Bespoke Cache Enclaves: LLC Set-Partitioning & Dead Block Prediction

Wamique Zia
Department of CSE
IIT Ropar
Ropar, India
2023csm1020@iitrpr.ac.in

T. V. Kalyan
Department of CSE
IIT Ropar
Ropar, India
kalyantv@iitrpr.ac.in

***Abstract* — Cache partitioning is a critical aspect of modern computer security, especially in multi-core systems where last-level caches (LLCs) are shared among multiple cores. These caches, while essential for enhancing performance, also pose security risks due to the potential for Cache Side-Channel attacks. As technology advances, the need for robust cache partitioning mechanisms becomes increasingly apparent.**

Bespoke Cache Enclaves (BCE) represents a significant step forward in addressing these challenges. By offering a scalable and flexible approach to cache partitioning, BCE ensures that hundreds of isolated cache partitions can coexist within a system without compromising performance or security. This level of granularity in cache allocation allows for efficient resource utilization while effectively thwarting side-channel attacks. While BCE provides security from Cache Side Channel attacks, it strives to maintain performance comparable to non-secured cache configurations. In this paper, we further enhance performance by integrating dead block prediction techniques. By proactively identifying and evicting unused cache blocks, BCE optimizes cache utilization, thereby improving overall system performance.

Keywords—Cache Partitioning, Dead Block

I. INTRODUCTION

Last-level caches (LLCs) represent a crucial shared resource in multi-core systems, enabling efficient data access across different processor cores. However, this shared nature also introduces vulnerabilities, particularly in the form of Cache Side-Channel attacks, which exploit unintended channels to extract sensitive information from victim processes. In timing-based cache side-channel attacks, for instance, attackers leverage variations in cache access latencies to infer accessed data without direct access.

To mitigate such security risks, various countermeasures like cache partitioning (way-partitioning and page coloring) and cache randomization techniques have been proposed. While effective, these countermeasures often come at a cost, leading to reduced system performance and increased complexity. Addressing these challenges, Bespoke Cache Enclave introduces a flexible set-based cache partitioning technique. This approach ensures fine-grained and scalable isolation from cache side channels by assigning caches to domains as clusters.

II. MOTIVATION

Cache-side channel attacks pose significant threats by exploiting information leakage through cache behaviour, potentially compromising system integrity and violating user privacy. In timing-based cache side-channel attacks, attackers leverage latency variations to discern accessed cache blocks, thereby extracting sensitive data. However, existing mitigation techniques often offer only coarse-grained isolation from such attacks, limiting their effectiveness. For instance, way-based set-partitioning is constrained by the number of cache ways available, resulting in inefficient allocation for memory-intensive processes. Moreover, existing solutions may incur notable performance overheads, hampering system efficiency. A more effective approach is required, one that offers fine-grained and scalable isolation from cache side-channel attacks on the last level cache while ensuring uniform and faster lookup for every memory access. Additionally, incorporating improvements such as dead block prediction can further enhance system performance while bolstering the security provided by the set partitioning technique of BCE.

III. PROPOSED IDEA

A. Implementation of BCE

The Bespoke cache enclave comprises two modules: the Load Balancing Hash (LBH) and the Cluster Indirection Module (CIM). When a new process is initiated, it is assigned either an existing or a new domain. If security is required, the operating system (OS) allocates an isolated partition in the LLC and assigns a domain ID to the process. All read/write accesses to the LLC in Bespoke Cache Enclave (BCE) are accompanied by the Domain ID.

LBH is responsible for uniformly mapping line addresses to logical clusters of the domain (LCID). It employs multiple randomizing hashes to achieve this. If the LCID is not obtained directly, a low-latency randomizing hash function is applied. The Cluster Indirection Module (CIM) translates LCIDs to Physical Cluster IDs (PCIDs) by referencing the Domain Base Table (DBT) and Cluster Location Table (CLT).

The DBT tracks the location of the base entry (LCID-0) in the CLT for each domain, while the CLT maintains an ordered list of valid LCID to PCID mappings. The CIM ensures that only LLC clusters allocated to a domain are accessible to cache lookups from that domain, providing cache isolation security.

BCE supports dynamic LLC partitioning using the BCE alloc instruction, which creates new LLC partitions for trusted domains. Conversely, the BCE dealloc instruction deallocates existing LLC partitions, flushing the lines in the physical LLC clusters of the domain and making PCIDs available for reallocation.

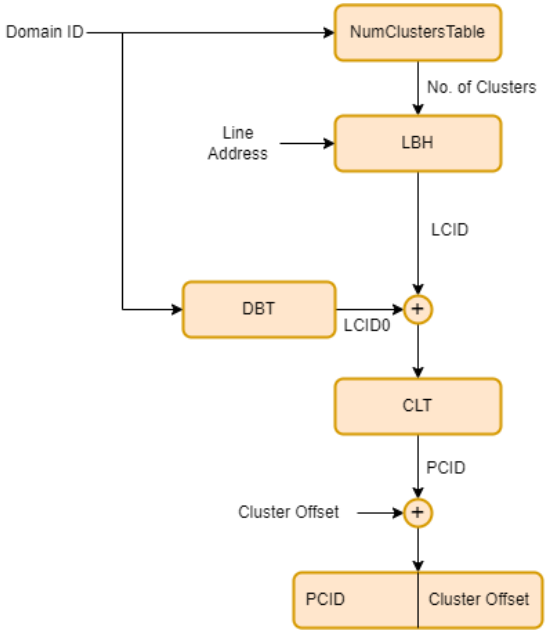

Fig. 1. BCE Architecture

B. Dead Block Prediction

To efficiently manage the removal of unused blocks from the cache, we devised a sophisticated approach centered around a re-reference map. This map, aptly named BLOCK_ACCESS, serves as a comprehensive repository, linking the domain ID (utilizing num_cpu) with the specific memory address (designated as full_address).

Each time a block is referenced, its count in the map is incremented, essentially keeping track of its re-referencing frequency. This meticulous tracking mechanism ensures that the most frequently accessed blocks remain in the cache, optimizing cache utilization.

Now, when a conflict arises necessitating the eviction of a block from the set, a careful selection process ensues. The current_cpu and full_address of the set in contention are utilized as the search criteria within our BLOCK_ACCESS map. Once located, the re-reference counts of all blocks residing in the set are compared.

The block with the minimum re-reference count is identified as the eviction candidate, signifying its lesser importance in the cache hierarchy. Subsequently, the re-reference count for this block is reset to zero, effectively denoting its absence from the last level cache.

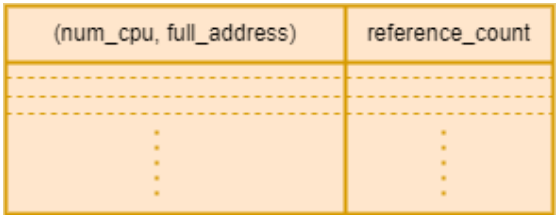

Fig. 2. Structure of Block Access map

IV. Experimental Setup

A. Simulator Details

To assess the performance of our proposed implementation, we conducted experiments using the ChampSim Multicore Out-of-Order Simulator. Each experiment involved a warmup phase comprising 10 million instructions across all scenarios. For single-core experiments, simulations were executed using 200 million instructions. Meanwhile, for experiments involving 2-core and 4-core configurations, our code was run for 100 million instructions to capture relevant performance metrics.

B. Configuration

Our experimental setup encompassed a 32KB 8-way L1 cache for instructions, a 512KB 8-way L2 cache, and a 32MB 16-way L3 cache. We employed a trace-driven simulator, ChampSim, to execute program execution traces spanning 10 million instructions.

V. Results

In many of the scenarios, it has been observed that the improvement on the Hit Rate is not consistent among various workloads. In some cases, such as 2-core run for perlbench and gcc, Hit Rate is improved by ~3% for perlbench, but simultaneously it is reduced for gcc by ~2%.

Similar phenomenon is seen throughout the various workloads, meaning that the dead block prediction was not effective in its working, employing further study into it.

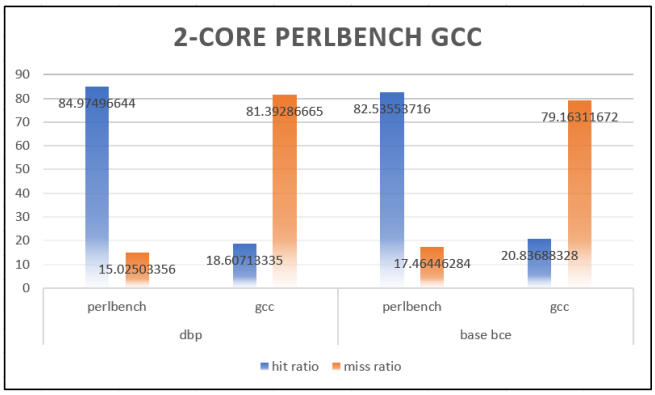

Fig. 3. 2-core perlbench gcc run

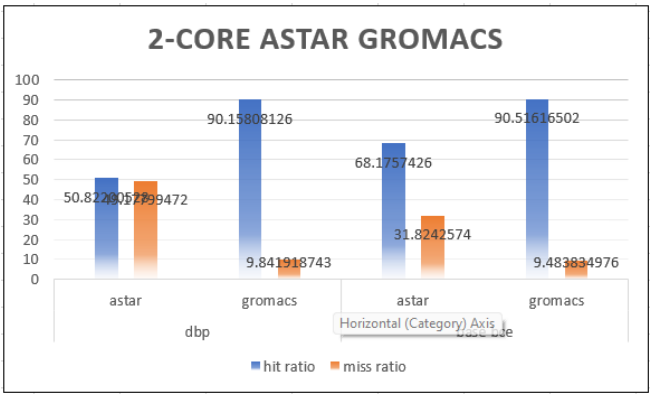

Fig. 4. 2-core astar gromacs run

ACKNOWLEDGMENT

The reimplementation of Bespoke Cache Enclaves is based on the work done by Harsh Raj (2022csm1004@iitrpr.ac.in) and Kumar Mangalam (2022aim1002@iitrpr.ac.in). Their work help in recreating the working of the BCE set-partitioning as mentioned in the original paper.

REFERENCES

[1] G. Saileshwar, S. Kariyappa and M. Qureshi, "Bespoke Cache Enclaves: Fine-Grained and Scalable Isolation from Cache Side-Channels via Flexible Set-Partitioning," *2021 International Symposium on Secure and Private Execution Environment Design (SEED)*, Washington, DC, USA, 2021, pp. 37-49.

[2] V. Kiriansky, I. Lebedev, S. Amarasinghe, S. Devadas and J. Emer, "DAWG: A Defense Against Cache Timing Attacks in Speculative Execution Processors," *2018 51st Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*, Fukuoka, Japan, 2018, pp. 974-987.

[3] Thomas Bourgeat, Ilia Lebedev, Andrew Wright, Sizhuo Zhang, Arvind, and Srinivas Devadas. 2019. MI6: Secure Enclaves in a Speculative Out-of-Order Processor. In Proceedings of the 52nd Annual IEEE/ACM International Symposium on Microarchitecture (MICRO '52). Association for Computing Machinery, New York, NY, USA, 42–56.