# Efficient Computing in a Safe Environment*

Michail Loukeris
Athens University of Economics and Business
Athens, Greece
loukerismichalis@gmail.com

## ABSTRACT

Modern computer systems are facing security challenges and thus are forced to employ various encryption, mitigation mechanisms, and other measures that affect significantly their performance. In this study, we aim to identify the energy and run-time performance implications of Meltdown and Spectre mitigation mechanisms. To achieve our goal, we experiment on server platform using different test cases. Our results highlight that request handling and memory operations are noticeably affected from mitigation mechanisms, both in terms of energy and run-time performance.

## CCS CONCEPTS

• **Security and privacy → Systems security**; • **Hardware → Power estimation and optimization**.

## KEYWORDS

energy consumption, security measures, safe environment

**ACM Reference Format:**
Michail Loukeris. 2019. Efficient Computing in a Safe Environment. In *Proceedings of the 27th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering (ESEC/FSE '19), August 26–30, 2019, Tallinn, Estonia.* ACM, New York, NY, USA, 3 pages. https://doi.org/10.1145/3338906.3342491

## 1 INTRODUCTION

Many mitigation patches and services have been introduced, targeting modern processors and kernels to shield them from vulnerabilities. However, the cost of protection is tightly associated with energy consumption and run-time performance tax. Nevertheless, disabling security mechanisms, to achieve performance gains, is feasible while performing computations in a safe environment such as non-cloud data center *e.g* Backup Data Center.

In this research, we focus to identify the energy and run-time performance of computer systems by dispensing measures that protect against malicious users such as Meltdown [9] and Spectre [6]. Meltdown and Spectre are security vulnerabilities aimed to steal currently processed data from modern processors. Prior works have investigated the Meltdown and Spectre and showed that they can degrade synthetic and realistic workloads run-time performance significantly [4, 11, 14]. However, protection measures that increase reliability but also guard against attacks, such as process separation [12] and array bounds checking [1], are retained.

To accomplish our task, we perform an empirical study on Meltdown and Spectre by utilizing a number of popular benchmarks. The obtained results highlight the energy and run-time performance impact of Spectre and Meltdown on different types of benchmarks.

## 2 METHODS

The goal of our study, is to evaluate the energy and run-time performance of different security measures to gain understanding over their impact on various application types. Therefore, we formulate our research questions as follow:

**RQ1.** *What are the energy and run-time performance implications of Meltdown and Spectre mitigation mechanisms?*

**RQ2.** *Which application type's energy and run-time performance are affected more from Meltdown and Spectre mitigation mechanisms?*

To setup our experiment we used the following subject systems:

**Experimental Platform:** We performed our experiments on a Lenovo ThinkCentre M910t [8] with Fedora 28 and kernel version 5.0.9-100. To retrieve energy consumption, we utilized an external device, the Watts Up? Pro (WUP) [21]. Also, we used a Raspberry Pi 3B to fetch the energy measurements from the WUP's internal memory via a Linux-based open source utility interface [2]. We followed this approach to avoid further overhead on the computer platform that could affect our energy measurements.

**Test Cases:** To investigate Spectre and Meltdown, we have selected benchmarks available from Phoronix [20]. Phoronix offers a computer system score by executing a number of collected, open source benchmarks; however, without giving the possibility to execute benchmark suites individually. Therefore, to obtain measurements for our test cases, we downloaded the desired benchmarks and wrote BASH scripts to execute them with the same parameters of Phoronix [7]. Specifically, we selected benchmarks to examine different functionalities of a computer system such as i) *Apache [16] and Nginx [18]* to handle client requests by using AB Apache command, ii) *OpenSSL* [15] to benchmark cryptographic algorithms, iii) *OSBench* [19] to examine operating system primitives such as creation of file, processes, and threads, iv) *CacheBench* [10] and *MCperf* [17] to investigate cache and main memory operations.

**Running Experiments:** Before starting our experiment we took a number of precautions to ensure the validity of our results. For instance, as suggested by Hindle [5], we shut down background processes and daemons found in modern operating systems. Also, we let a small time window of 30 seconds, between each test case, to avoid *power tails* [3] in our measurements. Moreover, we executed each test case 20 times to be statistically correct.

*The research described has been carried out as part of the CROSSMINER Project, which has received funding from the European Union's Horizon 2020 Research and Innovation Programme under grant agreement No. 732223

*ESEC/FSE '19, August 26–30, 2019, Tallinn, Estonia*

ACM ISBN 978-1-4503-5572-8/19/08.
https://doi.org/10.1145/3338906.3342491

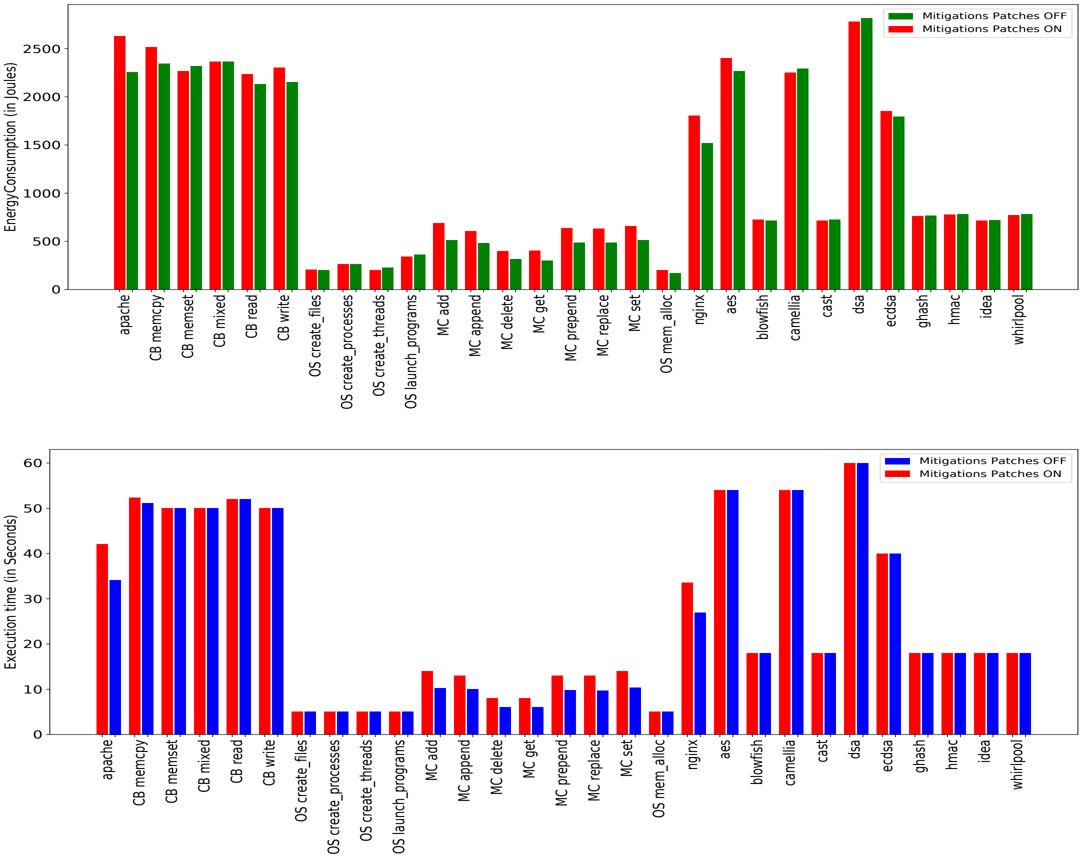

**Figure 1: Meltdown and Spectre benchmarks energy consumption and execution time**

## 3 RESULTS

By plotting histograms for each of the benchmarks' test case, we observed minor variations among their values. To this end, we decided to depict as results the mean values of energy consumption and run-time performance. Figure 1 illustrates the energy consumption in Joules (Top) and run-time performance impact in seconds (Bottom) of the tasks, respectively. The Figures X-axis tick labels with acronyms i) CB denotes the *CacheBench* tests, ii) OS indicates the *OSBench* applications, iii) MC shows the *MCPerf* tests, and iv) last ten refer to *OpenSSL* cryptograpihc aglorithm test cases.

For **RQ1**, we observe that the results show a significantly higher energy consumption and run-time performance overhead. Specifically, mitigation patches of Spectre and Meltdown may impact energy and run-time performance up to 26% and 27%, respectively. Likewise, for **RQ2** the benchmark types affected the most are the server benchmarks such as *Apache* and *Nginx* which were both affected by high energy consumption and run-time performance overhead. Moreover, a similar behavior is observed for memory-like operations, such as *memcpy, memset, read, write, add, append, replace, set* and *mem_alloc*. At this point, we should mention that the fact most of *OpenSSL* benchmarks don't seem to be affected by the mitigation patches, occurs because *OpenSSL* collects measurements of the maximum throughput in a given amount of time. By examining the cryptographic algorithms throughput without the application of mitigation patches, we experienced up to 17% of increased throughput. Therefore, for the given amount of time and energy, although the measurements look alike in Figure 1, the cryptographic algorithms without the mitigation patches processed more information. Moreover, we observe operations such as processes, files, and threads creation energy consumption where not affected or increased by dispensing the mitigation patches.

## 4 CONCLUSIONS AND FUTURE WORK

In this research, we found out that mitigation patches noticeably affect server systems and their memory operations, both in terms of energy and run-time performance. The results are very promising and, therefore, in the future we would like to further investigate the impact of more security measures such as TLS/SSL, *memory blanking*, *Zombieload, Fallout* and *Ridl* on more computer platforms in terms of performance and energy consumption.

As for future work, we aim to develop a requirements-driven adaptive software that takes into account the type of the application's main operations and switches on/off security measures accordingly [13].

# REFERENCES

[1] Periklis Akritidis, Manuel Costa, Miguel Castro, and Steven Hand. 2009. Baggy Bounds Checking: An Efficient and Backwards-Compatible Defense Against Out-of-Bounds Errors. (Aug. 2009).

[2] Peter Bailey. 2017. Watts Up Pro power meter interface utility for Linux. Retrieved 2018-10-23 from https://github.com/pyrovski/watts-up

[3] J. Bornholt, T. Mytkowicz, and K. S. McKinley. 2012. The model is not enough: Understanding energy consumption in mobile devices. In *2012 IEEE Hot Chips 24 Symposium (HCS)*. 1–3.

[4] Mark D. Hill, Jon Masters, Parthasarathy Ranganathan, Paul Turner, and John L. Hennessy. 2019. On the Spectre and Meltdown Processor Security Vulnerabilities. *IEEE Micro* PP (02 2019), 1–1. https://doi.org/10.1109/MM.2019.2897677

[5] Abram Hindle, Alex Wilson, Kent Rasmussen, E. Jed Barlow, Joshua Charles Campbell, and Stephen Romansky. 2014. GreenMiner: A Hardware Based Mining Software Repositories Software Energy Consumption Framework. In *Proceedings of the 11th Working Conference on Mining Software Repositories (MSR 2014)*. ACM, New York, NY, USA, 12–21.

[6] Paul Kocher, Jann Horn, Anders Fogh, and Daniel Genkin, Daniel Gruss, Werner Haas, Mike Hamburg, Moritz Lipp, Stefan Mangard, Thomas Prescher, Michael Schwarz, and Yuval Yarom. 2019. Spectre Attacks: Exploiting Speculative Execution. In *40th IEEE Symposium on Security and Privacy (S&P’19)*.

[7] Michel Larabel. 2019. Phoronix Test Suite. Retrieved 2019-05-15 from https://github.com/phoronix-test-suite/phoronix-test-suite

[8] lenovo thinkcentre. 2018. ThinkCentre M910 Tower | Power Your Business | Lenovo Australia. Retrieved 2018-05-22 from https://www3.lenovo.com/au/en/desktops-and-all-in-ones/thinkcentre/

[9] Moritz Lipp, Michael Schwarz, Daniel Gruss, Thomas Prescher, Werner Haas, Anders Fogh, Jann Horn, Stefan Mangard, Paul Kocher, Daniel Genkin, Yuval Yarom, and Mike Hamburg. 2018. Meltdown: Reading Kernel Memory from User Space. In *27th USENIX Security Symposium (USENIX Security 18)*.

[10] Philip J. Mucci. 2009. LLCbench - Low Level Architectural Characterization Benchmark Suite. Retrieved 2019-06-02 from http://icl.cs.utk.edu/llcbench/

[11] A. Prout, W. Arcand, D. Bestor, B. Bergeron, C. Byun, V. Gadepally, M. Houle, M. Hubbell, M. Jones, A. Klein, P. Michaleas, L. Milechin, J. Mullen, A. Rosa, S. Samsi, C. Yee, A. Reuther, and J. Kepner. 2018. Measuring the Impact of Spectre and Meltdown. In *2018 IEEE High Performance extreme Computing Conference (HPEC)*. https://doi.org/10.1109/HPEC.2018.8547554

[12] Niels Provos, Markus Friedl, and Peter Honeyman. 2003. Preventing Privilege Escalation. In *Proceedings of the 12th Conference on USENIX Security Symposium - Volume 12 (SSYM’03)*. USENIX Association, Berkeley, CA, USA.

[13] M. Salehie, L. Pasquale, I. Omoronyia, R. Ali, and B. Nuseibeh. 2012. Requirements-driven adaptive security: Protecting variable assets at runtime. In *2012 20th IEEE International Requirements Engineering Conference (RE)*. 111–120. https://doi.org/10.1109/RE.2012.6345794

[14] Nikolay A. Simakov, Martins D. Innus, Matthew D. Jones, Joseph P. White, Steven M. Gallo, Robert L. DeLeon, and Thomas R. Furlani. 2018. Effect of Meltdown and Spectre Patches on the Performance of HPC Applications. *arXiv:1801.04329 [cs]* (Jan. 2018). http://arxiv.org/abs/1801.04329 arXiv: 1801.04329.

[15] Phoronix Test Suite. 2018. OpenSSL [pts/openssl]. Retrieved 2019-06-02 from https://openbenchmarking.org/test/pts/openssl

[16] Phoronix Test Suite. 2019. Apache Benchmark [pts/apache]. Retrieved 2019-06-02 from https://openbenchmarking.org/test/pts/apache

[17] Phoronix Test Suite. 2019. Memcached mcperf [pts/mcperf]. Retrieved 2019-06-02 from https://openbenchmarking.org/test/pts/mcperf

[18] Phoronix Test Suite. 2019. NGINX Benchmark [pts/nginx]. Retrieved 2019-06-02 from https://openbenchmarking.org/test/pts/nginx

[19] Phoronix Test Suite. 2019. OSBench [pts/osbench]. Retrieved 2019-06-02 from https://openbenchmarking.org/test/pts/osbench

[20] Phoronix Test Suite. 2019. Phoronix Open-Source, Automated Benchmarking. Retrieved 2019-05-05 from https://www.phoronix-test-suite.com/

[21] WattsUpMeter. 2017. Watts up? Products: Meters. Retrieved 2018-01-15 from https://www.wattsupmeters.com/secure/products.php?pn=0