Performance Implications for Multi-Core RISC-V Systems with Dedicated Security Hardware

Samuel Chadwick, Scott Graham and James Dean
Air Force Institute of Technology, Wright-Patterson Air Force Base, USA
samuel.chadwick@afit.edu
scott.graham@afit.edu
james.dean@afit.edu

Abstract: The RISC-V instruction set architecture (ISA) is a promising open-source architecture supporting the Open Era of Computing. As RISC-V matures, consumers, industry leaders, and nation states are looking at the potential benefits RISC-V offers –especially for secure systems which may require privileged architecture implementations, physical memory protection (PMP), or trusted execution environments (TEEs) among other hardware-based security primitives. The inclusion of these security technologies unavoidably impacts the performance of any given compute system. To quantify the performance impacts introduced by secure enclave processing, representative computational benchmarks are executed on the Freedom U74-MC System-on-a-Chip (SoC) onboard the HiFive Unmatched development board by SiFive. These benchmarks are conducted across applicable modes of the RISC-V Privileged ISA specification to analyze Privileged ISA and PMP performance implications for Confidential Computing. To evaluate performance impacts, a theoretical model is applied to represent the interactions of the security monitor. The Keystone enclave framework tasks the security monitor with enforcing strict adherence to system security primitives while the Phoronix Test Suite (PTS) captures performance data. Individual benchmarks are conducted both with and without secure enclave technologies to characterize representative performance metrics.

Keywords: Confidential Computing, Keystone Security Monitor, Performance Characterization, Physical Memory Protection, RISC-V, Secure Enclave, Trusted Execution Environment

1. Introduction

1.1 Motivation

Existing encryption techniques provide confidentiality, integrity, and availability for *data at rest* and for *data in transit* yet are rarely employed to protect *data in use* during execution. Confidential Computing performs computation within hardware-based Trusted Execution Environments (TEEs) to adequately protect *data in use*. TEEs ensure that compute systems maintain confidentiality, integrity, and availability by implementing security primitives to provide secure boot, a secure source of randomness, and enable remote attestation. While proprietary TEE implementations exist —such as Intel Software Guard eXtensions (SGX), Advanced RISC Machines (ARM) TrustZone, and Advanced Micro Devices (AMD) Secure Encrypted Virtualization— they each impose threat model limitations and restrict changes to their underlying architectural Intellectual Property (IP). Keystone Enclave is an extensible open-source TEE alternative which leverages the Privileged RISC-V Instruction Set Architecture (ISA), Physical Memory Protection (PMP), and other RISC-V Security Primitives to enforce Confidential Computing. Keystone Enclave reduces the Trusted Computing Base (TCB) by applying a software-defined and hardware-enforced framework for developing custom threat models.

1.2 Hypothesis

Confidential Computing unavoidably impacts system performance.

1.3 Approach

To appropriately evaluate architectural design implementations, a functional closed queueing network is applied to measure average system response times for secure enclave applications running on RISC-V systems. Performance impacts are characterized by applying statistical methodologies to ensure that compute systems fulfil operational requirements while simultaneously enforcing security obligations.

Leveraging the HiFive Unmatched as the system under test, we evaluate representative workstation workloads via benchmarking: First with an unmodified Linux system without the Keystone Enclave; Then with the Keystone Enclave installed, but with the SM features unused; And finally, with the Keystone Enclave installed, with varying instances of secure enclaves. Each test records the execution time and assigns benchmark scores accordingly. To ensure accurate statistical and analytical analyses, we perform automated benchmark scheduling to dynamically repeat tests until our results fall within a predefined standard deviation. We then evaluate collected

data under the applied modelling framework to characterize trade-offs between overall system performance and security.

The next sections within this paper describe the following information: Section II provides relevant background knowledge; Section III describes experimental design configurations; Section IV describes preliminary benchmark results; while Section V concludes with current and future contributions.

2. Background

2.1 Modelling framework efficacy

Gorbachov et al. (2019) modelled embedded system performance impacts by applying a subject-object system framework to the simple closed queue network model. The active component responsible for managing and controlling access of subjects to objects is broadly described as a Reference Monitor (Gorbachov et al., 2019). As implemented, Keystone applies this theoretical model by leveraging RISC-V security primitives (such as the privileged ISA modes and PMP capabilities) to construct customizable TEEs (Lee, et al., 2020) and rebrands the previously described Reference Monitor subject as the Security Monitor (SM). Recent research has demonstrated the non-triviality for creation, execution, and destruction of secure enclaves for RISC-V systems and applications (Tullos et al., 2020). This work extends prior study by maturing the system under test from Field Programmable Gate Array (FPGA) technology to dedicated Application Specific Integrated Circuits (ASICs) found in commodity personal computer (PC) hardware.

The modified functional closed queuing network model, transposed for Keystone SM by Tullos, et al. (2021) is shown in Figure 1. This transposition characterizes secure enclave *creation* latency to correlate performance metrics with secure execution. From this model, the system response time R can be measured and regarded as performance overhead for tasks which require enclave initialization.

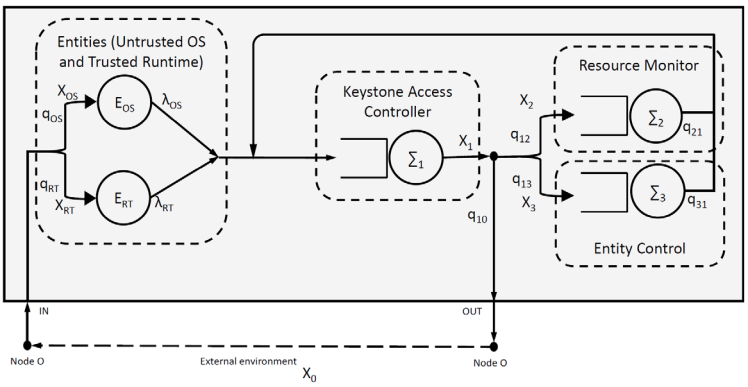


Figure 1: Functional closed queuing network model for Keystone SM enclave creation (Tullos, et al., 2021)

As proposed by Gorbachov et al. (2019) and as applied by Tullos et al. (2021), system service time R is based on the system output rate X_0 where the system flow-balance equations are as follows:

$$X_0 = q_{10} \cdot X_1$$
$$X_1 = X_0 + X_2 + X_3$$
$$X_2 = q_{12} \cdot X_1$$
$$X_3 = q_{13} \cdot X_1$$

The Interactive Response Time Formula presented by Gorbachov et al. (2019) is applied to calculate the mean system service time R, where N represents the number of subjects (secure processes), and λ describes the rate of requests for objects (secure accesses):

$$R = \frac{N}{X_0} - \frac{1}{\lambda}$$

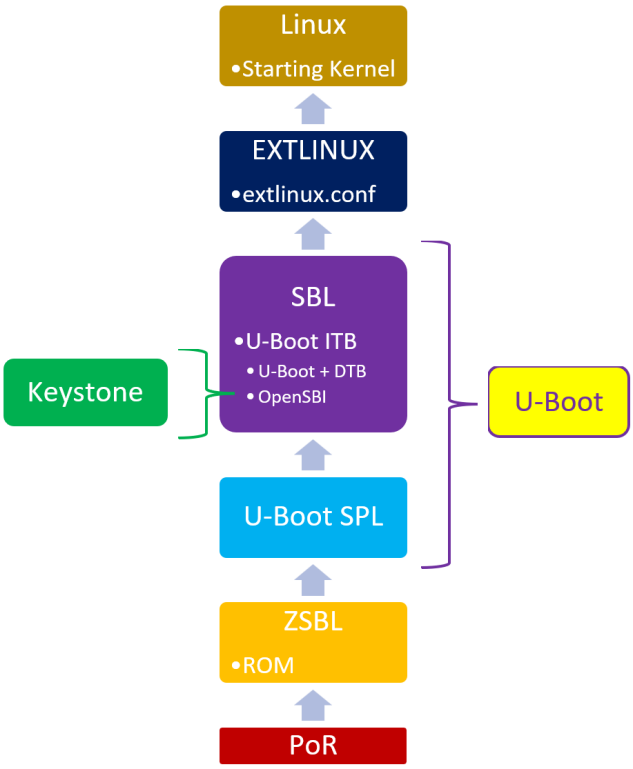
It follows that system service time R for enclaved computing can be determined by measuring the throughput of either the subject reference monitor Σ_2, or the object reference monitor Σ_3; these throughputs are labelled X_1, X_2, or X_3 accordingly. With known routing frequencies for q_{12} or q_{13}, the modified system flow-balance equations are solved and the system output rate X_0 is applied to calculate system service time R; assuming the remaining N and λ variables are known. For our computing system, the number of secure processes and number

of secure accesses are deterministic and based on the number of enclave processes and the processor frequency.

2.2 Keystone: an open-source secure enclave framework for RISC-V processors

The Keystone Enclave is designed for and built upon the RISC-V ISA. As an open-source, customizable TEE framework, it supports the four privilege modes outlined by "The RISC-V Instruction Set Manual, Volume II: Privileged Architecture:" U-mode (user) for applications, S-Mode (supervisor) for the kernel, H-mode (hypervisor) for hypervisor-level isolation, and M-mode (machine) for direct access to physical system resources. Keystone leverages RISC-V isolation primitives such as privilege modes and the PMP standard to enable isolation of memory-mapped control features *below* untrusted code (Lee et al., 2020).

We use the standardized boot flow, detailed in the "SiFive FU740-C000 Manual," for our system under test. This boot flow executes in the following order: (1) Power on Reset (PoR); (2) Zeroth Stage Bootloader (ZSBL) stored within Read Only Memory (ROM); (3) U-Boot Secondary Program Loader (SPL); (4) Second Bootloader (SBL) supporting (i) the U-Boot Image Tree Blob (ITB), (ii) Device Tree Blob (DTB) and (iii) OpenSBI; (5) EXTLINUX containing extlinux.conf; and (6) the Linux Kernel. Keystone operates on standard RISC-V cores and requires a hardware root of trust. Installation on the system is accomplished by installing Keystone as M-mode bootloader software (i.e., firmware) using an *out-of-tree* platform build configuration supported by OpenSBI. Figure 2 captures the described boot flow and provides context for how Keystone enforces resource accesses between secure and unsecure processes.



Once installed, the Keystone SM facilitates OS secure enclave requests by calling the appropriate Keystone Supervisor Binary Interface (SBI) functions described in Table 1. Keystone enclaves experience three distinct phases during their lifecycle: creation, execution, and destruction. Upon a *creation* request by the OS E_{OS}, the Keystone access controller $\sum_1$ measures the enclave memory, ensuring that the OS E_{OS} correctly loaded the enclave binaries into physical memory. Then, the access controller $\sum_1$ hashes the page contents and the virtual addresses along with configuration data to send to the Entity Monitor (EM) $\sum_3$. At *execution*, the access controller $\sum_1$ sets PMP entries and delegates control to the Resource Monitor (RM) $\sum_2$. After program completion, the runtime entity E_{RT} calls the *exit* function prompting the OS E_{OS} to initiate a *destruction* request. Upon a *destruction* call, the access controller $\sum_1$ clears the enclave memory region prior to returning the memory address space to the OS E_{OS} (Lee et al., 2020).

Table 1: Keystone SM SBI Functions

Caller	SBI Function	Description
E_{OS}	create	Validate & measure the enclave
	run	Start enclave & boot runtime
	resume	Resume enclave execution
	destroy	Clean & release enclave memory
E_{RT}	stop	Pause enclave execution
	exit	Terminate the enclave
	attest	Get signed attestation report
	random	Get secure random values

Due to the rapid adoption of the RISC-V ISA, expanding accessibility to RISC-V hardware, and maturing application development landscape, Keystone itself has undergone its first major version release. Keystone Enclave 1.0.0 (released on 2 March 2021) exclusively supports the RISC-V Open Source Serial Binary Interface (OpenSBI) which replaces the deprecated Berkley Boot Loader (BBL) to enforce a standardized RISC-V Boot Flow. Such changes offer opportunities to prove initial performance conclusions atop wider ranges of representative test platforms and use cases.

2.3 HiFive Unmatched by SiFive

We selected the HiFive Unmatched RISC-V Powered Linux Development Kit as our test platform of choice for four primary reasons: (a) form factor, (b) commodity hardware compatibility, (c) Linux support, and (d) enhanced System-on-a-Chip (SoC) monitoring. At the time of writing, the HiFive Unmatched is the only available RISC-V platform which supports the mini-Information Technology eXtended (ITX) form factor used by many x86_64 ISA systems. SiFive's decision to support this form factor enables flexible use of Peripheral Component Interconnect Express (PCIe) and Non-Volatile Memory Express (NVMe) hardware commonly found within the PC ecosystem. In practice, the native ability to run Ubuntu 21.04 (Hirsute Hippo) as the host OS simplifies testing procedures and provides a realistic environment suitable for benchmarking system performance. Aside from the form factor, hardware compatibility, and familiar OS, the Unmatched lends a unique perspective to TEE performance characterizations because it dedicates a fifth Freedom S7 monitor core exclusively for monitoring the four Freedom U74 cores. Precise SoC specifications are found in the FU740-C000 Manual, with our test configuration explicitly detailed in Section 3.

While other ASIC RISC-V systems offer similar or competing features, at present no other commercially available offering culminates all these features onto a single package. Moreover, company leadership at SiFive is comprised of three RISC-V ISA co-founders: Yunsup Lee, Chief Technology Officer; Krste Asanovic, Chief Architect, and Andrew Waterman, Chief Engineer. Their involvement at SiFive inspires confidence for long-term hardware support of SiFive products as RISC-V specifications mature.

2.4 Related work

Executing software with guaranteed integrity and confidentiality on a remote computing system, which is owned and maintained by an untrusted party, requires secure remote computation. Intel's Software Guard Extensions (SGX) aims to provide a trusted computing solution for x86 platforms by leveraging trusted hardware within the remote computer (Costan et al., 2017). Performance loss estimation studies have been explored which attempt to correlate SGX protections with performance degradation; however, the complex overlapping between security and performance factors is not bijective (Zheng et al., 2018). Accordingly, performance characterizations are often limited to time intensive trial-and-error analyses (Weichbrodt, et al., 2018).

Tullos et al. (2021) extended the Reference Monitor framework proposed by Gorbachov et al. (2019) to measure Keystone's system performance impact on FPGA softcore processor implementations. The applied Keystone SM framework describes the object reference monitor as the Resource Monitor (RM) and the subject reference monitor as the Entity Monitor (EM). The RM is responsible for ensuring system resource accesses adhere to privilege modes and PMP configurations, while the EM monitors and enforces authorized enclave usage by ensuring that TEEs are created, executed, and destroyed as authorized (Tullos et al., 2021). Figure 3 illustrates the adapted SM concept model and captures its conceptual operation. Our work supplements this Keystone SM framework (depicted in Figure 1) by comparably evaluating system performance impacts on RISC-V ASIC hardware across the enclave lifecycle –throughout *creation, execution*, and *destruction*.

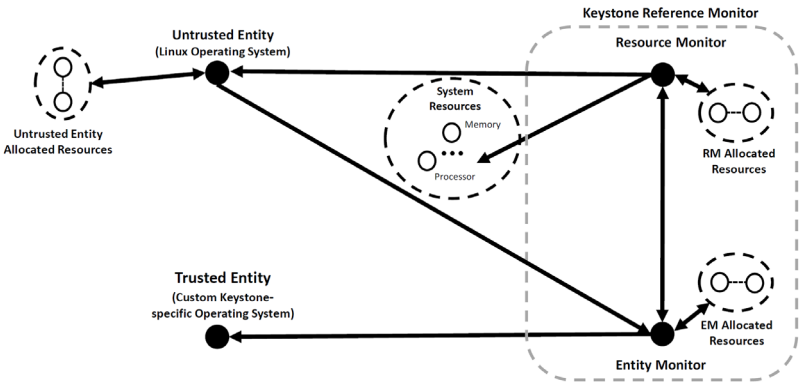


Figure 3: Keystone SM Concept Model (Tullos, et al., 2021)

3. Experimental design

3.1 Test platform configuration

The HiFive Unmatched SoC (FU740-C000) includes a 64-bit S7 monitor core and four 64-bit U74 application cores. The S7 monitor core has a dual-issue in-order execution pipeline supporting a peak sustained execution rate of two instructions per clock cycle (IPC). The monitor core supports M and U privilege modes for standard Multiply, Atomic, and Compressed RISC-V extensions (RV64IMAC). Each of the U74 application cores have dual-issue in-order execution pipelines supporting a peak sustained execution rate of two IPC per core. The application cores support M, S, and U privilege modes for standard Multiply, Single-Precision Floating Point, Double-Precision Floating Point, Atomic, and Compressed RISC-V extensions (RV64IMAFDC or RV64GC). Only the U74 application cores are Linux capable, although all five cores maintain a fully-coherent two Megabyte (2MB) shared level two (L2) cache. Assembly level software development for the S7 monitor core exceeds the scope of this proposal, although future exploration of its interoperability and monitoring capabilities warrants future study.

Our test system makes use of the NMVe M.2 2280 and 2230 form factors to utilize a Samsung 980 PRO PCIe 4.0 NVMe M.2 Solid-State Drive (SSD) for booting and an Intel Wireless-AC 9260 Wi-Fi card for internet access. Figure 4 captures our system under test as configured. The x16 PCIe Gen 3 Expansion Slot remains unused but can be configured to support select Advanced Micro Devices (AMD) RX 500-series or Radeon HD 6000-series graphical processing units (GPUs). Graphical capabilities and implications are left for future investigation.

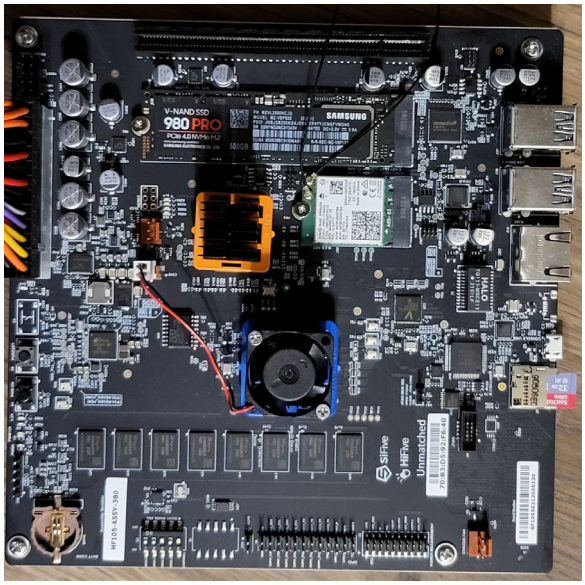


Figure 4: HiFive Unmatched Development Platform (test configuration with SSD and Wi-Fi cards installed)

Officially supported Linux distributions for RISC-V and for the Unmatched are currently limited. The distribution included on the provided microSD card equips users with an OpenEmbedded implementation; however, in order to better capture common usage scenarios, we selected Ubuntu 21.04 (Hirsute Hippo) as our host OS due to its

popularity and ample community support. Thanks to official support by Canonical, many conveniences of the existing Ubuntu ecosystem and the Debian Linux distribution are readily available in support of our specific hardware configuration –support for our desired software evaluation programs notwithstanding.

3.2 Benchmark suite configuration

We selected the Phoronix Test Suite (PTS) as our benchmarking platform during testing. The PTS offers a comprehensive testing and benchmarking experience while providing an extensible framework for adding custom benchmarking tests. From the Ubuntu terminal, the PTS enables effective, reproducible, and automated testing with an integrated remote management system to schedule, install, and archive system, test, and installation logs. Benchmark results are further configurable to automatically record to the OpenBenchmarking.org web viewer.

If desired, the PTS offers extensible markup language (XML) architecture the ability to run custom benchmarks and suites. Once added and installed, the PTS can be configured to automatically re-run tests if the results exceed a predefined standard deviation threshold. Without the pertinent need for a custom testing suite, we opted to generate preliminary benchmark results by using existing suites hosted by OpenBenchmarking.org.

3.3 Baseline performance characterization

For experimentation, we selected a compatible subset of the "Stress-NG" benchmarking suite for its intentional exercise of physical computer subsystems and of OS-to-Kernel interfaces. Openly published by Canonical, this suite encompasses over 260 stress tests exercising Central Processing Units (CPUs) and virtual memory operations among other subsystems. The "Stress-NG" Linux stress tool has been configured to execute 20 commonly used benchmarks on our Unmatched system running Ubuntu 21.04 with the "5.11.0-1020-gerneric (riscv64)" Kernel. These benchmarks were selected primarily because of their compatibility with the RISC-V ISA and for their common usage across numerous applications. Table 2 lists each benchmark selected from the suite to be executed, along with a brief description.

Table 2: Preliminary Benchmarking Metrics for Baseline Performance

	Benchmark	Description
0	Stress-NG 1.4.0	Linux Stress Test
1	MMAP	Memory Map
2	NUMA	Non-Uniform Memory Access
3	MEMFD	Anonymous Kernel Memory Management
4	Atomic	Atomic Operations
5	Crypto	MD5, SHA-256, SHA-512, scrypt, NT, yescrypt
6	Malloc	Memory Allocation
7	Forking	CPU Forking
8	IO_uring	Asynchronous Input / Output
9	SENDFILE	Read/Write
10	CPU Cache	Cache Thrashing
11	CPU Stress	Integer, Multiply, Floating Point, & Double Precision Operations
12	Semaphores	Shared Resources
13	Matrix Math	Two- & Three-Dimensional Matrix Operations
14	Vector Math	128-Bit Vector Operations
15	Memory Copying	memcpy() Method Operation
16	Socket Activity	IPv4, TCP Congestion Control
17	Context Switching	Memory Clobbering
18	Glibc C String Functions	Glibc C String Functions
19	Glibc Qsort Functions	Glibc Qsort Functions
20	System V Message Passing	System V Message Passing

4. Preliminary results and expectations

4.1 Baseline performance characterization

The RISC-V ISA, while maturing, is still new; compatible hardware optimized applications are not yet common. Unsurprisingly, our system benchmark results confirm this reality. For context, current generation Raspberry Pi 400 series devices frequently score double that of our HiFive Unmatched system. As an embedded development device, the ability to run Linux is impressive; however, much work remains for RISC-V adoption within the

datacenter and among consumers. Figure 5 details our precise testing configuration while Figures 6, 7, and 8 highlight baseline performance metrics and result confidence.

OpenBenchmarking.org	Phoronix Test Suite 10.6.0
SiFive RISC-V (4 Cores)	Processor
SiFive HiFive Unmatched A00	Motherboard
16GB	Memory
Samsung SSD 980 PRO 500GB + 32GB SD32G	Disk
Intel-AC 9260	Network
Ubuntu 21.04	OS
5.11.0-1020-generic (riscv64)	Kernel
X Server 1.20.11	Display Server
GCC 10.3.0	Compiler
ext4	File-System

Figure 5: Testing Configuration for Performance Characterizations

The baseline benchmark results charted below highlight the novelty of the RISC-V ISA and of supporting hardware. Benchmark scores are listed in operations per second, with higher scores indicating better performance. For context, at a comparable price point of $600, an ARM or x86 system would handily achieve scores in the millions compared to our hundreds or thousands.

211008-HFU-ICCWS-Baseline

ptsh	HiFive Unmatched - Ubuntu 21.04 - Stress-NG - No Keystone
stress-ng: MMAP	1.40
stress-ng: NUMA	14.46
stress-ng: MEMFD	6.78
stress-ng: Atomic	46477.32
stress-ng: Crypto	75.13
stress-ng: Malloc	1419228.21
stress-ng: Forking	2203.43
stress-ng: IO_uring	2359.91
stress-ng: SENDFILE	5816.28
stress-ng: CPU Cache	13.21
stress-ng: CPU Stress	169.61
stress-ng: Semaphores	110475.28
stress-ng: Matrix Math	518.02
stress-ng: Vector Math	364.58
stress-ng: Memory Copying	28.68
stress-ng: Socket Activity	174.78
stress-ng: Context Switching	98326.92
stress-ng: Glibc C String Functions	14647.47
stress-ng: Glibc Qsort Data Sorting	4.83
stress-ng: System V Message Passing	242686.42

OpenBenchmarking.org

Figure 6: Baseline Performance Results, Raw

By charting our results on a graph scaled for mature ISAs and hardware, it becomes apparent that RISC-V is still new. Currently, only memory allocation operations are comparable to traditional workstation performance.

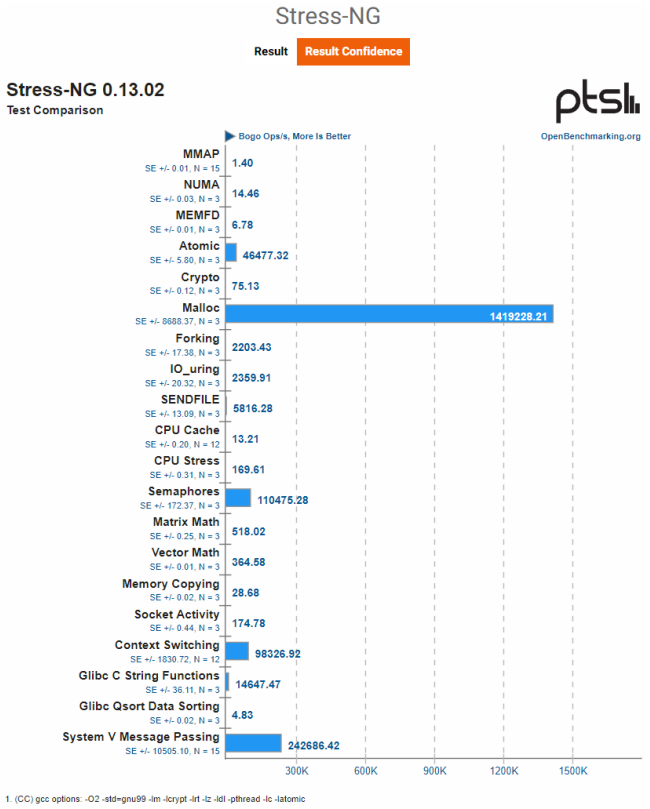


Figure 7: Baseline Performance Results, Charted

Confidence intervals for each benchmark are charted in Figure 8. and indicate that our results are consistent across repeated tests.

Figure 8: Baseline Performance Result Confidence, Charted

For further investigation, our complete benchmarking results are available online at the following Uniform Resource Locator (URL): https://openbenchmarking.org/result/2110085-EALD-211008H69.

5. Current and future work

5.1 Keystone performance characterization

The RISC-V security primitives applied by Keystone are unrivalled by any other framework. Unfortunately, at the time of writing, the required Keystone firmware configuration files are currently under development by the Keystone Enclave community to support the HiFive Unmatched. Enclave experimentation cannot be evaluated until either (a) a version of Keystone is released which supports our test hardware, or (b) we successfully implement these configuration files manually. The clear path forward is targeted firmware development, focused on enabling Keystone on the HiFive Unmatched while also promoting hardware-agnostic adoption.

5.2 Impacts of Keystone Enclave on system performance

Without a functional hardware implementation of the Keystone Enclave, concrete performance degradation conclusions have not yet been measured –especially for capturing performance implications related to context switching. Expected results will likely conclude that the performance overhead for enforcing Confidential Computing moderately degrades system performance. The cost of adding TEEs is significant; however, the degree to which degradation occurs during representative testing remains to be seen.

Disclaimer

The views expressed in this paper are those of the authors, and do not reflect the official policy or position of the United States Air Force, United States Space Force, Department of Defense, or the U.S. Government. This document has been approved for public release; distribution unlimited, case #88ABW-2021-0928.

References

Asanović, K. and Patterson, D.A. (2014) *Instruction Sets Should Be Free: The Case for RISC-V,* EECS Department, University of California, Berkeley, Tech. Rep. UCB/EECS-2014-146.

Costan, V., Lebedev, I. and Devadas, S. (2016) *Sanctum: Minimal hardware extensions for strong software isolation*, In 25th USENIX Security Symposium (USENIX Security 16), pp. 857-874.

Gorbachov, V., Batiaa, A.K., Ponomarenko, O. and Kotkova, O. (2019) *Impact Evaluation of Embedded Security Mechanisms on System Performance*, 2019 IEEE International Scientific-Practical Conference Problems of Infocommunications, Science and Technology, pp. 407-410.

Lee, D., Kohlbrenner, D., Shinde, S., Song, D. and Asanovic, K. (2020) *Keystone: An Open Framework for Architecting Trusted Execution Environments*, In Proceedings of the Fifteenth European Conference on Computer Systems, pp. 1-16.

SiFive FU740-C000 Manual v1p3, p. 68. [online]. Available at: https://sifive.cdn.prismic.io/sifive/de1491e5-077c-461d-9605-e8a0ce57337d_fu740-c000-manual-v1p3.pdf (Accessed 10 October 2021)

Tullos, J. C. (2021) *Characterizing Security Monitor and Embedded System Performance Across Distinct RISC-V IP-Cores*, *AFIT-ENG-MS-21-M-087*, p. 183. doi: 10.1109/TE.1962.4322266.

Waterman, A., Lee, Y., Avizienis, R., Patterson, D.A. and Asanovic, K. (2021) *The RISC-V Instruction Set Manual Volume II: Privileged Architecture, Document Version 20210921*, EECS Department, University of California, Berkeley, Tech. Rep. UCB/EECS-2016- 129.

Weichbrodt, N., Aublin, P.L. and Kapitza, R. (2018) *Sgx-perf: A performance analysis tool for Intel SGX enclaves*, Proceedings of the 19th International Middleware Conference, pp. 201-213.

Zheng, W., Cao, S., Gao, Z., Wu, X. and Ding, Q., (2018) *The Performance Evaluation Model of Intel SGX-Based Data Protection*, 2018 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computing, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation, pp. 1289-1292.