Lightweight KV-based Distributed Store for Datacenters

Chanwoo Chung, Jinhyung Koo[†], Arvind, and Sungjin Lee[*]

Massachusetts Institute of Technology (MIT) [†]Inha University
[*]Daegu Gyeongbuk Institute of Science & Technology (DGIST)

A great deal of digital data is generated every day by content providers, end-users, and even IoT sensors. This data is stored in and managed by thousands of distributed storage nodes, each comprised of a power-hungry x86 Xeon server with a huge amount of DRAM and an array of HDDs or SSDs grouped by RAID [2]. Such clusters take up a large amount of space in datacenters and require a lot of electricity and cooling facilities. Therefore, packing as much data as possible into a smaller datacenter space and managing it in an energy- and performance-efficient manner can result in enormous savings.

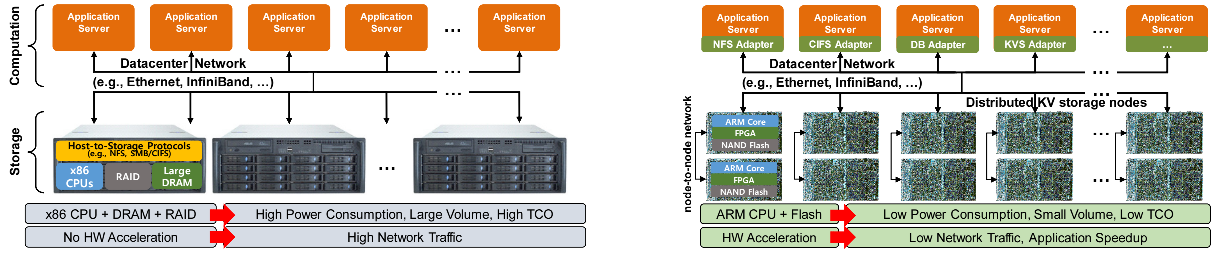
Existing storage nodes have to run complex software to manage distributed and local file systems, and support various host-to-storage protocols, such as NFS, SMB/CIFS, and RADOS. Data reduction techniques like deduplication and compression are often implemented in storage nodes. We argue that the conventional distributed storage architecture with such nodes is over-designed with excessive hardware resources and unnecessarily complex software.

We believe that the hardware and software of existing storage systems can be refactored to run on a lightweight storage node comprised of low-power ARM cores, FPGAs, and raw NAND flash chips. Such a node can be a drive-sized embedded system, which can directly interact with application servers over a datacenter network such as Ethernet. A single server with *N* drives in the conventional system can be replaced with *N* new drive-sized nodes. For example, a flash storage solution composed of a Xeon server and 25 SSDs requires 6 U of rack space and 800 W of power, while the same capacity is achieved with 25 proposed nodes that require only 2 U and 250 W, resulting in tremendous savings in power, space, and a total cost of ownership. Both the proposed and conventional architectures are shown below. The cost-effective performance of such a system is achieved by hardware accelerators and software optimization.

For software, we propose to use a key-value store (KVS) based on LSM-trees because KVS's simplicity makes it possible to run it on low-power cores. KVSs are widely deployed in datacenters and have a popular server-client protocol [6, 5]. In addition, the flexibility of KVSs allows us to emulate existing data stores, such as file systems and DBMSs, in a virtualized manner – protocol adapters implemented in application servers may translate file or database I/Os to a set of KVS operations. In this way, the necessity of running various host-to-storage protocols can be removed from storage nodes. Moreover, LSM-tree engines are flash-friendly because of append-only writes [4]. We can optimize and simplify existing implementations of flash management with little modification, reducing overhead dramatically.

We propose to augment the computing power of the embedded cores with an FPGA between the processor and raw flash chips, and use it to implement a flash chip controller, a node-to-node network controller, and various hardware accelerators for deduplication, compression, and even application logics [1]. The accelerators preprocess data sent to application servers and thus effectively reduce datacenter network traffic and latency. Additionally, a separate node-to-node network can be used to scale the capacity of the storage nodes without RAID and reduce the number of nodes that directly connect to the datacenter network. The simplified flash management duties may migrate from software to hardware, which further enables us to process I/Os quickly.

The proposed nodes can also serve as tools for big data analytics like graph processing with in-store computing capability from hardware accelerators on FPGAs coupled with flash [3].



(a) Conventional x86-based storage nodes

(b) ARM-based distributed KV storage nodes

Figure: A comparison of the conventional distributed store and the proposed one.

References

[1] Caulfield, A. M., Chung, E. S., Putnam, A., Angepat, H., Fowers, J., Haselman, M., Heil, S., Humphrey, M., Kaur, P., Kim, J. Y., Lo, D., Massengill, T., Ovtcharov, K., Papamichael, M., Woods, L., Lanka, S., Chiou, D., and Burger, D. A cloud-scale acceleration architecture. In *Proceedings of the IEEE/ACM International Symposium on Microarchitecture (MICRO)* (2016), pp. 1–13.

[2] Dell EMC. *EMC XtremIO 4.0 System Specifications*, 2015. Retrieved May 15, 2017, from `https://www.emc.com/collateral/data-sheet/h12451-xtremio-4-system-specifications-ss.pdf`.

[3] Jun, S.-W., Liu, M., Lee, S., Hicks, J., Ankcorn, J., King, M., Xu, S., and Arvind. Bluedbm: An appliance for big data analytics. In *Proceedings of the International Symposium on Computer Architecture (ISCA)* (2015), pp. 1–13.

[4] Lee, S., Liu, M., Jun, S., Xu, S., Kim, J., and Arvind. Application-managed flash. In *Proceedings of the USENIX Conference on File and Storage Technologies (FAST)* (2016), pp. 339–353.

[5] Redis Lab. *Redis Protocol Specification*. Retrieved May 15, 2017, from `https://redis.io/topics/protocol`.

[6] Wu, X., Xu, Y., Shao, Z., and Jiang, S. Lsm-trie: An lsm-tree-based ultra-large key-value store for small data items. In *Proceedings of the USENIX Annual Technical Conference (USENIX ATC)* (2015), pp. 71–82.