Mitigating Transient-Execution Attacks with CHERI Compartments

Franz A. Fuchs, Jonathan Woodruff, Peter Rugg, and Simon W. Moore*

Department of Computer Science and Technology, University of Cambridge

Abstract

We propose to use and extend CHERI compartments in order to mitigate transient-execution attacks. These attacks have presented a major threat in recent years, driving deployment of software mitigations and research into hardware solutions. We envision that the additional state provided by CHERI capabilities and compartments presents a viable way to solve transient-execution attacks on an architectural level. In this poster, we want to show how we can extend and use the CHERI protection model for that.

Introduction

Transient-execution attacks have greatly impacted the computer security landscape. Transient-execution attacks aim to encode secrets in microarchitectural state changes, which are then made visible via side channels. One example is the Spectre v1 attack [1], which mistrains a branch direction predictor and then leaks a speculatively accessed value. This can used to facilitate speculative out-of-bounds access to secrets.

CHERI (Capability Hardware Enhanced RISC Instructions) extends instruction sets with unforgeable bounded pointers called *capabilities*[2]. Capabilities not only *describe* a region of address space, but also *authorise* access according to the permissions field. A capability consists of the pointer itself, metadata, and a one bit validity tag. CHERI guarantees security for spatial and temporal memory safety as well as offering fine-grained memory protection.

Transient Execution and CHERI

Previous work [3] has shown that a superscalar, out-of-order CHERI-RISC-V processor is vulnerable to transient-execution attacks. The results of [3] are depicted in Table 1. They show that the processor under test mitigated Spectre-PHT and the two applicable Meltdown-style attacks, but could not mitigate the remainder of the Spectre-style attacks. While the authors [3] realise that the capability configuration needs to be precise and correct in order for CHERI microarchitectures to successfully mitigate the above named subset of transient-execution attacks, the authors do take into account that the CHERI ISA does not provide architectural guarantees for speculation. At the current state, CHERI-RISC-V processors *can* mitigate Spectre-PHT and Meltdown-like attacks, but there is no architectural obligation to enforce that. Any other CHERI-RISC-V microarchitecture could choose to speculate more freely than our processor under test and therefore be vulnerable to these attacks.

	Legacy CHERI-RISC-V
Spectre-PHT	*Safe*
Spectre-BTB	*Vulnerable*
Spectre-RSB	*Vulnerable*
Spectre-STL	*Vulnerable*
Meltdown-US-CHERI	*Safe*
Meltdown-GP-CHERI	*Safe*

Table 1: *Results obtained by [3]. The table shows an overview of attempted transient-execution attacks on an out-of-order CHERI-RISC-V processor, and whether this processor is safe or vulnerable against an attack.*

In general, there is no clarity on what microarchitectures should allow and forbid from happening. Many transient-execution attacks and defence mechanisms come with different expectations and threat models [4]. While one could constrain many ways in which microarchitectures speculate and operate in general, this would not be an expedient measure: one would overconstrain the architecture. Instead, industry has chosen to enforce protections on a *context* switch [5].

Compartmentalisation

Compartmentalisation is a form of privilege decomposition increasingly used in secure systems. Splitting programs into different processes is a well-known and predominantly used compartmentalisation technique. However, this approach is very coarse grained and can hardly be applied for smaller protection units. Not only are the transient vulnerability security guarantees insufficient, but also process switching is conducted by the operating system and thus leads to a big cycle overhead. Having small protection units in every part

*Corresponding author: franz.fuchs@cl.cam.ac.uk

of a system and the ability to performantly switch between them is essential for the transient-execution security of modern systems.

CHERI Compartments

Therefore – as opposed to the coarse-grained industry approach – we are leveraging CHERI compartments to mitigate transient-execution attacks. CHERI offers fine-grained compartments via various different capability mechanisms, e.g., capability *sealing* and *invocation*[2]. Compartments in CHERI can be as small as simple functions in C and therefore can give high security guarantees for small units of protection. CHERI compartments offer an increase of robustness compared to process-based compartmentalisation due to the domain transition mechanisms baked into the architecture. This offers protection on every transition where previously there has been no protection. Software compartmentalisation is an ongoing research effort and allows for multiple different compartmentalisation models.

In order to fulfil the security guarantees needed by any software compartmentalisation model, processors must protect the microarchitectural state of one compartment from other compartments in the same system. Therefore, microarchitectures need to decide which state they can share between compartments. We do not propose to close the side-channel leaking a secret to an attacker, but rather the source of these secrets, which are based in observing microarchitectural state not belonging to the currently executing compartment. This requires a full microarchitectural audit, which can be carried out automatically or by hand. Furthermore, we distinguish between long-term microarchitectural state and transient state. Long-term microarchitectural state are buffers that are kept around for many cycles and mostly store prediction information that will be used long-ahead in the future, e.g., a branch target buffer (BTB). On the other hand, transient state is short-lived, e.g., a microarchitectural queue of a few entries. Both forms of microarchitectural state can be effectively used by attackers to leak secrets.

Mitigating Transient-Execution Attacks

We envision CHERI compartments to be a robust solution to mitigate transient-execution attacks. While process-level security manages to defend against Spectre-BTB and Spectre-RSB, these measures remain on a coarse level. Processes are usually running large-sized code, which often allows for lucrative intra-domain transient-execution attacks. Furthermore, process-level security does not offer mitigation for any other than the above mentioned Spectre-like attacks or for Meltdown-like attacks at all.

However, CHERI compartments will not only be able to mitigate Spectre-BTB and Spectre-RSB, but also Spectre-PHT in all cases. The notion of CHERI capabilities enables us to enumerate the reachable memory, which is described by the transitive closure over the entire set of currently accessible capabilities. Hardware can safely use approximations of the maximum reachable memory, e.g., only the capabilities in the current architectural register state without any capabilities in memory, in order to implement protection mechanisms. Capability information being available to hardware helps to prevent illegal speculative actions from happening in the microarchitecture. In this way, CHERI compartments will also defend against Meltdown-style attacks.

A key point of developing CHERI was that it not only defends against today's attacks, but also mitigates future attacks. This will play an even more important role for transient-execution attack scenarios. CHERI capabilities allow hardware to know of the architecturally granted privileges and CHERI compartments allow robust enforcement against transient-execution attacks.

Acknowledgements

This work was supported by the Engineering and Physical Sciences Research Council EP/S030867/1. We would like to thank the entire CHERI team for their contributions to the project.

References

[1] Paul Kocher et al. "Spectre Attacks: Exploiting Speculative Execution". In: *2019 IEEE Symposium on Security and Privacy*. IEEE Computer Society, 2019, pp. 1–19. DOI: 10.1109/SP.2019.00002. URL: https://doi.org/10.1109/SP.2019.00002.

[2] Robert N. M. Watson et al. *Capability Hardware Enhanced RISC Instructions: CHERI Instruction-Set Architecture (Version 8)*. Tech. rep. UCAM-CL-TR-951. University of Cambridge, Computer Laboratory, Oct. 2020. URL: https://www.cl.cam.ac.uk/techreports/UCAM-CL-TR-951.pdf.

[3] Franz A. Fuchs et al. "Developing a Test Suite for Transient-Execution Attacks on RISC-V and CHERI-RISC-V". In: *Workshop on Computer Architecture Research with RISC-V*. 2021.

[4] Claudio Canella et al. "A Systematic Evaluation of Transient Execution Attacks and Defenses". In: *Proceedings of the 28th USENIX Conference on Security Symposium*. SEC'19. Santa Clara, CA, USA: USENIX Association, Aug. 2019, pp. 249–266. ISBN: 9781939133069.

[5] *Arm v8.5-A/v9 CPU updates*. https://developer.arm.com/documentation/102822/0105. May 2021.