Tracing-Based I/O Diagnosis for Performance Optimization of Software-Defined Storage Systems

Lina SADI†‡ - Chakib Islam KEDADSA†‡ - Vincent LANNURIEN† - Stéphane RUBINI* - Alexandre SKRZYNIARZ^ - Jalil BOUKHOBZA†

† Lab-STICC, CNRS UMR 6285, ENSTA, Institut Polytechnique de Paris, 29806 Brest, France
‡ Ecole nationale Supérieure d'Informatique (ESI-Alger)
* Lab-STICC, CNRS UMR 6285, Université de Bretagne Occidentale
^ THALES DEFENSE & MISSION SYSTEMS, Brest, France

Outline

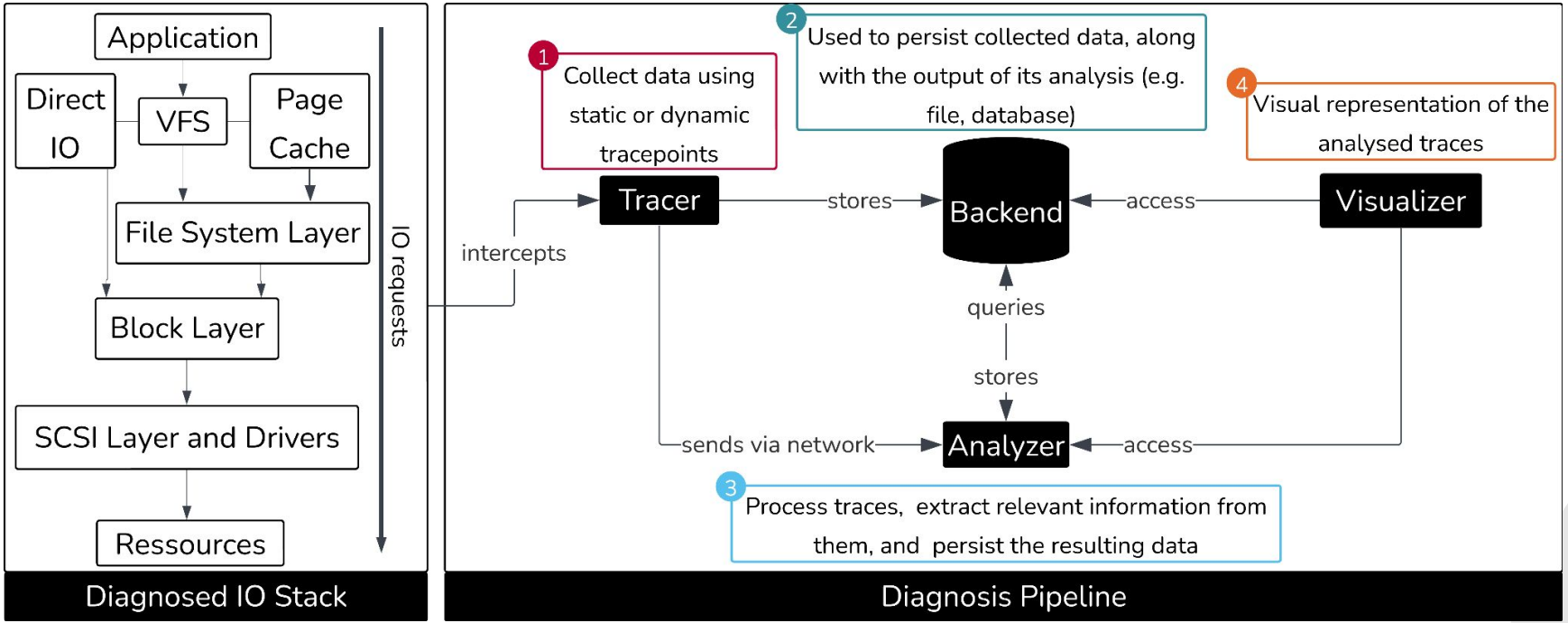
01 Context and problem statement

- SDS provides flexibility and scalability for edge computing.
- Performance bottlenecks occur under heavy workloads.
- Tracing offers fine-grained visibility into I/O operations.
- Kernel-level optimizations can mitigate these bottlenecks.

Problem Statement

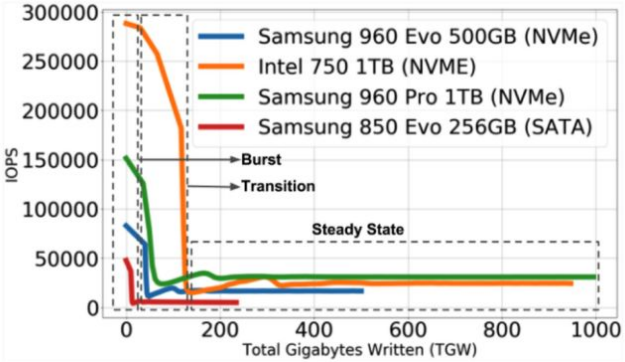
How can tracer-based analysis help diagnose and optimize I/O performance in SDS for edge environments?

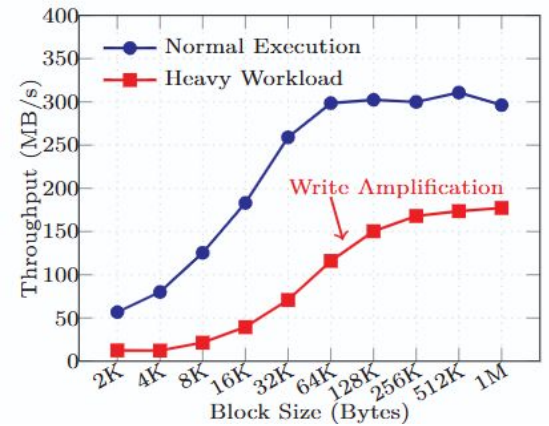
02 Background



03 Experimental Motivation

- SSDs are common in SDS due to performance and energy efficiency.
- However, their internal mechanisms can cause performance degradation
- Main Issue: **Write amplification** causes excess data writing.





- Method: I/O traces collected from Linux with different SSDs under varied workloads.
- Result: Throughput significantly drops under heavy loads across block sizes.

04 Tracing-based diagnosis and correction framework

1 **Offline modeling**

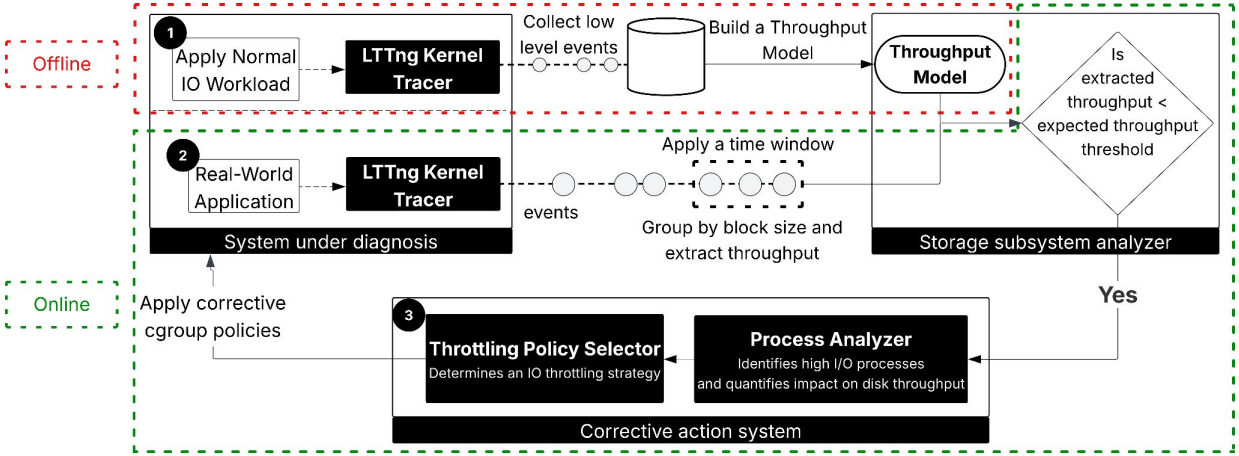
Builds a baseline throughput model from controlled workloads, using tracepoints focused on SSD.

2 **Online tracing**

Monitors real-time I/O, compares measured throughput to the baseline to detect deviations.

3 **Optimization**

On degradation, identifies intensive I/O processes and applies cgroup-based throttling to maintain performance.



05 Conclusion and Perspectives

Our contributions include:

- Online detection of SSD throughput degradation using low-level kernel tracepoints.
- Adaptive correction via cgroup throttling to mitigate performance issues.

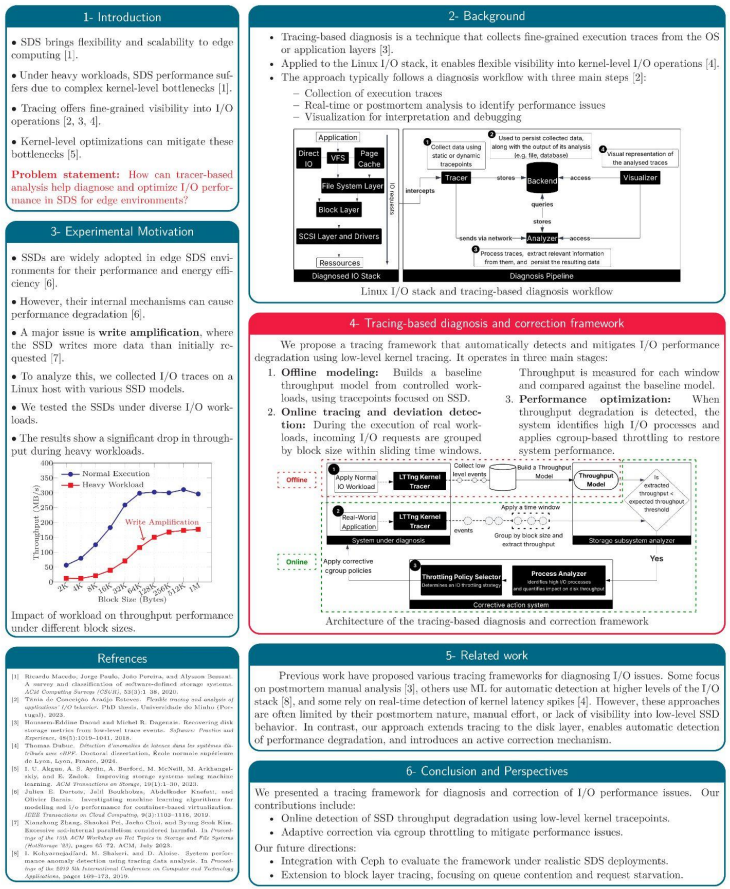
Our future directions:

- Integration with Ceph to evaluate the framework under realistic SDS deployments.
- Extension to block layer tracing, focusing on queue contention and request starvation.

Thank you !

I invite you to discuss it in front of my poster

- lina.sadi@ensta.fr/ kl_sadi@esi.dz
- islam.kedadsa@ensta.fr/ki_kedadsa@esi.dz
- vincent.lannurien@ensta.fr
- stephane.rubini@univ-brest.fr
- alexandre.skrzyniarz@fr.thalesgroup.com
- jalil.boukhobza@ensta.fr



Tracing-Based I/O Diagnosis for Performance Optimization of Software-Defined Storage Systems

Lina Sadi*[1,2], Islam Chakib Kedadsa*[1,2], Vincent Lannurien[1], Stephane Rubini[4], Alexandre Skrzyniarz [3], Jalil Boukhobza[1]

[1] Lab-STICC, CNRS UMR 6285, ENSTA, Institut Polytechnique de Paris, 29806 Brest, France
[2] Ecole nationale Supérieure d'Informatique (ESI-Alger)
[3] THALES DEFENSE & MISSION SYSTEMS, Brest, France
[4] Lab-STICC, CNRS UMR 6285, Université de Bretagne Occidentale
* These 2 authors contributed equaly to the design of this contribution

1- Introduction

- SDS brings flexibility and scalability to edge computing [1].
- Under heavy workloads, SDS performance suffers due to complex kernel-level bottlenecks [1].
- Tracing offers fine-grained visibility into I/O operations [2, 3, 4].
- Kernel-level optimizations can mitigate these bottlenecks [5].

Problem statement: How can tracer-based analysis help diagnose and optimize I/O performance in SDS for edge environments?

2- Background

- Tracing-based diagnosis is a technique that collects fine-grained execution traces from the OS or application layers [3].
- Applied to the Linux I/O stack, it enables flexible visibility into kernel-level I/O operations [4].
- The approach typically follows a diagnosis workflow with three main steps [2]:
  - Collection of execution traces
  - Real-time or postmortem analysis to identify performance issues
  - Visualization for interpretation and debugging

Linux I/O stack and tracing-based diagnosis workflow

3- Experimental Motivation

- SSDs are widely adopted in edge SDS environments for their performance and energy efficiency [6].
- However, their internal mechanisms can cause performance degradation [6].
- A major issue is **write amplification**, where the SSD writes more data than initially requested [7].
- To analyze this, we collected I/O traces on a Linux host with various SSD models.
- We tested the SSDs under diverse I/O workloads.
- The results show a significant drop in throughput during heavy workloads.

Impact of workload on throughput performance under different block sizes.

4- Tracing-based diagnosis and correction framework

We propose a tracing framework that automatically detects and mitigates I/O performance degradation using low-level kernel tracing. It operates in three main stages:

1. **Offline modeling:** Builds a baseline throughput model from controlled workloads, using tracepoints focused on SSD.
2. **Online tracing and deviation detection:** During the execution of real workloads, incoming I/O requests are grouped by block size within sliding time windows. Throughput is measured for each window and compared against the baseline model.
3. **Performance optimization:** When throughput degradation is detected, the system identifies high I/O processes and applies cgroup-based throttling to restore system performance.

Architecture of the tracing-based diagnosis and correction framework

Refrences

[1] Ricardo Macedo, Jorge Paulo, João Pereira, and Alysson Bessani. A survey and classification of software-defined storage systems. *ACM Computing Surveys (CSUR)*, 53(3):1–38, 2020.

[2] Tânia do Conceição Araújo Esteves. *Flexible tracing and analysis of applications' I/O behavior*. PhD thesis, Universidade do Minho (Portugal), 2023.

[3] Reinaldo-Eddine Daoud and Michel R. Dagenais. Recovering disk storage metrics from low-level trace events. *Software: Practice and Experience*, 48(5):1019–1041, 2018.

[4] Thomas Dubuc. *Détection d'anomalies de latence dans les systèmes distribués avec eBPF*. Doctoral dissertation, École normale supérieure de Lyon, Lyon, France, 2024.

[5] I. U. Akgun, A. S. Aydin, A. Burford, M. McNeill, M. Arkhangelsky, and E. Zadok. Improving storage systems using machine learning. *ACM Transactions on Storage*, 19(1):1–30, 2023.

[6] Julien E. Dorsioz, Jalil Boukhobza, Abdelkader Kouhri, and Olivier Dorsin. Investigating machine learning algorithms for modeling ssd i/o performance for container-based virtualization. *IEEE Transactions on Cloud Computing*, 9(2):1102–1116, 2019.

[7] Xiaosheng Zhang, Shoukai Pei, Jaeho Choi, and Sysung Souk Kim. Excessive unintentional parallelism considered harmful. In *Proceedings of the 15th ACM Workshop on Hot Topics in Storage and File Systems (HotStorage '23)*, pages 65–72. ACM, July 2023.

[8] I. Kobyzevmajdard, M. Shakeri, and D. Alaize. System performance anomaly detection using tracing data analysis. In *Proceedings of the 2019 5th International Conference on Computer and Technology Applications*, pages 169–173, 2019.

5- Related work

Previous work have proposed various tracing frameworks for diagnosing I/O issues. Some focus on postmortem manual analysis [3], others use ML for automatic detection at higher levels of the I/O stack [8], and some rely on real-time detection of kernel latency spikes [4]. However, these approaches are often limited by their postmortem nature, manual effort, or lack of visibility into low-level SSD behavior. In contrast, our approach extends tracing to the disk layer, enables automatic detection of performance degradation, and introduces an active correction mechanism.

6- Conclusion and Perspectives

We presented a tracing framework for diagnosis and correction of I/O performance issues. Our contributions include:

- Online detection of SSD throughput degradation using low-level kernel tracepoints.
- Adaptive correction via cgroup throttling to mitigate performance issues.

Our future directions:

- Integration with Ceph to evaluate the framework under realistic SDS deployments.
- Extension to block layer tracing, focusing on queue contention and request starvation.