Revisiting Computational Storage for Data Integrity and Security

Chao Shi[†], Anthony Manschula[†], Tabassum Mahmud[†], Zeren Yang[§], Mai Zheng[†]
Yong Chen[‡], Jim Wayda[‡], Matthew Wolf[‡], Byungwoo Bang[‡]
[†]Iowa State University [§]University of Wisconsin-Madison [‡]Samsung

The idea of computational storage device (CSD) has come a long way since at least 1990s [1], [22]. By embedding computing resources within storage devices, CSDs could potentially offload computational tasks from CPUs and enable near-data processing (NDP), reducing data movements and/or energy consumption significantly. While the initial hard-disk-based CSDs suffer from severe limitations in terms of on-drive resources, programmability, etc., the storage market has witnessed the commercialization of solid-state-drive (SSD) based CSDs (e.g., Samsung SmartSSD [2], ScaleFlux CSDs [15]) recently, which has enabled CSD-based optimizations for a variety of application scenarios (e.g., [6], [20], [14]).

Nevertheless, existing CSD research efforts mainly focus on performance acceleration of regular operations, leaving the potentials on system reliability/security largely unexplored. In this work, we attempt to bridge the gap. We revisit the classic idea of CSDs from a new angle: Can we leverage CSD to improve data integrity and/or security? To answer the question, we look into three representative I/O-intensive reliability/security techniques for data protection, and explore their similarities and potentials for CSD-based optimizations:

- *Fault Injection (FI)* is an indispensable method for testing the failure recovery of various storage systems (e.g., [19], [9], [10], [27], [8], [5], [11]). We observe that the core operations of FI typically involve *intercepting I/O blocks* at certain software layer (e.g., kernel block layer [19], FUSE [8], drivers [10], [5], [11]) to implement the functionality. A CSD-based FI solution could potentially achieve similar I/O interception and manipulation at the bottom of the storage stack (i.e., device) to enable full-stack testing with high fidelity.
- *Erasure Coding (EC)* is an essential fault-tolerance mechanism for modern distributed storage systems (DSS) (e.g., Ceph [7], HDFS [23]). We observe that the core operations of EC involve *matrix multiplications* for encoding/decoding. In particular, locally repairable codes (LRC) [13] have been proposed to reduce the network and/or storage I/O cost by leveraging local parities, which could potentially benefit from FPGA-based optimization with a small set of collaborative CSDs.
- *Ransomware Detection & Recovery (RDR)* is increasingly important for protecting user data as ransomware has grown to a national security threat recently [16]. We observe that one major category of RDR solutions rely on SSDs [4], [3], [12], [17], [18], [21], [26] or hypervisor [25] to achieve *I/O pattern monitoring* for ransomware detection and *intra-device data movement* for data recovery, both of which aligns well with CSD characteristics. A CSD-based RDR could potentially achieve higher flexibility (compared to regular SSD-based RDR) and efficiency (compared to hypervisor-based RDR).

Based on the key observations above, we design a generic SmartSSD CSD library called `CSDGuard` to serve as a building block for constructing CSD-optimized reliability/security solutions. The library follows the Computational Storage Architecture Programming Model[24] to cover the core operations (e.g., host-device buffer management, I/O interception and monitoring, multi-dimensional array multiplication) of representative FI, EC, and RDR algorithms. Moreover, it provides a simple set of APIs to abstract away unnecessary CSD internals and support controlling data and metadata operations between host and CSDs with flexible configurability.

To demonstrate the potential of such a solution, we build a prototype of `CSDGuard` based on the Samsung SmartSSD platform [2]. The prototype leverages the peer-to-peer (P2P) transfer between NVMe flash storage and on-drive FPGA to minimize data communication between the host and the CSD, and applies a set of directive-based optimizations (e.g., `HLS INTERFACE`, `HLS ARRAY_PARTITION`, `HLS UNROLL`) to make full use of the massive parallelism of FPGA and thus achieve efficient near data processing. Our preliminary results are promising: Measuring the execution time of our library with directive-based optimizations applied, the overall latency was successfully reduced up to 70% across several experimental data sizes (e.g., the tested matrix size ranges from 384x384 to 2048x2048). With regard to P2P data transfer time, we observed similar performance to the conventional software-based data transfer approach between the CSD and host device. We believe we may be incurring some additional overhead in the system calls, which may lead to the behavior that we observed. We plan to extend the preliminary prototype to cover different use cases (e.g., FI, EC, and RDR) and evaluate with realistic systems (e.g., Ceph/HDFS with EC configuration) and datasets (e.g., VirusTotal) to fully demonstrate the potentials of CSD for data protection.

Acknowledgements: This work was supported in part by National Science Foundation (NSF) under grands CNS-1855565 and CNS-1943204, and a Global Research Outreach (GRO) Award from Samsung (2022).

REFERENCES

[1] Anurag Acharya, Mustafa Uysal, and Joel Saltz. Active disks: programming model, algorithms and evaluation. *SIGOPS Oper. Syst. Rev.*, 32(5):81–91, oct 1998.

[2] AMD Xilinx. Samsung smartssd. https://www.xilinx.com/applications/data-center/computational-storage/smartssd.html.

[3] SungHa Baek, Youngdon Jung, Aziz Mohaisen, Sungjin Lee, and DaeHun Nyang. Ssd-insider: Internal defense of solid-state drive against ransomware with perfect data recovery. In *2018 IEEE 38th International Conference on Distributed Computing Systems (ICDCS)*, pages 875–884, 2018.

[4] Sungha Baek, Youngdon Jung, David Mohaisen, Sungjin Lee, and DaeHun Nyang. Ssd-assisted ransomware detection and data recovery techniques. *IEEE Transactions on Computers*, 70(10):1762–1776, 2021.

[5] Jinrui Cao, Om Rameshwar Gatla, Mai Zheng, Dong Dai, Vidya Eswarappa, Yan Mu, and Yong Chen. Pfault: A general framework for analyzing the reliability of high-performance parallel file systems. In *Proceedings of the 2018 International Conference on Supercomputing*, ICS '18, page 1–11, New York, NY, USA, 2018. Association for Computing Machinery.

[6] Wei Cao, Yingqiang Zhang, Xinjun Yang, Feifei Li, Sheng Wang, Qingda Hu, Xuntao Cheng, Zongzhi Chen, Zhenjun Liu, Jing Fang, Bo Wang, Yuhui Wang, Haiqing Sun, Ze Yang, Zhushi Cheng, Sen Chen, Jian Wu, Wei Hu, Jianwei Zhao, Yusong Gao, Songlu Cai, Yunyang Zhang, and Jiawang Tong. Polardb serverless: A cloud native database for disaggregated data centers. In *Proceedings of the 2021 International Conference on Management of Data*, SIGMOD '21, page 2477–2489, New York, NY, USA, 2021. Association for Computing Machinery.

[7] Ceph. (accessed april 3, 2024), https://ceph.com/en/.

[8] Aishwarya Ganesan, Ramnatthan Alagappan, Andrea C. Arpaci-Dusseau, and Remzi H. Arpaci-Dusseau. Redundancy does not imply fault tolerance: analysis of distributed storage reactions to single errors and corruptions. In *Proceedings of the 15th Usenix Conference on File and Storage Technologies*, FAST'17, page 149–165, USA, 2017. USENIX Association.

[9] Om Rameshwar Gatla, Muhammad Hameed, Mai Zheng, Viacheslav Dubeyko, Adam Manzanares, Filip Blagojević, Cyril Guyot, and Robert Mateescu. Towards robust file system checkers. In *Proceedings of the 16th USENIX Conference on File and Storage Technologies (FAST)*, 2018.

[10] Om Rameshwar Gatla, Mai Zheng, Muhammad Hameed, Viacheslav Dubeyko, Adam Manzanares, Filip Blagojevic, Cyril Guyot, and Robert Mateescu. Towards robust file system checkers. *ACM Trans. Storage*, 14(4), dec 2018.

[11] Runzhou Han, Om Rameshwar Gatla, Mai Zheng, Jinrui Cao, Di Zhang, Dong Dai, Yong Chen, and Jonathan Cook. A study of failure recovery and logging of high-performance parallel file systems. *ACM Trans. Storage*, 18(2), apr 2022.

[12] Jian Huang, Jun Xu, Xinyu Xing, Peng Liu, and Moinuddin K. Qureshi. Flashguard: Leveraging intrinsic flash properties to defend against encryption ransomware. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, CCS '17, page 2231–2244, New York, NY, USA, 2017. Association for Computing Machinery.

[13] Saurabh Kadekodi, Shashwat Silas, David Clausen, and Arif Merchant. Practical design considerations for wide locally recoverable codes (lrcs). *ACM Trans. Storage*, 19(4), nov 2023.

[14] Hwajung Kim, Heon Y Yeom, and Hanul Sung. Understanding the performance characteristics of computational storage drives: A case study with smartssd. *Electronics*, 10(21):2617, 2021.

[15] Libin Liu, Hong Xu, Zhixiong Niu, Jingzong Li, Wei Zhang, Peng Wang, Jiamin Li, Jason Chun Xue, and Cong Wang. Scaleflux: Efficient stateful scaling in nfv. *IEEE Transactions on Parallel and Distributed Systems*, 33(12):4801–4817, 2022.

[16] Boyang Ma, Yilin Yang, Jinku Li, Fengwei Zhang, Wenbo Shen, Yajin Zhou, and Jianfeng Ma. Travelling the hypervisor and ssd: A tag-based approach against crypto ransomware with fine-grained data recovery. In *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security*, CCS '23, page 341–355, New York, NY, USA, 2023. Association for Computing Machinery.

[17] Donghyun Min, Yungwoo Ko, Ryan Walker, Junghee Lee, and Youngjae Kim. A content-based ransomware detection and backup solid-state drive for ransomware defense. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 41(7):2038–2051, 2022.

[18] Jisung Park, Youngdon Jung, Jonghoon Won, Minji Kang, Sungjin Lee, and Jihong Kim. Ransomblocker: a low-overhead ransomware-proof ssd. In *2019 56th ACM/IEEE Design Automation Conference (DAC)*, pages 1–6, 2019.

[19] Vijayan Prabhakaran, Lakshmi N. Bairavasundaram, Nitin Agrawal, Haryadi S. Gunawi, Andrea C. Arpaci-Dusseau, and Remzi H. Arpaci-Dusseau. Iron file systems. In *Proceedings of the Twentieth ACM Symposium on Operating Systems Principles*, SOSP '05, page 206–220, New York, NY, USA, 2005. Association for Computing Machinery.

[20] Yifan Qiao, Xubin Chen, Ning Zheng, Jiangpeng Li, Yang Liu, and Tong Zhang. Closing the b+-tree vs.{LSM-tree} write amplification gap on modern storage hardware with built-in transparent compression. In *20th USENIX Conference on File and Storage Technologies (FAST 22)*, pages 69–82, 2022.

[21] Benjamin Reidys, Peng Liu, and Jian Huang. Rssd: defend against ransomware with hardware-isolated network-storage codesign and post-attack analysis. In *Proceedings of the 27th ACM International Conference on Architectural Support for Programming Languages and Operating Systems*, ASPLOS '22, page 726–739, New York, NY, USA, 2022. Association for Computing Machinery.

[22] Erik Riedel, Garth A. Gibson, and Christos Faloutsos. Active storage for large-scale data mining and multimedia. In *Proceedings of the 24rd International Conference on Very Large Data Bases*, VLDB '98, page 62–73, San Francisco, CA, USA, 1998. Morgan Kaufmann Publishers Inc.

[23] Konstantin Shvachko, Hairong Kuang, Sanjay Radia, and Robert Chansler. The hadoop distributed file system. In *2010 IEEE 26th Symposium on Mass Storage Systems and Technologies (MSST)*, pages 1–10, 2010.

[24] SNIA. Computational Storage Architecture and Programming Model v1.0. https://www.snia.org/sites/default/files/technical-work/computational/release/SNIA-Computational-Storage-Architecture-and-Programming-Model-1.0.pdf. Accessed: 2022-08-30.

[25] Fei Tang, Boyang Ma, Jinku Li, Fengwei Zhang, Jipeng Su, and Jianfeng Ma. Ransomspector: An introspection-based approach to detect crypto ransomware. *Computers & Security*, 97:101997, 2020.

[26] Peiying Wang, Shijie Jia, Bo Chen, Luning Xia, and Peng Liu. Mimosaftl: Adding secure and practical ransomware defense strategy to flash translation layer. In *Proceedings of the Ninth ACM Conference on Data and Application Security and Privacy*, CODASPY '19, page 327–338, New York, NY, USA, 2019. Association for Computing Machinery.

[27] Mai Zheng, Joseph Tucek, Dachuan Huang, Feng Qin, Mark Lillibridge, Elizabeth S. Yang, Bill W Zhao, and Shashank Singh. Torturing Databases for Fun and Profit. In *11th USENIX Symposium on Operating Systems Design and Implementation (OSDI 14)*, pages 449–464, 2014.