

GPU Erasure Coding for Campaign Storage

Walker Haddock¹, Matthew L. Curry², Purushotham V. Bangalore¹, and Anthony Skjellum³

¹ Dept. of Computer and Information Sciences
University of Alabama at Birmingham

² Center for Computing Research
Sandia National Laboratories

³ Dept. of Computer Science and Engineering &
McCrary Institute for Critical Infrastructure Protection and Cyber Systems
Auburn University

Abstract. High-performance computing (HPC) demands high bandwidth and low latency in I/O performance leading to the development of storage systems and I/O software components that strive to provide greater and greater performance. However, capital and energy budgets along with increasing storage capacity requirements have motivated the search for lower cost, large storage systems for HPC. With Burst Buffer technology increasing the bandwidth and reducing the latency for I/O between the compute and storage systems, the back-end storage bandwidth and latency requirements can be reduced, especially underneath an adequately sized modern parallel file system. Cloud computing has led to the development of large, low-cost storage solutions where design has focused on high capacity, availability, and low energy consumption at lowest cost. Cloud computing storage systems leverage duplicates and erasure coding technology to provide high availability at much lower cost than traditional HPC storage systems. Leveraging certain cloud storage infrastructure and concepts in HPC would be valuable economically in terms of cost-effective performance for certain storage tiers. To enable the use of cloud storage technologies for HPC we study the architecture for interfacing cloud storage between the HPC parallel file systems and the archive storage. In this paper, we report our comparison of two erasure coding implementations for the Ceph file system. We compare measurements of various degrees of sharding⁴ that are relevant for HPC applications. We show that the Gibraltar GPU Erasure coding library outperforms a CPU implementation of an erasure coding plugin for the Ceph object storage system, opening the potential for new ways to architect such storage systems based on Ceph.

⁴ The literature uses the term “stripe” for a set of data that is protected by RAID or erasure coding implementation. The stripe is divided into k data chunks and protected by m parity or coding chunks. In this paper, the term “strip” and “shard” are used synonymously and refer to these chunks.

1 Introduction

With the compute core density increasing per node in the past decade in high-performance computing (HPC), a trend that will likely continue for the next decade, I/O bandwidth requirements per node have also increased. This increase in computing power is applying pressure on the entire storage capacity and bandwidth for HPC systems. To minimize the time required for applications to complete I/O operations for initialization, checkpoint/restart (CR), and application result outputs, it is necessary to provide I/O bandwidth to the compute nodes that is much higher than today’s petascale supercomputers. The stated requirements for the exascale initiative is for applications to run 50 times faster than they do on today’s 20 PFLOP systems [21]. Los Alamos National Labs (LANL) is introducing Burst Buffers (BB) as an intermediate tier between the compute nodes and the Parallel File System (PFS) on Trinity. These BBs use Solid State Disks (SSDs) and Nonvolatile RAM (NVRAM) to provide high-speed storage to meet the faster IOPS and bandwidth requirements. BBs enable the applications to complete the IO operations in an acceptable time for CR or application recording task and get back to making forward progress on the application problem solution [12].

Both the BB and PFS file systems are expensive and, like registers and cache in the CPU memory hierarchy, are at the top of the storage pyramid, having minimal size but much greater performance [12]. Requirements also provide the constraints that the life time of the data in the compute node RAM is hours, the life time in the BBs is hours, and the life time in the PFS is weeks. The BBs are referred to as tier-1 in the storage pyramid and the PFSs are referred to as tier-2. Data that is needed to be kept for longer periods of time may be stored on lower tiers of the storage pyramid where lower latency and bandwidth requirements may be defined but with greater availability requirements due to the longer life of the data residence. The third layer of the storage stack, tier-3 storage is referred to as the “campaign” storage layer, a pre-archive, longer term, and higher capacity disk store [11]. This campaign storage layer is a strong candidate for lower-cost, cloud-type storage that provides availability with erasure coding and higher bandwidth than the fourth tier, the archive tier, which is magnetic tape [16] at LANL. The Campaign storage tier has been designed to store data for a period of time while the research project is actively computing so that it can be quickly moved to the PFS and BB when needed for computation or to the archive for longer term storage.

The key contribution of this paper is as follows. To enable the use of cloud storage technologies for HPC, we study the architecture for interfacing cloud storage between the HPC parallel file systems and the archive storage. In this paper we show that computing erasure coding for a high degree of sharding on the Ceph Object File System [38] with GPUs outperforms a modern Intel CPU, opening the potential for new ways to architect such storage system based on Ceph. For use cases where data are moved to object storage systems via single points of mediation, such as the File Transfer Appliances (FTAs) in the LANL Trinity system, these mediators may be equipped with GPUs to perform erasure

encoding and recovery at high speed and utilization. High degrees of sharding along with sufficient coding shards determined by the disk failure rate can result in lower capital costs and lower operating costs [13].

The remainder of this paper is organized as follows. We first discuss the architecture and performance of Ceph, a high performance distributed storage systems [38], particularly the plugin feature for erasure coding modules, review RAID and discuss the exascale campaign storage requirements for Trinity (see section 2). We also discuss the Gibraltar GPU erasure coding and decoding library [4] there. In section 3, we discuss our implementation of the Ceph erasure coding plugin using Gibraltar. We present our findings from our measurements of our experiments in section 4. In section 5 we discuss other research using GPU erasure coding for HPC storage followed by our conclusions in section 6.

2 Background

We utilize Ceph as a platform for our work because of its convenient plugin architecture for erasure coding libraries. This structure enabled us to focus our work on the Gibraltar library and to follow the implementation of the Ceph plugin interfaces provided in its erasure code plugin classes. The existing erasure coding plugins in Ceph provide us with well know baselines against which to compare our results.

In previous work, we designed and reduced to practice a library that performs erasure coding on GPU hardware, Gibraltar [3, 4]; this approach can further lower the price/performance for storage systems and provide opportunities for performing compute close to the data. One of the consequences of erasure coding in the design of high performance distributed file systems is the high computational and data transfer costs of reconstruction of a failed disk. By including GPUs in the architecture, we provide additional compute resources that can raise the achievable performance. As common disk drive storage capacities have increased from 750 GB in 2006 [9] to 10 TB in 2016 [34], this architecture performance enhancement will become even more important by off-loading computation for erasure coding to the GPU.

2.1 Ceph

Ceph is a distributed high performance file system that decouples metadata from data and provides a deterministic function for mapping metadata to data location, CRUSH – Controlled Replication Under Scalable Hashing [37]. It is an object storage system that uses peer-to-peer sharing of a compact hierarchical description of the cluster configuration and replication policy. This innovation distributes the computation to determine replica placement to any member of the cluster, including clients, thus eliminating the serialization that would otherwise result from determining data placement on a centralized metadata service. The CRUSH algorithm uses rule sets to define policies on data placement that result in evenly distributed storage of data across all of the Object Storage Devices

(OSDs) in the cluster. These rules also enforce availability policies; for example, replicas must not be in the same rack or other defined failure domain in the data center. Ceph implements the data storage layer of file systems with the library *librados*, which exposes an interface to the Ceph object store. Traditional block based file systems can access the Ceph cluster object storage via the *RADOS Block Device*, a driver for Linux kernels based on *librbd* [39]. The Ceph POSIX file system (CephFS) uses the Metadata Service (MDS), which provides the POSIX compatible file name space features as well as the management of atomicity for operations (file creation, file deletion, file renaming, attribute changes, permissions, locks, etc). CephFS consults the MDS to provide the client with the layout of a given file upon which operations are being performed.

In 2013, the Ceph community implemented a plugin framework to provide erasure coding features [8]. The Ceph development team used the framework to implement a concrete erasure coding capability using the Jerasure library [17]. The plugin includes an *EraseCode*, *EraseCodeInterface*, *EraseCodePlugin*, and *EraseCodePluginRegistry* classes. Implementers of concrete plugins can follow the example of the Jerasure plugin module in order to wrap their own erasure coding library into Ceph. The mechanism is activated by Ceph pools, which are configured to use replication or erasure coding with specific parameters. Erasure coding provides for various configuration selections based on the concrete implementation to include the algorithm, number of data shards, k , that object stripes will be divided into and the number of equally sized coding shards, m , that will be used to store the objects [7]. Choosing between replication or erasure coding for reliability trades space for computation. Choosing higher degrees of sharding distributes the object stripes over a greater number of disks which reduces the time required to put or get the data on the disk by increasing parallelism. The disk read or write time for an object stripe is inversely proportional to the degree of sharding because the size of the shards are inversely proportional to the degree of sharding (k). Erasure coding can survive the loss of up to m shards. Where replication consumes raw storage at the rate of n times the size of the object where n is equal to the number of replicas + 1, the proportion of space used by k shards is usually about 20% of the size of the data (which can survive the loss of 1 shard out of 5) [32, 36].

2.2 RAID

Since the redundant array of inexpensive disks/devices (RAID) was introduced in 1988 by Patterson, Gibson and Katz [27] that provided an economical way for systems to be more resilient against data loss compared to other options such as pure mirroring, research has continued to provide more techniques for improving availability of data and improving performance. The principle methods for mitigating the loss of data resulting from media or system failure has been replication, RAID and erasure coding. The design choices between these methods must be balanced between the higher cost of storage for replication of $n \times r$ where n is the size of the data and r is the number of replicas plus one versus the computational cost of parity generation for RAID and erasure coding.

Erasure coding provides a higher degree of durability in that the storage system can survive the loss of a greater number of disks while using less additional storage than replication [32, 36]. The property that erasure coding can provide a higher order of redundancy by generating more than two parity disks has been heavily studied by James Plank [17, 28, 29, 30]. Another consideration for data reliability is locality. Storage subsystems that replicate data or store parity on direct attached media can provide data storage services incurring a lower communications cost as compared to storage systems that distribute replicas or parity throughout a set of storage nodes that are connected over a high speed network. This particularly is the case of reconstructing parity for RAID 5, RAID 6 as compared to erasure coding where the minimum set of data or coding shards must be copied over the network and be assembled in a contiguous memory location for the erasure coding program to recompute the missing data or coding. After the data are reconstructed, the repaired shards must be copied back to their storage locations over the network. There is strong evidence that using erasure coding with commodity hardware for durability in high performance computing is more economical and faster than dedicated storage subsystems [31, 33]. For instance, Microsoft has chosen to implement the storage systems in their Azure cloud service using erasure coding [15]. A thorough treatment of performance measurement for erasure coding is given in [14]. The power efficiency of erasure coding has been discussed previously by Greenan [10]. Lastly, DACO proposes a scheme where remote code is executed by disk drive controllers to update parity directly on the media saving on the data transfer costs that are usually associated with updates to erasure-coded stripes [20].

Storage services for high performance computing systems can be provided by storage area networks (SANs), which provide data resilience and high speed communications over specialized networks. Some high performance distributed file systems rely on these types of storage providers where the responsibility for data reliability is handled by the SAN [1, 2]. These file systems can also be configured to provide availability in the event of the loss of data serving nodes by providing multi-path connections to the SAN storage. The SAN subsystems present storage volumes to the storage servers in the form of LUNs; these are logical volumes of media blocks formed by the SAN subsystem that have the resilience properties that have been specified by the administrator, providing the semantics otherwise of a local disk volume.

The Gibraltar project demonstrated that erasure codes could be efficiently generated and decoded with GPUs [3, 4, 5, 6]. The Gibraltar library was designed to compute Reed-Solomon erasure codes for a wide range of k data shards and m coding shards. We have used this library to provide GPU-assisted erasure coding for Ceph through Ceph's Erasure Coding Plugin subsystem.

2.3 System Requirements

The Los Alamos National Laboratory (LANL) has presented requirements for a Campaign Storage System for the Trinity Super Computer[19]. The Campaign Storage should have about 25 PB capacity with future expansion capability. The

bandwidth should be between 20 to 25 GB/s, which should increase with capacity. The files stored in the campaign storage system will not be updated in place. The system should use archive-grade hard disk drives, and gain performance through large scale parallel access. The system should use erasure coding for reliability. The system is not intended for high duty cycle workloads [19]. LANL expects to have 20 to 25 batch file transfer agents (FTAs) to move data between user home storage, Lustre PFS systems, archive storage and the campaign storage [19]. These requirements imply that the FTAs will be able to move about 1 GB/s each not including enough additional performance to provide fault tolerance.

LANL has also indicated the needed capability to store 1 PB sized checkpoints in the near future [19]. Baselineing with archive storage disk drives with a capacity of 8 TB, it would require a minimum of 128 disk drives plus about 20% more for the erasure coding overhead to store 1 PB. Given this capacity requirement, a reasonable approach would be to use 128 data shards to distribute the 1 PB file over this number of disk drives. Choosing a ratio of 1 coding shard to 5 data shards would require another 25 disk drives. Using 8 TB disk drives, the 25 PB campaign storage system would therefore contain a minimum of 3,825 drives. The big advantage to this large degree of sharding is the lower bandwidth requirement to each of the target disk drives, 8 MB/s in this example where there are 128 shards and the FTA is delivering 1 GB/s to the storage system, Eqn. $\frac{1GB}{second}$. In this case, the bandwidth requirement at the leaf OSD can be met with a 100 Mb/s network and is well under the 150MB/s peak performance for the current archive type 8 TB disks.

$$\left(\frac{\frac{1GB}{second}}{128shards} \right) \quad (1)$$

3 Ceph Erasure Coding Plugin Implementation for Gibraltar

Ceph provides a well-defined interface for integrating erasure coding libraries into the product. This mechanism provides a means to incorporate new erasure coding libraries into the Ceph file system. The plugin architecture is modularized into two functional areas: registration of erasure coding profiles and the interface for the erasure coding/decoding services of the library [7]. Gibraltar can theoretically provide up to 256 data and coding shards in a stripe [4], although practical limits currently restrict $k+m$ to fewer total shards (see section 4.2). Ceph erasure coded profiles can be constructed with many combinations of k and m .

The Ceph ErasureCodePlugin class is subclassed in our work in order to instantiate a Gibraltar instance; this instance is configured according to the parameters provided by the Ceph administrator command to create an erasure code profile. Gibraltar uses the NVIDIA[®] CUDA[®] library [25] to offload computation and retrieve results from the K40 GPU [24] in our system. The subclass

ErasureCodePluginGibraltar calls the Gibraltar `gib_cuda_driver` function to initialize a CUDA context for the profile. The profile can then be used to create an erasure coded pool in Ceph.

The Ceph ErasureCode class is subclassed in our work to implement the Ceph ErasureCodeInterface functions for the Gibraltar library. We modified the Gibraltar erasure code library application programmer interface (API) to make it compatible with the Ceph architecture. Ceph uses a bufferlist data structure and aggregates $k + m$ shards for each erasure coded stripe where each shard is referenced by a pointer to the head of the list data structure for the shard. The call to Gibraltar has been modified to provide an array of these pointers and the logic copies each shard of data onto a contiguously allocated GPU memory block. The outputs of the coding or decoding are copied back to the Ceph bufferlist data structures in a similar way. In figure 1, we show how data is passed to the plugin. The plugin appends m coding shards onto the Bufferlist object and includes the pointers to these data structures in the array. New versions of the Gibraltar functions to encode and regenerate were created, whereas the original functions operated on a contiguous data block that was passed in the call. The original library interface for Gibraltar had proved sufficient for the target RAID system when originally designed, and was chosen to reduce register pressure in the GPU by reducing the number of variables [3].

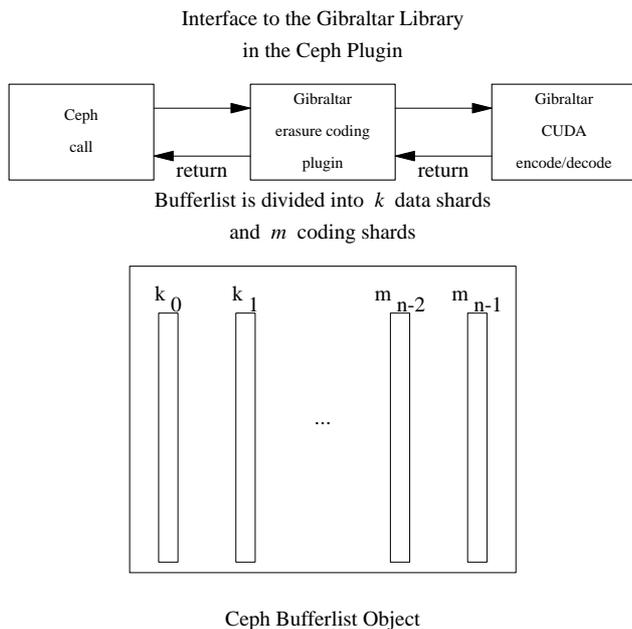


Fig. 1. Ceph calls the erasure coding module with a Bufferlist object containing the stripe to be written to the object. The Plugin divides the Bufferlist into k data shards and adds m coding shards. Gibraltar is called to perform the coding or recovery.

4 Evaluation and Measurement

Here, we discuss the configuration of the system used to perform the experiments and then we show the results that we obtained.

4.1 System Description

We conducted the measurements on a Dell R730 server with a GPU. Table 1 lists the configuration of our test system.

Table 1. Dell R730 with GPU Configuration

| | |
|---------------|--|
| CPU | 2x Xeon E5-2650 v3 @ 2.3 GHz (HT-enabled: 40 threads) |
| RAM | 128 GB 2133 MT/s RDIMM |
| Network | 2 port Mellanox ConnectX-3 MCX354A-FCBS Intel X520 DP 10Gb DA/SFP+, I350 DP 1Gb Ethernet |
| GPU | NVIDIA [®] K40m GPU |
| System Drives | 2x 300 GB 10K SAS 2 2x 200 GB INTEL SSDSC2BG20 SATA 2x 400 GB TOSHIBA PX02SMF040 SAS 3 |

4.2 Erasure Code Generation and Reconstruction Performance

Four experiments were conducted to understand the benefits of GPU erasure coding for the campaign storage application. The first two experiments were run using the ISA-L [35] erasure coding plugin included in Ceph v12.0.0 to provide a baseline for reference to our Gibraltar plugin performance. The Cauchy algorithm was selected for ISA-L because it was able to perform the encoding and decoding with the number of shards we needed to test while the ISA-L Vandermonde matrix implementation was limited to a maximum of 32 data shards in Ceph in order to guarantee an MDS codec (see ErasureCodeIsa.cc). The Reed-Solomon algorithm with the Vandermonde matrix is well suited to computation on a GPU using a precomputed lookup table which has been implemented in the Gibraltar library. We selected a range of degrees of sharding between 20 and 128 based on experiments that are being conducted for Trinity campaign storage at LANL. Coding and decoding tests for 1 GB data size used degrees of sharding ranging between 50 and 128. Erasure decoding for Gibraltar is currently limited to 118 shards. We selected the number of coding shards for each data sharding choice to meet the 1 to 5 ratio. Test data set sizes of 512 MB and 1 GB were used for the experiments. We used the erasure coding benchmark tool included with v12.0.0 of Ceph to run the test cases. The benchmark tool instantiates an erasure coding profile as specified in our execution parameters and then runs a series of encoding generations and reconstructions over a set of data. We set

the CPU core affinity to use a single core on the first CPU in the system because the NVIDIA[®] GPU is connected to the PCI bus of this CPU socket. In each experiment, the same CPU core was used for each erasure coding library. A set of 10 iteration runs were made totaling 10 GB of data for each run and the average reported. The results in figure 2(a) show that the Gibraltar library generates parity at 5 times the rate of ISA-L in the 50+10 test and at 7 times ISA-L performance for 128+24 test. The ISA-L method was stable between the 512 MB and 1 GB data sizes but the Gibraltar method was 5% faster for the 1 GB data size with 128 shards than the corresponding 512 MB test.

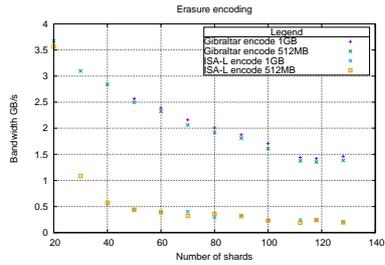
In figures 2(b) and 2(c) we show the performance of reconstructing erasures for our second experiment. Times are shown for reconstructing 1 and 4 erasures in the configurations tested. At the 1 GB test data size, Gibraltar performance is 43% better than the ISA-L's performance with the 50+10 test with 1 erasure and is 18% faster than ISA-L's 118+24 with 1 erasures. At the 1 GB test data size, Gibraltar performance is 3.77 times faster than the ISA-L performance for 50+10 test with 4 erasures and is 2.98 times the performance of ISA-L with 4 erasures at the 118+24 test data size.

Figure 2(d) shows the shard sizes for the 512 MB and 1 GB data sizes that we used in the experiments. A larger degree of sharding results in smaller chunks of the data, lowering the bandwidth requirements to copy the shard to the OSD.

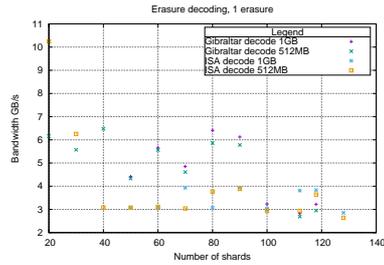
The third experiment measured the Gibraltar encoding execution using the CUDA nvprof program. The same test configurations were used as in the first two experiments. We ran 10 iterations with 1 GB test data. For the 128+24 configuration we measured 25% of the time was spent copying the data to the GPU memory, 70% of the time was spent generating the erasure coding shards and the remaining 5% of the time was spent copying the coded shards back to the host.

The fourth experiment measured the reconstruction using the 118+24 configuration and 4 erasures. In the Gibraltar reconstruction, the Galois inversion matrix is computed on the host and copied to the GPU for the specific erasures that are present in the data. The Galois computation consumed 15% of the time. We measured 55% of the time was spent copying the data to the GPU memory, 23% of the time was spent reconstructing the data and parity, and 4% of the time was spent copying the 4 reconstructed data shards from the GPU memory to the host.

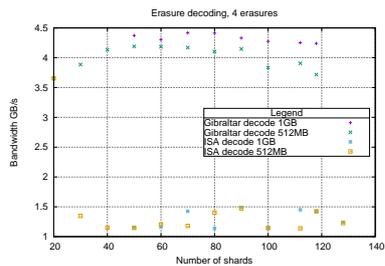
These experiments show that the Gibraltar GPU library can sustain a high bandwidth performance with larger degrees of sharding. For recovery, the Gibraltar GPU library can recover multiple erasures without loss of performance as compared with the ISA-L library. Using higher performance GPUs like the NVIDIA[®] Pascal[23] with NVLink[®] should provide even greater performance where more bandwidth is required.



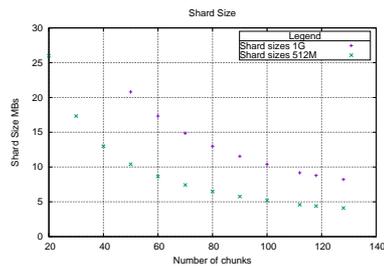
(a) Erasure coding bandwidth results with increasing number of shards. Coding shards are held to a ratio of 1 coding shard to 5 data shards.



(b) Erasure recovery bandwidth results with increasing number of shards and 1 erasure. Coding shards are held to a ratio of 1 coding shard to 5 data shards.



(c) Erasure recovery bandwidth results with increasing number of shards and 4 erasures. Coding shards are held to a ratio of 1 coding shard to 5 data shards.



(d) Shard sizes used in the erasure coding and decoding measurements.

Fig. 2. Erasure encoding and reconstruction comparisons between ISA-L and Gibraltar.

5 Previous Work

Ceph provides an erasure coding plugin class as an extensible way for erasure coding libraries to be implemented in the product. Currently, in release 12.0.0, there are four erasure coding libraries implemented, namely: Jerasure [17], ISA-L [35], lrc [26] and Shingled Erasure Code (SHEC) [22]. Our study has shown that the Gibraltar library [5] can perform about half as well as these libraries for a sharding degree less than 40 while Gibraltar performs better than these for greater degrees of sharding. Khasymski et al., showed that GPU-assisted erasure coding and reconstruction can be performed on the Lustre file system. Their work required a software shim to integrate into Lustre which has no defined interface for erasure coding libraries. Their work only implemented RAID 6, providing $m=2$, but showed that the approach was feasible and can provide strong fault tolerance [18] without depending on RAID subsystems and failover mechanisms for availability.

6 Conclusion

We have implemented the Gibraltar GPU erasure coding library[4] as a plugin in the Ceph product and shown that it provides high bandwidth for large degrees of sharding. This capability can increase the value of cloud storage technologies in HPC by increasing the number of coding shards to provide extended availability and reducing the need for recovery of failed members. In these experiments, the Gibraltar library performance proved much greater than ISA-L [35] on the Intel E5-2650 v3 CPU. To achieve ISA-L performance measured here, it was necessary to dedicate a single core of the host computer whereas the Gibraltar library required 70% of the NVIDIA[®] K40 GPU's total capacity for generation and 23% of the NVIDIA[®] K40 GPU for reconstruction of 4shards.

Our measurements show that the Ceph Gibraltar plugin can generate over 120 shards at nearly 1.5GB/s while the ISA-L plugin has dropped to less than 250MB/s. The Ceph Gibraltar plugin continues to reconstruct erasures at over 4GB/s for 1 or 4 erasures with 118 shards while the ISA-L Ceph plugin drops from 3GB/s for 1 erasure to 1.5GB/s for 4 erasures. Ceph with the Gibraltar plugin can meet or exceed the current requirements for Trinity campaign storage with respect to erasure coding and reconstruction.

7 Future Work

The architecture for campaign storage at LANL moves all data between the campaign storage system and the other storage systems via File Transfer Appliances (FTAs) [19]. Generation of erasure codes requires data locality and is ideally performed on the file transfer appliance (FTA). The FTAs having GPU accelerator devices can generate coding and reconstruct shards efficiently where there is a high degree of sharding. Gibraltar can provide greater than $m=2$ parity to support higher fault tolerance which is needed for larger capacity disk

drives. Using the FTAs to perform erasure coding and reconstruction in a Ceph implementation will require modification of the current interfaces and concepts for the Ceph erasure coding plugin architecture. The current implementation of erasure coding in Ceph only provides for OSDs to perform erasure coding and reconstruction.

A higher degree of sharding spreads the volume of data over a correspondingly large number of disk drives. The input bandwidth to the FTA is fanned out by the sharding degree resulting in the opportunity to use lower bandwidth communications for the Object Storage Nodes without reducing throughput performance of the FTA. This provides an opportunity for reducing the cost of campaign storage. We are currently studying the configuration options of campaign storage with regard to sharding degree, throughput, reliability and cost in greater depth.

8 Acknowledgments

This material is based upon work supported by the National Science Foundation under Grants Nos. ACI-1541310, CNS-0821497 and CNS-1229282. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

This work has also been supported by Sandia National Laboratories is a multi-mission laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

References

- [1] Braam, P.J., Schwan, P.: Lustre: The intergalactic file system. In: Ottawa Linux Symposium. p. 50 (2002)
- [2] Corbett, P.F., Feitelson, D.G., Prost, J.P., Almasi, G.S., Baylor, S.J., Bolmarcich, A.S., Hsu, Y., Satran, J., Snir, M., Colao, R., Herr, B.D., Kavaky, J., Morgan, T.R., Zlotek, A.: Parallel file systems for the IBM SP computers. *IBM Systems Journal* 34(2), 222–248 (1995), <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5387272>
- [3] Curry, M.L., Skjellum, A., Lee Ward, H., Brightwell, R.: Accelerating Reed-Solomon coding in RAID systems with GPUs. In: Proceedings of the 2008 IEEE international parallel & distributed processing symposium. pp. 1–6. IEEE, Miami, FL, USA (Apr 2008), <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4536322>
- [4] Curry, M.L., Skjellum, A., Lee Ward, H., Brightwell, R.: Gibraltar: A Reed-Solomon coding library for storage applications on programmable graphics processors. *Concurrency and Computation: Practice and Experience* 23(18), 2477–2495 (Dec 2011), <http://doi.wiley.com/10.1002/cpe.1810>

- [5] Curry, M.L., Ward, H.L., Skjellum, A., Brightwell, R.: A Lightweight, GPU-Based Software RAID System. pp. 565–572. IEEE, San Diego, CA, USA (Sep 2010), <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5599249>
- [6] Curry, Matthew L.: A highly reliable GPU-based RAID system. Ph.D. thesis, University of Alabama at Birmingham (2010), <http://contentdm.mhsl.uab.edu/cdm/ref/collection/etd/id/854>
- [7] Dachary, L.: Ceph Replication vs Erasure Coding (Jul 2013), <http://dachary.org/?p=2171>
- [8] Dachary, Loic, Samuel Just: Erasure Code (Aug 2013), https://github.com/dachary/ceph/blob/wip-4929/doc/dev/osd_internals/erasure-code.rst
- [9] Farrance, R.: Timeline: 50 Years of Hard Drives (Sep 2006), <http://www.pcworld.com/article/127105/article.html>
- [10] Grenan, K.: Reliability and Power-Efficiency in Erasure-Coded Storage Systems. Tech. Rep. UCSC-SSRC-09-08, University of California, Santa Cruz (Dec 2009)
- [11] Grider, G.: MarFS (May 2015), <http://storageconference.us/2015/Presentations/Grider.pdf>
- [12] Grider, Gary: HPC Storage and IO Trends and Workflows (Apr 2016), <http://salishan.ahsc-nm.org/program.html>
- [13] Haddock, Walker, Matthew L. Curry, Puri Bangalore, Anthony Skjellum: Using GPU Erasure coding to lower HPC Pre-Archive Storage Costs (to appear)
- [14] Hafner, J.L., Deenadhayalan, V., Kanungo, T., Rao, K.: Performance metrics for erasure codes in storage systems. IBM Res. Rep. RJ 10321 (2004)
- [15] Huang, C., Simitci, H., Xu, Y., Ogus, A., Calder, B., Gopalan, P., Li, J., Yekhanin, S.: Erasure Coding in Windows Azure Storage. In: Usenix annual technical conference. pp. 15–26. Boston, MA (2012)
- [16] Inman, J., Grider, G., Chen, H.B.: Cost of Tape versus Disk for Archival Storage. pp. 208–215. IEEE, Anchorage, AK, USA (Jun 2014), <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6973743>
- [17] James S. Plank, Scott Simmerman, Catherine D. Schuman: Jerasure: A Library in C/C++ Facilitating Erasure Coding for Storage Applications. Tech. Rep. Technical Report CS-08-627, University of Tennessee, Knoxville, TN 37996 (2008), <http://www.cs.utk.edu/plank/plank/papers/CS-08-627.html>
- [18] Khasymski, A., Rafique, M.M., Butt, A.R., Vazhkudai, S.S., Nikolopoulos, D.S.: On the Use of GPUs in Realizing Cost-Effective Distributed RAID. pp. 469–478. IEEE, Washington, DC, USA (Aug 2012), <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6298207>
- [19] Lamb, Kyle: Trinity Campaign Storage and Usage Model (Aug 2015), https://www.lanl.gov/projects/trinity/_assets/docs/trinity-usage-model-presentation.pdf
- [20] Li, M., Shu, J.: DACO: A high-performance disk architecture designed specially for large-scale erasure-coded storage systems. Computers, IEEE Transactions on 59(10), 1350–1362 (2010)
- [21] Messina, Paul: A Path to Capable Exascale Computing (Jul 2016), <http://press3.mcs.anl.gov/atpesc/files/2016/07/MessinaJul31.dinner.pdf>
- [22] Miyamae, T., Nakao, T., Shiozawa, K.: Erasure Code with Shingled Local Parity Groups for Efficient Recovery from Multiple Disk Failures. In: 10th Workshop on Hot Topics in System Dependability (HotDep 14). USENIX Association, Broomfield, CO (Oct 2014), <https://www.usenix.org/conference/hotdep14/workshop-program/presentation/miyamae>
- [23] NVIDIA: NVIDIA Tesla P100, images.nvidia.com/content/pdf/tesla/whitepaper/pascal-architecture-whitepaper-v1.2.pdf

- [24] NVIDIA: Tesla K40 GPU Active Accelerator (Nov 2013), https://www.nvidia.com/content/PDF/kepler/Tesla-K40-Active-Board-Spec-BD-06949-001_v03.pdf
- [25] NVIDIA: CUDA Parallel Computing Platform (Mar 2017), http://www.nvidia.com/object/cuda_home_new.html
- [26] Papailiopoulos, D.S., Dimakis, A.G.: Locally repairable codes. *IEEE Transactions on Information Theory* 60(10), 5843–5855 (2014)
- [27] Patterson, D.A., Gibson, G., Katz, R.H.: A Case for Redundant Arrays of Inexpensive Disks (RAID). In: *Proceedings of the 1988 ACM SIGMOD International Conference on Management of Data*. pp. 109–116. SIGMOD '88, ACM, New York, NY, USA (1988), <http://doi.acm.org/10.1145/50202.50214>
- [28] Plank, J.S.: A Tutorial on Reed-Solomon Coding for Fault-Tolerance in RAID-like Systems. *Software Practice & Experience* 27(9), 995–1012 (Sep 1997)
- [29] Plank, J.S., Blaum, M., Hafner, J.L.: SD codes: erasure codes designed for how storage systems really fail. In: *FAST-2013: 11th Usenix Conference on File and Storage Technologies*. pp. 95–104 (2013)
- [30] Plank, J.S., Thomason, M.G.: A practical analysis of low-density parity-check erasure codes for wide-area storage applications. In: *Dependable Systems and Networks, 2004 International Conference on*. pp. 115–124. IEEE (2004)
- [31] Rashmi, K., Shah, N.B., Gu, D., Kuang, H., Borthakur, D., Ramchandran, K.: A "hitchhiker's" guide to fast and efficient data reconstruction in erasure-coded data centers. pp. 331–342. ACM Press (2014), <http://dl.acm.org/citation.cfm?doid=2619239.2626325>
- [32] Rodrigues, R., Liskov, B.: High availability in DHTs: Erasure coding vs. replication. In: *Peer-to-Peer Systems IV*, pp. 226–239. Springer (2005)
- [33] Saito, Y., Frlund, S., Veitch, A., Merchant, A., Spence, S.: FAB: building distributed enterprise disk arrays from commodity components. p. 48. ACM Press (2004), <http://portal.acm.org/citation.cfm?doid=1024393.1024400>
- [34] Shilov, A.: Seagate Unveils 10 TB Helium Filled Hard Disk Drive (Jan 2016), <http://www.anandtech.com/show/9955/seagate-unveils-10-tb-heliumfilled-hard-disk-drive>
- [35] Tucker, G.: ISA-L open source v2.14 API doc (Apr 2014), https://01.org/sites/default/files/documentation/isa-l-open_src_2.10.pdf
- [36] Weatherspoon, H., Kubiatowicz, J.D.: Erasure coding vs. replication: A quantitative comparison. In: *Peer-to-Peer Systems*, pp. 328–337. Springer (2002), 10.1007/3-540-45748-8_31
- [37] Weil, S., Brandt, S., Miller, E., Maltzahn, C.: CRUSH: Controlled, Scalable, Decentralized Placement of Replicated Data. pp. 31–31. IEEE, Tampa, FL, USA (Nov 2006), <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4090205>
- [38] Weil, S.A., Brandt, S.A., Miller, E.L., Long, D.D., Maltzahn, C.: Ceph: A scalable, high-performance distributed file system. In: *Proceedings of the 7th symposium on Operating systems design and implementation*. pp. 307–320. USENIX Association (2006)
- [39] Weil, S.A., Leung, A.W., Brandt, S.A., Maltzahn, C.: RADOS: a scalable, reliable storage service for petabyte-scale storage clusters. In: *In Proceedings of the 2nd international workshop on Petascale data storage: held in conjunction with Supercomputing '07 (PDSW '07)*. p. 35. ACM Press, Reno, Nevada (2007), <http://portal.acm.org/citation.cfm?doid=1374596.1374606>