerson studied mathematics at the University of Michigan. While at CohortFS he has worked on the Ganesha NFS server, especially improving support for NFSv4.1 and pNFS. He also collaborated in the CohortFS design process for data placement, encryption, and metadata striping. aemerson@cohortfs.com

Marcus Watts is a programmer at CohortFS. He previously worked for a large education institution where he worked with AFS and identity management. Way back when, he wrote a computer conferencing program, PicoSpan, which was the basis for the Well in California. mdw@cohortfs.com

Object storage systems have become quite popular, with implementations ranging from Amazon's S3 to backends for NFSv4.1. We describe the history of object storage, the practice and standards in use today, and work being done by groups such as the Ceph project, as well as some of our own development.

Object storage fills a gap between block storage and file systems. Simple block storage consists of fixed-size blocks as provided by traditional disk drives. Blocks can be accessed randomly but must be allocated by some scheme. Most modern file systems provide a directory hierarchy that contains variable length byte-array "files" as the leaves. Modern network applications frequently need a higher abstraction than can be provided by simple block storage, yet don't need all the complexity and limitations of a traditional file system. Object storage fills this gap by providing for larger variable-sized segments and a simple flat-naming scheme.

The most common object abstraction provides one or more collections of uniquely named objects of arbitrary size associated with some amount of metadata. Object storage tends to emphasize getting or putting entire objects, rather than reading and writing byte-ranges as is more common in file systems.

We will talk about object storage systems as two groups, which we will call the "device-like family" and the "HTTP-like family." The device-like family exposes a cluster of individual devices to clients. One example of device-like objects is the SCSI T10 object standard. The HTTP-like family presents a single interface, hiding details of how data is distributed. The best-known example of the HTTP-like family is Amazon's S3.

History and Character

Our two divisions of object storage grew up independently but have crossed over and stimulated each other.

The Device-Like Family

The modern device-like object storage paradigm traces back to work by Garth Gibson and others on the NASD (Network Attached Secure Disks) project at CMU, whose goal was the creation of a scale-out storage system. They designed intelligent drives storing variable-length objects with access being granted by a cacheable token; this allowed scale-out similar to a SAN but without clients having to be involved with actual block allocation, and so allowing disk devices to perform on-platter optimization [3]. In NASD (and its direct descendant PanFS) the file server was responsible for assigning objects to devices; NASD used a middleware to stripe virtual objects across real devices. Even in the beginning, objects were used for more than just building file systems. For example, the NASD project built a distributed streaming MPEG2 server. The SCSI T10 committee standardized the Object Storage Device command set (drawn from the NASD model), a second version has been finalized (OSDv2), and a third is currently in development.

Further research in the device-like family of object storage focused on decentralizing placement of objects on devices [4]. Other work included scaling objects to more devices as the amount of storage grew, and responding to failures and gracefully reorganizing data as devices were added or removed, as in the RUSH [5] family of algorithms. The Ceph project builds on previous work, with a cluster of OSDs cooperating to perform automatic replication, recovery, and snapshots.

CleverSafe implements its own object storage system (with a proprietary protocol) where object names are effectively hierarchical addresses within a cluster. CleverSafe makes heavy use of Cauchy Reed-Solomon erasure codes [7] for fault tolerance as well as information dispersal. Information dispersal starts with a piece of data and derives multiple chunks from it, some number of which are needed to reassemble the original. CleverSafe optionally performs encryption, integrity, and compression as part of the write operation.

The HTTP-Like Family

The most successful (and current de facto standard) representative of the HTTP-like object storage family is Amazon's S3, which, like the Elastic Compute Cloud, was launched to expose and sell access to the global infrastructure Amazon developed to run its own business. S3's operations (getting, putting, and deleting, generally of entire objects at once) fall naturally out of the common REST architecture, which structures APIs around the standard methods of HTTP [2]. This gives S3 a high-level abstraction free from many assumptions or implied structures, similar to T10 OSD. S3 provides a flat namespace of objects within "buckets," which both partition objects into flat namespaces and dictate policy.

S3 has been enormously successful, not just as a service but as an API, and it has been adopted by other cloud service providers and by software such as the CloudStack framework and the Eucalyptus cloud computing system.

OpenStack's Swift service fills a similar niche. Even though it provides storage implemented by members of a cluster to members of that cluster, all requests go through an HTTP proxy server that hides the details of distribution and abstracts away the clustered nature of access from the client.

Hybrid Models

As both these families have been developed, they've borrowed from each other. Ceph's RADOS protocol implements object pools that function much like S3's buckets, and they map directly onto buckets in the RADOS Gateway, a Web service that hides the clustered nature of Ceph behind a Web proxy.

Huawei's Universal Data Storage goes one step further, selling hardware (clusters of smart disk drives) that speaks S3 to clients while providing enterprise functionality and management.

Anonymity Networks

Anonymity networks such as Freenet and GNUnet have converged on the object-like semantics of publishing and retrieval of blobs of data in a flat namespace on a wide-scale cluster. Clients interact with the individual nodes on the peer-to-peer (or friend-to-friend) network, but may have their interactions with the ultimate endpoints obscured by layers of onion routing and cover traffic depending on their security settings. New designs (referred to in GNUnet documentation and source as "multicast") for trusted replication among peers allow HTTP-like functionality, such as resilience or distributed service of resources in high demand, while preserving anonymity and privacy.

Current Uses

Many object storage systems can be used as arbitrary key-value stores with good performance for large values. T10 OSDv2 is a notable exception as it uses 64-bit integers to name objects within a partition; it requires an index, which may be implemented in the object system, to link more interesting names to integers. Object stores are also often used as building blocks for richer systems.

Database Integration

BLOB (Binary Large OBject) fields store mostly uninterpreted data in database records and have always been awkward due to their large size, which can drive other data out of cache in the database client. The BLOBs themselves are often served more slowly than would be ideal since they have to be pulled through that database connector interface. Many database programmers address this by storing objects in files and then storing the file names in the database.

One major downside of storing BLOBs as files is that most file systems don't offer the same reliability, integrity, or replication features that cover data stored in the database. Developers are using object stores and any replication and reliability that the store in question might provide to get around this limitation. Often the object will be named with a hash of the BLOB's content to provide implicit integrity checking and deduplication. This approach has become so popular that it's starting to become integrated into database backends. OblakSoft's Cloud Storage Engine for MySQL introduces a "WEBLOB" field type that integrates storage of BLOBs using Amazon's S3 protocol directly into the database. Using HTTP-accessible objects specifically also allows Web assets to be displayed to a client by passing a URL, without proxying the data through the application.

Streaming Storage

Video streaming from object stores was prototyped with MPEG2 on NASD but hit great commercial success with Netflix's adoption of Amazon S3 to back its streaming video service. This has become successful enough that Amazon has pursued the streaming market by adding RTMP access to S3 objects [1].

Virtual Machine Images

One of Ceph's biggest current successes is the RADOS Block Device. This is a set of conventions for organizing Ceph objects into disk images that can then be booted from or mounted by virtual machines in a cloud environment. This allows the use of the object store's capabilities, such as snapshots and replication, to provide checkpointing and resilience. Snapshot layering provides a crude approximation (due to snapshots being read-only) for copy-on-write storage and deduplication.

File Systems

OSDv2 is used by Panasas in PanFS and by the free ExoFS project as the backing store for their file systems. Ceph follows this same approach, building a file system on top of the RADOS object access protocol.

The S3 FUSE utility builds a file system on top of cloud-based storage, and Tahoe LAFS's RAIC (Redundant Array of Inexpensive Clouds) plans to build a highly reliable, secure file system that straddles the object storage systems of multiple cloud providers for reliability in the event of a provider's failure.

pNFS

T10 OSDv2-based object-backed file systems have been standardized as a scale-out and reliability component of NFSv4.1 through Parallel NFS (pNFS). pNFS introduces the concept of recallable layouts to represent both permission to and details on how to access data directly at the point that it is stored [8]. Clients then access back-end storage directly without going through a front-end server. pNFS allows differing access protocols, like striping data over several NFSv4 servers, accessing data as block ranges on SCSI devices, and arranging data in recursive RAID configurations over T10 objects.

The OSD layout type lets clients be aware of, participate in, and take advantage of replication and erasure coding. Clients can read stripes from multiple devices for improved speed or perform erasure coding at the time of writing.

Ceph

As Ceph is of current interest in the storage community, and because we are basing much of our work on it, we give Ceph some special mention.

Contrast with T10

Ceph's architecture can be contrasted with that of T10 OSDv2. Object storage devices in T10 are independent of each other and under the control of some director that grants access through security tokens. Replication occurs when clients write the same data to several devices, and parity is calculated on the client and written as normal data. How data and parity blocks are distributed among devices is outside the scope of the T10 OSDv2 standard.

Ceph organizes object storage devices into a cooperative group under control of a small number of servers called monitors. In Ceph, monitors keep globally known data coordinated through Paxos. Data is distributed over devices under the control of a globally known collection of rules and data structures, which are maintained consistently by the monitors. Administrators describe the organization of storage devices, breaking them down hierarchically into zones of potential failure. Administrators also set the number of replicas and policies about where to place objects. The placement logic shared between clients and storage devices combines this policy with a monitor-maintained map of storage device status (operating, temporarily down, out of service) to calculate where individual objects are located. Clients perform writes to one object storage device, and the storage devices coordinate between themselves to perform replication and data recovery [9]. Ceph currently lacks support for erasure coding, but multiple efforts are underway to add it.

The present RADOS protocol lacks access control beyond a public key needed to communicate with the cluster at all. There are designs for access control on extremely large scale object storage systems [6].

Immediate Applications

Ceph is a large, complex system currently undergoing active development and gaining new capabilities. There are several use-cases it can address out of the box.

Ceph provides an immediately replicated file system in a single datacenter environment. A stable write to a file is considered to be complete when it has been stably recorded on all devices replicating the given block, giving resilience against drive failure. Per-directory immutable snapshots can be made by unprivileged users allowing them to version their data.

Even without the file system, Ceph can be used in the construction of private clouds that leverage the large number of applications made to work with the S3 protocol. Ceph can be dropped in as a replacement for public cloud services simply by setting up a cluster and configuring applications to use the RADOS Gateway server as the target for requests. Also, Ceph can be used as a proxy server for Swift requests in OpenStack installations, though its Swift interface is less complete than its S3 interface.

RADOS block devices (RBD) have full Linux kernel support, and can be used any way you would use any other block device, but with replication and a snapshot capability. Some people have experimented with re-exporting RADOS block devices over iSCSI to make them available to other operating systems, such as FreeBSD or Windows. RBD's biggest success has been in virtualization environments. In addition to providing an RBD device as a normal block device, RBD support has been integrated into virtualization and cloud computing systems, such as OpenStack, and any systems using libvirt (such as CloudStack and virt-manager).

CohortFS Development

Our goals at CohortFS include development in the field of object storage. Currently, we're focusing on improving the capabilities of Ceph as both an object store and a file system, and on improving NFS to take the best advantage of advanced functionality that a file system provides.

Ceph via NFS

We have added NFS access to the Ceph file system and to the Ganesha user space NFS server, and have implemented pNFS access for objects striped over Ceph OSDs. In the future, we will be developing a new layout type to better take advantage of placement strategies other than repeated striping patterns; we will also be adding support to Ceph for a recallable layout that better matches the requirements of NFS than do current Ceph capabilities.

Volumes

We are adding an implementation of volumes to Ceph, which will allow a single cluster to hold multiple independently rooted file systems or collections of objects, each with its own administrative domains of control to support delegated multi-tenancy. Our future development in this area includes automatic allocation of object-storage devices with different capabilities to fill administrator-specified quality-of-service requirements.

Erasure Coding and Client Offload

We are currently adding erasure coding support to Ceph, using the Jerasure library. In CohortFS, clients will be able to perform replicated writes and generate erasure codes rather than having to leave those to the OSDs. This frees us of the requirement to have multiple OSDs coordinate in a computation for each write, while still allowing OSDs to repair faults automatically. This is in contrast with another erasure coding project for Ceph where erasure codes are generated cooperatively by the OSDs.

Dynamically Generated Placement Functions

We take the notion of a globally known placement function to its logical conclusion by dynamically generating placement functions as fragments of executable code that are distributed throughout a Ceph cluster and to clients. This allows us to tailor data placement specifically to the requirements of the use-case. Additionally, expensive optimization of the function against the cluster description can be performed once, centrally. This gains faster placement calculation without loss of generality. Finally, this allows us to change the behavior of the system radically without having to go through the expensive and error-prone operation of a cluster-wide upgrade.

Availability

Much of the software mentioned here is available with source on the Internet. An implementation of a T10 OSD target is available from http://www.open-osd.org. They also developed an initiator and a scale-out network file system built on top of the T10 OSD protocol called ExoFS. ExoFS and the T10 initiator have been integrated into recent Linux source trees.

Ganesha is a user-space server for versions 3 and 4 of the NFS protocol and for 9P, a file-system protocol originally used for the Plan 9 operating system that is now seeing some use in high-performance computing environments. Ganesha's design is centered around a File System Abstraction Layer, allowing it to serve systems as diverse as the Linux open-by-handle interface, ZFS (through libraries), and even a proxy to other NFS servers. It is used as an NFS front-end for both free and proprietary file systems, and also functions as a platform for development of new server functionality. Ganesha is available from http://nfs-ganesha.github.com.

The Ceph distributed object store and file system is available from http://ceph.com.

OpenStack is a free Infrastructure-as-a-Service framework that implements the Swift object storage service as well as integrating well with other object storage systems, such as S3 and Ceph. OpenStack is available from http://www.openstack.org.

The Jerasure library implements many freely available erasure codes and is available from http://Web.eecs.utk.edu/~plank/plank/papers/CS-08-627.html.

Much of our work is available in the Ganesha NFS server, and some has (or will shortly be) submitted to Ceph. Other parts of CohortFS will become freely available as they are completed.

Acknowledgments

We would like to thank the National Science Foundation, which has funded our work on CohortFS through a Small Business Innovation Research grant, Peter Honeyman for his work in helping to design CohortFS, and the Ceph community for providing a flexible and open platform for development. We would also like to thank the Ganesha community for embodying everything that is good about collaborative, free software development.

References

[1] Amazon.com, "Working with RTMP Distributions," October 2013: http://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/WorkingWithStreamingDistributions.html.

[2] R. T. Fielding, and R. N. Taylor, "Principled Design of the Modern Web Architecture," *ACM Transactions on Internet Technology* 2 (2002), pp. 115-150.

[3] G. A. Gibson, D. F. Nagle, W. I. Courtright, N. Lanza, P. Mazaitis, M. Unangst, and J. Zelenka, "NASD Scalable Storage Systems," *Proceedings of USENIX 1999*, Linux Workshop, Monterey CA, June 9–11, USENIX Association.

[4] R. J. Honicky, "A Fast Algorithm for Online Placement and Reorganization of Replicated Data," *Proceedings of the 17th International Parallel & Distributed Processing Symposium (IPDPS 2003)*.

[5] R. J. Honicky and E. L. Miller, "Replication Under Scalable Hashing: A Family of Algorithms for Scalable Decentralized Data Distribution," *Proceedings of the 18th International Parallel & Distributed Processing Symposium (IPDPS 2004)*.

[6] A. Leung and E. L.Miller, "Scalable Security for Large, High Performance Storage Systems," *Proceedings of the 2nd ACM Workshop on Storage Security and Survivability (StorageSS 2006)* (Alexandria, VA, October 2006), ACM.

[7] J. Plank, "Erasure Codes for Storage Systems: A Brief Primer," USENIX *;login:* (December 2013, Volume 38, Number 6).

[8] S. Shepler, M. Eisler, and D. Noveck, Network File System (NFS) Version 4 Minor Version 1 Protocol, RFC 5661 (Proposed Standard), January 2010.

[9] S. A. Weil, "Ceph: Reliable, Scalable, and High-Performance Distributed Storage," Ph.D. thesis, University of California at Santa Cruz, December 2007.

usenix

THE ADVANCED COMPUTING SYSTEMS ASSOCIATION

Buy the Box Set!

Whether you had to miss a conference, or just didn't make it to all of the sessions, here's your chance to watch (and re-watch) the videos from your favorite USENIX events. Purchase the "Box Set," a USB drive containing the high-resolution videos from the technical sessions. This is perfect for folks on the go or those without consistent Internet access.

Box Sets are available for:

- **LISA '13:** 27th Large Installation System Administration Conference
- **USENIX Security '13:** 22nd USENIX Security Symposium
- **HealthTech '13:** 2013 USENIX Workshop on Health Information Technologies
- **WOOT '13:** 7th USENIX Workshop on Offensive Technologies
- **UCMS '13:** 2013 USENIX Configuration Mangement Summit
- **HotStorage '13:** 5th USENIX Workshop on Hot Topics in Storage and File Systems
- **HotCloud '13:** 5th USENIX Workshop on Hot Topics in Cloud Computing
- **WiAC '13:** 2013 USENIX Women in Advanced Computing Summit
- **NSDI '13:** 10th USENIX Symposium on Networked Systems Design and Implementation
- **FAST '13:** 11th USENIX Conference on File and Storage Technologies
- **LISA '12:** 26th Large Installation System Administration Conference

Learn more at:
www.usenix.org/boxsets