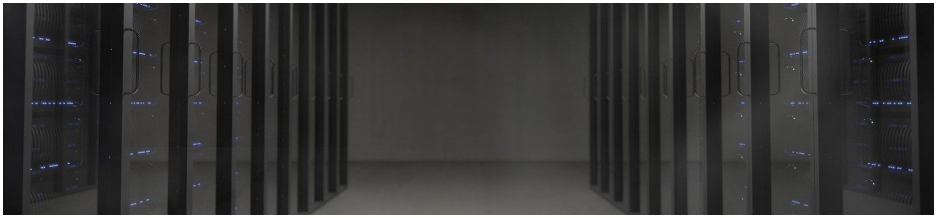
Dynamically Reconfiguring Hardware-Vulnerability Mitigations to Improve Energy Efficiency

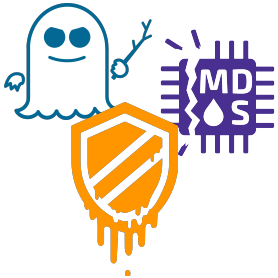
September 20, 2022
Fachgruppentreffen Betriebssysteme

Henriette Hofmeier, Benedict Herzog, Christian Eichler, Stefan Reif, Luis Gerhorst and Timo Hönig

Ruhr-Universität Bochum (RUB)
Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU)

The Price of Hardware-Vulnerability Mitigations





Hardware Vulnerabilities & Mitigations

Spectre

- ⇒ leak of speculatively accessed data
- STOP restriction of speculation

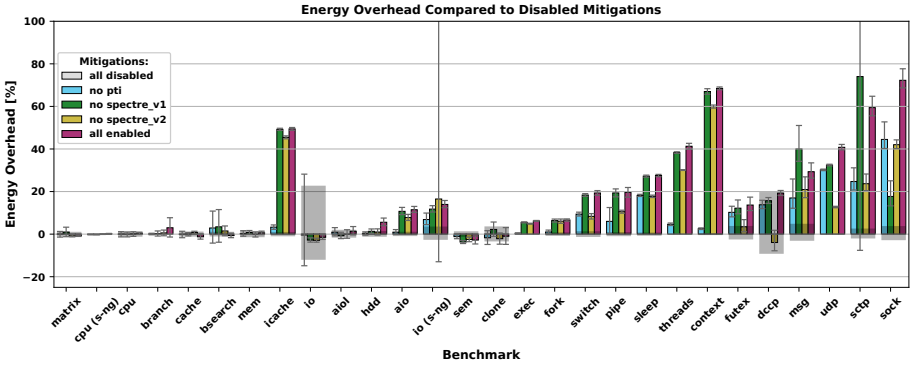
Meltdown

- ⇒ leak of speculatively accessed kernel data
- STOP Kernel Page Table Isolation (KPTI)

Microarchitectural Data Sampling (MDS)

- ⇒ sampling of data from shared on-core resources
- STOP disabling Simultaneous Multithreading (SMT), Core Scheduling

The Price of Spectre and Meltdown Mitigations

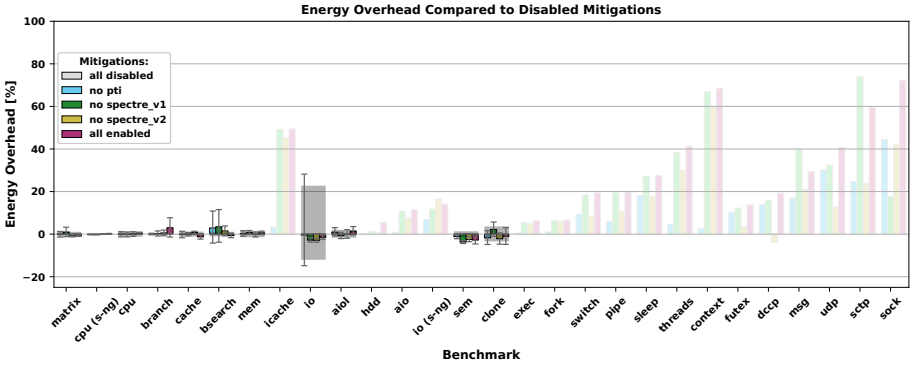


Benedict Herzog et al.

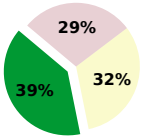
"The Price of Meltdown and Spectre: Energy Overhead of Mitigations at Operating System Level."

In: *Proceedings of the 14th European Workshop on Systems Security (EuroSec '21).*

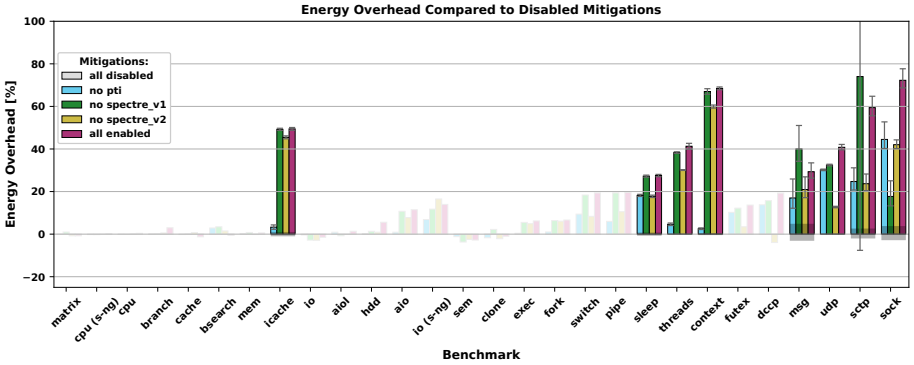
The Price of Spectre and Meltdown Mitigations



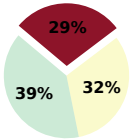
→ 11 out of 28 benchmarks have an overhead below 5 %



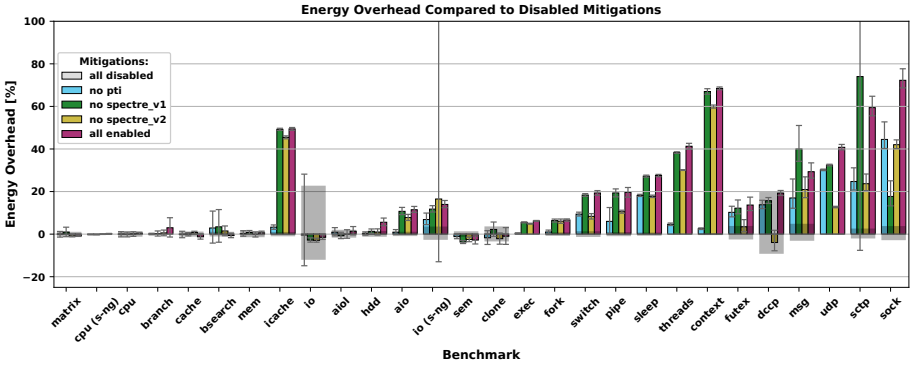
The Price of Spectre and Meltdown Mitigations



→ 8 out of 28 benchmarks have an overhead above 25 %

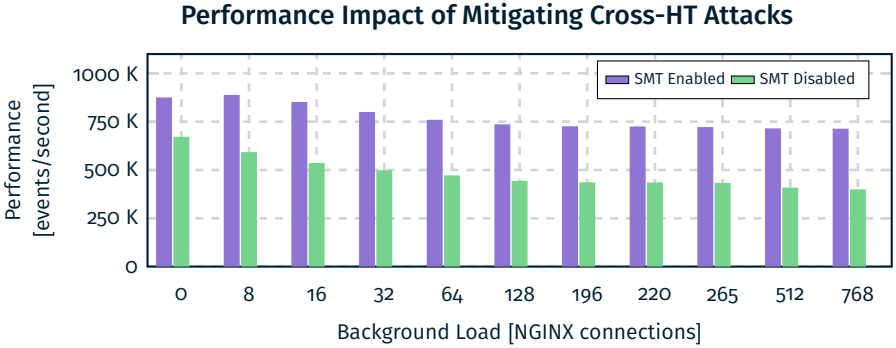


The Price of Spectre and Meltdown Mitigations



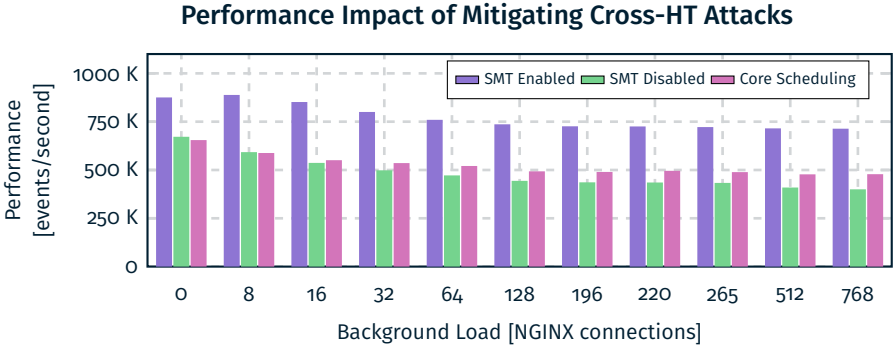
The overhead is highly application-dependent and ranges from ~0 % to 72 %.

The Price of Cross-HT Attack Mitigations



→ **disabling SMT incurs high performance penalties of up to 44 %**

The Price of Cross-HT Attack Mitigations



→ **Core Scheduling reduces performance penalties under high system load**

Potential of Customization and Reconfiguration

Customization

- adapt to application requirements
- adapt to current system state
- adapt to hardware

Reconfiguration

- adapt dynamically at run time
- avoid unnecessary overhead

optimize the system's energy efficiency &
utilize security features for improvements

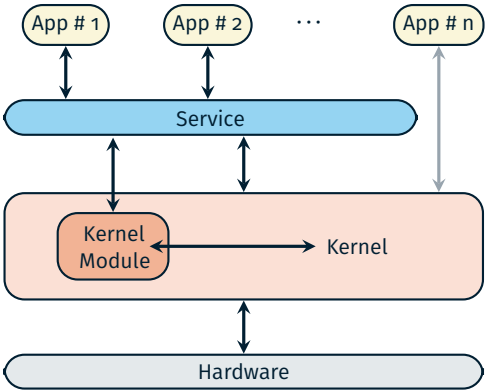
Dynamic Hardware-Vulnerability Mitigation Reconfigurations

Reconfiguration

- based on system state and application requirements
- update on any state changes

Kernel Reconfigurability

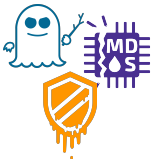
- information on vulnerabilities and available mitigations
- reconfiguration interface
- code-patching initiation

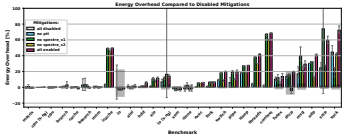


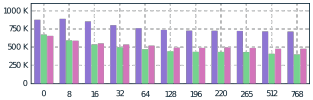
State of Work

First Results

- extensive evaluation of Spectre and Meltdown mitigations
- fully functional reconfiguration implementation
  → minimal performance overhead







Next Steps

- extensive evaluation of Core Scheduling
- continuously adapt system to current developments
- extension of system to include further mitigations

References (1)

[1] Core scheduling – the linux kernel documentation (version 5.14).
[Online] last accessed 2021-06-10.

[2] Mds - microarchitectural data sampling – the linux kernel documentation (version 5.14).
[Online] last accessed 2021-06-08.

[3] Page table isolation (pti) – the linux kernel documentation (version 5.14).
[Online] last accessed 2022-06-07.

[4] Spectre side channels – the linux kernel documentation (version 5.14).
[Online] last accessed 2021-10-13.

[5] Static keys – the linux kernel documentation (version 5.14).
[Online] last accessed 2022-05-23.

[6] Enrico Barberis, Pietro Frigo, Marius Muench, Herbert Bos, and Cristiano Giuffrida.
Branch history injection: On the effectiveness of hardware mitigations against Cross-Privilege spectre-v2 attacks.
In *Proceedings of the 31st USENIX Security Symposium (USENIX Security '22)*, 2022.
Prepublication.

[7] Claudio Canella, Daniel Genkin, Lukas Giner, Daniel Gruss, Moritz Lipp, Marina Minkin, Daniel Moghimi, Frank Piessens, Michael Schwarz, Berk Sunar, Jo Van Bulck, and Yuval Yarom.
Fallout: Leaking data on meltdown-resistant cpus.
In *Proceedings of the 26th ACM SIGSAC Conference on Computer and Communications Security (CCS '19)*, pages 769–784, 2019.

[8] S.J. Eggers, J.S. Emer, H.M. Levy, J.L. Lo, R.L. Stamm, and D.M. Tullsen.
Simultaneous multithreading: a platform for next-generation processors.
IEEE Micro, 17(5):12–19, 1997.

References (2)

[9] Daniel Gruss, Moritz Lipp, Michael Schwarz, Richard Fellner, Clémentine Maurice, and Stefan Mangard.
Kaslr is dead: Long live kaslr.
In Eric Bodden, Mathias Payer, and Elias Athanasopoulos, editors, *Proceedings of the 9th International Symposium on Engineering Secure Software and Systems (ESSoS '17)*, pages 161–176, 2017.

[10] Dave Hansen.
Lkml: [patch 00/23] kaiser: unmap most of the kernel from userspace page tables.
[Online] last accessed 2022-05-22.

[11] John L Hennessy and David A Patterson.
Computer Architecture – A Quantitative Approach.
Morgan Kaufmann, 5 edition, 2012.

[12] John L Hennessy and David A Patterson.
Computer Organization and Design – The Hardware / Software Interface.
Morgan Kaufmann, 5 edition, 2014.

[13] Benedict Herzog, Stefan Reif, Julian Preis, Wolfgang Schröder-Preikschat, and Timo Hönig.
The price of meltdown and spectre: Energy overhead of mitigations at operating system level.
In *Proceedings of the 14th European Workshop on Systems Security (EuroSec '21)*, pages 8–14, 2021.

[14] Paul Kocher, Jann Horn, Anders Fogh, Daniel Genkin, Daniel Gruss, Werner Haas, Mike Hamburg, Moritz Lipp, Stefan Mangard, Thomas Prescher, Michael Schwarz, and Yuval Yarom.
Spectre attacks: Exploiting speculative execution.
In *Proceedings of the 40th IEEE Symposium on Security and Privacy (SP '19)*, pages 1–19, 2019.

References (3)

[15] 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.
Meltdown: Reading kernel memory from user space.
In *Proceedings of the 27th USENIX Security Symposium (USENIX Security '18)*, pages 973–990, 2018.

[16] Michael Schwarz, Moritz Lipp, Daniel Moghimi, Jo Van Bulck, Julian Stecklina, Thomas Prescher, and Daniel Gruss.
Zombieload: Cross-privilege-boundary data sampling.
In *Proceedings of the 26th ACM SIGSAC Conference on Computer and Communications Security (CCS '19)*, pages 753–768, 2019.

[17] Dean M. Tullsen, Susan J. Eggers, and Henry M. Levy.
Simultaneous multithreading: Maximizing on-chip parallelism.
ACM SIGARCH Computer Architecture News, 23(2):392–403, 1995.

[18] Stephan van Schaik, Alyssa Milburn, Sebastian Österlund, Pietro Frigo, Giorgi Maisuradze, Kaveh Razavi, Herbert Bos, and Cristiano Giuffrida.
Ridl: Rogue in-flight data load.
In *Proceedings of the 40th IEEE Symposium on Security and Privacy (SP '19)*, pages 88–105, 2019.