Smart In-Network Deduplication for Storage-aware SDN

Yu Hua
WNLO, School of Computer
Huazhong University of
Science and Technology
Wuhan, China
csyhua@hust.edu.cn

Xue Liu
School of Computer Science
McGill University
Montreal, Canada
xueliu@cs.mcgill.ca

Dan Feng
WNLO, School of Computer
Huazhong University of
Science and Technology
Wuhan, China
dfeng@hust.edu.cn

ABSTRACT

In order to efficiently handle the rapid growth of data and reduce the overhead of network transmission, we propose an in-network deduplication for storage-aware Software Defined Network (SDN), called SMIND. Unlike conventional source or destination deduplication schemes, SMIND implements in-network deduplication via SDN. Moreover, to address the performance bottleneck of accessing and indexing SDN controller, we implement an SDN-enabled Flash Translation Layer (FTL) in a real prototype of Solid State Disk (SSD). Experimental results demonstrate the efficiency and efficacy of SMIND.

Categories and Subject Descriptors

C.2.1 [**COMPUTER-COMMUNICATION NETWORKS**]: Network Architecture and Design

Keywords

Software Defined Network, Deduplication, Storage Systems

1. INTRODUCTION

The amount of information created and replicated has rapidly increase, in which there exists a large fraction of redundant data. The heavy redundancy causes non-trivial overheads to the network performance [1]. One solution is to detect data redundancy and reduce the amount of duplicate data to be transmitted [2, 3].

Data deduplication can be executed in either source (e.g., clients) or destination (e.g., servers) end systems. Specifically, source deduplication is to remove data redundancy before transmission to the destination. It works through client software that communicates with the server to compare arriving data with previously stored data. If the server contains previously stored data, they will not be sent. However, the source deduplication suffers from the long latency of message communications between source and destination systems. On the other hand, destination deduplication is to remove redundant data at the destination servers, after these data have been transmitted to the servers. This scheme consumes substantial resources of servers, while incurring heavy network transmission overheads.

Both source and destination schemes need to handle the data in an end-to-end manner, say between the source and the destination, to obtain the responses. The reason is the property of simply processing data in conventional networks, without data analytics. Software defined network (SDN) [4, 5] offers an opportunity to implement an in-network deduplication. The in-network scheme allows the deduplication to be executed within the network.

Our contributions are twofold. First, based on existing SDN platform using OpenFlow protocol [4], we implement an in-network deduplication scheme. SMIND detects the data redundancy by fast checking the memberships of their fingerprints in the SDN controller. Second, to address the performance bottleneck of indexing data in the SDN controller, we propose to use the SSD devices due to their high "read" performance and energy efficiency. These properties perfectly match the access patterns to the SDN controller. We design and implement Flash Translation Layer (FTL) in SSD to support the SDN functionalities.

2. SSD-BASED SDN CONTROLLER

Since network bandwidth is often a performance-limiting factor, we propose to leverage data reduction techniques to decrease the unmanaged redundancy and improve the effective throughput. The deduplication can split files into multiple chunks. Each chunk generates corresponding hash signature, called a fingerprint. The fingerprint uniquely identifies the chunk. Hence, we can determine duplicate data by comparing fingerprints that summarize the contents. The operations of checking their fingerprints avoid byte-by-byte comparisons. By exploiting and exploring the contents of data, SMIND offers the functionalities of fingerprint generation, fingerprint lookups and mapping management. The generated fingerprints are used for the lookups of redundant data. The mapping function manages the correlation between the host-viewable logical addresses and physical flash addresses in SSD.

We maintain the fingerprint index in the SDN controller. In order to deliver high performance, SSD is used to meet the needs of high I/O (especially "read") performance and energy efficiency. To allow SDN to work in the SSD devices, we design and implement the SDN-enabled FTL. As shown in Figure 1, SSD maintains data in an array of flash blocks. Each block contains 32-64 pages. A page is the smallest unit of read and write operations. FTL is a block-device software layer that simulates the flash as a hard disk. It has the ability to implement address mapping between a logical address in file systems to a physical address in the flash. By offering a disk-like interface, FTL is an intermediate software layer inside an SSD. It receives logical read and write commands from the applications and then transforms them to the internal commands in the flash.

SIGCOMM'13, August 12–16, 2013, Hong Kong, China.
ACM 978-1-4503-2056-6/13/08.

Figure 1 illustrates the deduplication-aware process in the SDN controller. The workflow of SMIND includes fingerprint generation and lookups from indexing requests. Besides the conventional functions in FTL, such as address translation, wear leveling and garbage collection, we improve the FTL by adding SDN-enabled components. The components of virtualized deduplication mainly include redundancy identification, flow table management, per-flow scheduling and energy optimization. SMIND hence can offer efficient data deduplication services by leveraging the salient performance properties of SSD. For example, since SMIND needs to determine the memberships of new arriving data, the SDN controller mainly reads exiting information. SSD has good performance of read operation. Moreover, by using the virtualization, SMIND builds the SDN-enabled control plane for managing large amounts of data.

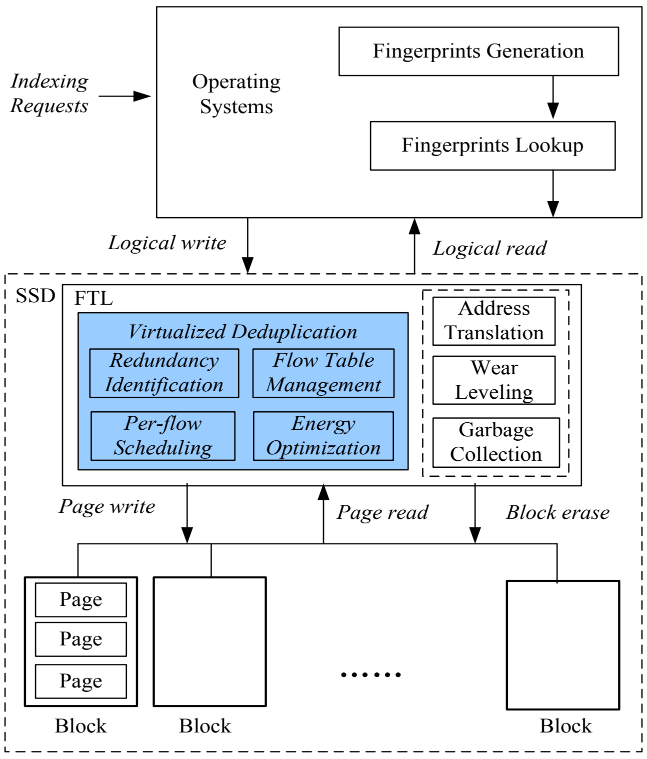

Figure 1: SSD based design in SMIND.

To support and evaluate SDN-enabled functionalities in SSD, we implement a real SSD prototype that is event-driven, modularly structured and multi-tiered. The SSD prototype consists of the buffer and request-scheduling module, the FTL and allocation module, and the low-level hardware platform module. Specifically, the first module is responsible for buffer organization and scheduling requests. In the second module, the FTL module executes the fingerprint based schemes for SDN, and the allocation sub-module supports the allocation between the logical and physical pages. The third module supports the basic flash operations.

3. RESULTS AND CONCLUSION

Figure 2 shows the transmission latencies of SMIND, source and destination deduplication schemes, when transmitting up to 1TB data. The maximum bandwidth is 50Gb/s in our OpenFlow based data centers. The SDN controller uses our SSD prototype. We observe the destination scheme incurs the longest latency due to its full data transmission in the network. Although both source and SMIND need to compute the fingerprints, the source scheme needs to wait for the response from the servers and then decide to transmit the data, which incurs extra latency especially for new data. Unlike them, SMIND obtains the smallest latency due to its in-network deduplication, while alleviating heavy bandwidth overhead and extra waiting time.

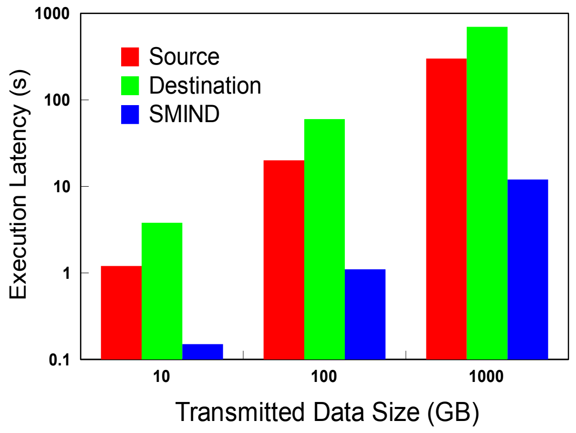

Figure 2: Deduplication based transmission latency.

To the best of our knowledge, SMIND is the first work to implement the SDN-enabled SSD in the software-defined controller. The proposed SMIND has the salient properties of high performance, energy efficiency and storage awareness. Our work efficiently addresses the performance bottleneck of indexing the controller. We have implemented a real prototype of SSD to support the SDN design.

4. ACKNOWLEDGMENT

This work was supported in part by National Natural Science Foundation of China (NSFC) under Grant 61173043; National Basic Research 973 Program of China under Grant 2011CB302301; NSFC under Grant 61025008; the NSERC Discovery Grant 341823; US National Science Foundation Award 1116606. The authors would like to thank the anonymous reviewers for their valuable and constructive comments.

5. REFERENCES

[1] A. Muthitacharoen, B. Chen, and D. Mazieres, “A low-bandwidth network file system,” *Proc. SOSP*, 2001.

[2] A. Anand, V. Sekar, and A. Akella, “SmartRE: an architecture for coordinated network-wide redundancy elimination,” *Proc. SIGCOMM*, 2009.

[3] B. Aggarwal, A. Akella, A. Anand, A. Balachandran, P. Chitnis, C. Muthukrishnan, R. Ramjee, and G. Varghese, “EndRE: an end-system redundancy elimination service for enterprises,” *Proc. NSDI*, 2010.

[4] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, “Openflow: enabling innovation in campus networks,” *ACM SIGCOMM CCR*, vol. 38, no. 2, pp. 69–74, 2008.

[5] M. Yu, J. Rexford, M. J. Freedman, and J. Wang, “Scalable flow-based networking with DIFANE,” *Proc. SIGCOMM*, 2010.