Talk: Transient-execution attacks on the CHERI Morello platform

Jacqueline Henes[1], Marius Muench[1], David Oswald[1], and Hany Ragab[2]

[1] University of Birmingham, UK
{jkh703,m.muench,d.f.oswald}@bham.ac.uk
[2] Vrije Universiteit Amsterdam, The Netherlands
hany.ragab@vu.nl

1 Introduction

Capability Hardware Enhanced RISC Instructions (CHERI) is an Instruction Set Architecture (ISA) extension providing spatial memory protection and compartmentalization [7]. CHERI protections hold promise in securing computer systems from common memory safety bugs, but as the implementations continue to mature, we investigate whether the security guarantees offered by CHERI at the architectural level could also hold at the microarchitectural level. One class of microarchitectural attacks that affect superscalar and out-of-order processors are transient execution attacks such as Spectre and Meltdown [6]. These attacks can break the architectural security domain isolation model at the microarchitectural level and leak sensitive information by exploiting microarchitecural design optimizations, such as speculative/out-of-order execution, branch predictors, and cache timing side channels.

In this work, we explore whether the hardware implementation of the CHERI paradigm mitigates or exacerbates transient execution attacks, focusing on CHERI's latest prototype implementation on the ARM architecture, the ARMv8-A Morello platform [1]. We have reproduced Spectre-PHT and Spectre-BTB on the Morello platform using CHERI-aware C/C++ code. Building upon our baseline work we then also test variations targeting CHERI-specific operations in speculation. We discuss what the results of both the baseline tests and their CHERI variations mean in relation to the Morello implementation of the CHERI model.

2 Background and Related Work

CHERI extends traditional ISAs, introducing the concept of pointer capabilities, that is, the properties of a pointer that define its bounds and permissions to the pointed memory region [7]. Capabilities maintain key properties that result in efficient and granular memory protection at an architectural level. CHERI's

This research was supported by the UK Engineering and Physical Sciences Research Council (EPSRC) under grant EP/V000454/1, funded by the DSbD (Digital Security by Design) Programme delivered by UKRI to support the DSbD ecosystem.

overall goal is to provide spatial memory protection against common memory bugs such as buffer overflows. The use of CHERI's protection model has since broadened to include temporal memory safety as well as performant software compartmentalization, with functional prototypes implementations already implemented in CHERI systems [2].

CHERI capabilities extend hardware memory addressing primitives with extra metadata. Each capability is twice the width of the native architecture's integer pointer type, and also are associated with a 1-bit validity "tag". Half of the capability contains a full-precision integer address, while the remaining half contains metadata representing the level of access the capability has. This includes a representation of the lower and upper bounds of memory to which the capability has authority, as well as a permissions mask describing valid operations for this memory area. Capability tags are stored out of band separately from their associated capability. A capability can only be used if its tag bit is valid; illegal modifications will clear this tag bit, marking this capability as invalid. Invalid capabilities cannot be dereferenced, instead causing a hardware trap. Our target CHERI implementation in this work is the Morello platform, which represents the latest hardware implementation of the CHERI paradigm. It is based on an out-of-order superscalar Armv8-A implementation of the Neoverse N1 microarchitecture [1].

Transient Execution Attacks on CHERI-RISC-V The publication of Meltdown and Spectre [6] has sparked discussion and work on how CHERI architectures are affected by microarchitecture-targeted attacks. Watson et al. [8] published a technical report focusing on the impact of transient-execution attacks on CHERI architectures. The authors discuss whether CHERI would be vulnerable to Spectre and Meltdown, and how CHERI could also prevent transient-execution attacks. In particular, bounds enforcement and data cache loads in CHERI are assumed to be sound under speculation, as long as the microarchitecture ensures that any speculative memory access also undergoes the associated capability checks. A highlighted concern is the potential speculation of capability values such as bounds and permissions, which could violate assumed invariants. Further on CHERI and transient-execution attacks, Fuchs et al. [3] created a test suite for testing current speculative execution attacks on RISC-V and CHERI-RISC-V. As part of their work, they identified major applicable transient-execution attacks on RISC-V architectures and reproduced this on the CHERI RiscyOO out-of-order superscalar processor using both RISC-V and CHERI-RISC-V assembly. The two variants of Meltdown tested were unsuccessful on both RISC-V and CHERI-RISC-V. In contrast, all four Spectre attacks that were tested were successful on RISC-V. CHERI-RISC-V could mitigate Spectre-PHT exploits as long as appropriate bounds were used. Following on from this, Fuchs et al. [4] propose and evaluate *Capability Speculation Contracts* as a mitigation for transient-execution. This work also classifies a new type of transient-execution attack specific to CHERI: Meltdown-CF (Capability Forgery).

3 Experimental Setup

We targeted the Arm Morello platform as one of the major CHERI implementations that also supported a developing CHERI-enabled software stack. CHERI as an academic and industrial project resulted in variety of usable yet volatile software development environments. As a result, we faced design challenges unique to evaluating CHERI software. We tested our exploits on CheriBSD, which is an extension of FreeBSD that implements full support of CHERI capabilities into a conventional operating system [2]. Other OSs that support capabilities do exist; however, CheriBSD is the OS with the most comprehensive capability support. The CHERI software stack offers two operating modes; pure and hybrid. Pure-capability (*purecap*) mode programs are configured to have all pointers with CHERI capabilities. Hybrid mode provides a more incremental approach to integrating CHERI protection as it allows programmers to explicitly declare which pointers are compiled as capabilities. The hardware architecture supports toggling capability enforcement on and off when changing exception level to support this behaviour. While we compile proof-of-concept code for both modes, we specifically target purecap binaries for our experiments.

4 Preliminary results

We have successfully reproduced the two original Spectre variants, Spectre-PHT (Pattern History Table) v1 and Spectre-BTB (Branch Target Buffer) v2, in both hybrid and purecap mode running on CheriBSD on Morello. As a starting point, we use BranchDifferent's [5] implementation of Spectre-PHT and Spectre-BTB attacks on ARMv8-based Apple silicon. For our timing primitive, we used the ARMv8 system control register `CNTVCT_EL0` which is accessible from userspace. On the Morello platform, the `DC CIVAC` cache maintenance instruction works successfully as part of a Flush+Reload [9] cache side-channel attack. For each Spectre variant, we first checked that the exploit worked in hybrid mode, and then went on to test the exploit in purecap mode. This is not only because hybrid compilation is intentionally trivial, but purecap compilation often requires CHERI-specific changes. This also allows us to have a working baseline with which we compare the purecap mode exploits. With this baseline we can isolate potential CHERI-specific issues or effects.

Spectre-PHT exploits the pattern history table to trick the CPU into transiently executing a specific conditional branch. This can, for example, cause the CPU to speculatively dereference a pointer, leaking sensitive information into the cache. In CHERI purecap, all pointers are now capabilities, with bounds and permissions. In our naive purecap implementation of Spectre-PHT, the sensitive information is included in this capability's bounds. This implementation allows for a successful recovery of sensitive information. However, modifying the code to have more appropriate bounds and, therefore, *better take advantage* of the CHERI capability protection gives different results. In our modified CHERI

Spectre-PHT exploit, before mistraining we can narrow the bounds to explicitly exclude this sensitive information. The data is still in memory where it was originally located, but any attempts to architecturally access said data with the narrowed capability will cause a capability fault, in line with CHERI protection principles. We found that these narrowed bounds *also* prevent speculative access to leak the secret since it does not trigger any capability fault, but the secret was not observed in the cache side-channel. The decreased bounds of the capability appear to be reflected in speculative path: either no memory access to the secret data occurs, or if it does happen it is not encoded via the cache side-channel. In either case, it appears that setting more strict, appropriate bounds is a successful mitigation against this style of Spectre-PHT attack.

Spectre-PHT PoC analysis We further develop our evaluation by testing the limits of CHERI capabilities on speculative execution paths, focusing on Spectre-PHT with this CHERI mitigation. In particular, seeing that CHERI operations have meaningful effects in the microarchitecture, the question then becomes whether CHERI operations can be used maliciously in speculation. We detail particularly interesting preliminary results using our Spectre-PHT PoC below:

Transiently increasing/decreasing bounds Attempting to increase capability bounds during speculation, and then access previously out-of-bounds memory with this same capability does not successfully leak out-of-bounds data. However, doing the inverse where during speculation we *decrease* the bounds, we observe that the recovered data always mirrors the speculative bounds change.
Using invalid capabilities If invalid capabilities or illegal capability operations can be done during speculation to leak information (as in Spectre-PHT), this would not only violate CHERI's protection model but also create avenues for leaking data from higher privileged memory areas. In our tests to date, this style of exploit has been unsuccessful at leaking any sensitive data or otherwise breaking existing security barriers.
Speculating upon capability values A specific corner-case we found used an uninitialised capability. In the normal case, during a conditional check that is only true during mistraining and *not* the attack, a capability is initialised so that an ensuing access does not dereference a null pointer. However, this variant resulted in a successful recovery of the secret, which indicates that during speculation the microarchitecture incorrectly performs this capability assignment.

Spectre-BTB works by mistraining the branch target buffer to divert program flow in speculation. We found that Spectre-BTB is successful in both hybrid and purecap mode - a mispredicted target branch can leak sensitive information via a victim pointer/capability. CHERI broadly governs memory access, not branching logic, so implementing CHERI has little effect on Spectre-BTB.

5 Discussion and Future Work

From our current results, we find that at least on the Morello platform, CHERI protection principles are also enforced microarchitecturally to some extent. We can see this most clearly when experimenting with the bounds in our Spectre-PHT test: any reduction in bounds is reflected in the recovered data but attempts to increase bounds prevents data leakage. Spectre-BTB being successful leads us to believe that the BTB on Morello works similarly to its CHERI-RISC-V counterpart: the BTB stores the whole capability, and so a given entry not only has the address but all capability privileges associated with it. Consequently, a branch target misprediction will pass on these privileges arbitrarily. On the RiscyOO processor, the return stack buffer implementation for CHERI-RISC-V [8] is also vulnerable to Spectre-RSB attacks for the same reason, as it stores entire capabilities. On this basis, we tentatively assume that Spectre-RSB attacks may be possible on the Morello platform, although we have yet to confirm this.

From our tests, transient-execution attacks that try to modify or otherwise break capability protections are not successful. Attacks that rely on speculating upon capability values themselves (such as Spectre-BTB) however may be more likely to work. Future work includes more in-depth testing of more Spectre variants, as well as reproducing Meltdown exploits.

References

1. Arm Ltd: Arm Morello Program, https://www.arm.com/architecture/cpu/morello
2. CTSRD, University of Cambridge: CheriBSD, https://www.cl.cam.ac.uk/research/security/ctsrd/cheri/cheribsd.html
3. Fuchs, F.A., Woodruff, J., Moore, S.W., Neumann, P.G., Watson, R.N.M.: Developing a Test Suite for Transient-Execution Attacks on RISC-V and CHERI-RISC-V. In: Computer Architecture with RISC-V workshop (CARRV) (2021)
4. Fuchs, F.A., Woodruff, J., Rugg, P., Joannou, A., Clarke, J., Baldwin, J., Davis, B., Neumann, P.G., Watson, R.N.M., Moore, S.W.: Safe Speculation for CHERI. In: IEEE 42nd International Conference on Computer Design (ICCD) (Nov 2024)
5. Hetterich, L., Schwarz, M.: Branch Different - Spectre Attacks on Apple Silicon. In: Detection of Intrusions and Malware, and Vulnerability Assessment (DIMVA'22)
6. Kocher, P., Horn, J., Fogh, A., Genkin, a.D., Gruss, D., Haas, W., Hamburg, M., Lipp, M., Mangard, S., Prescher, T., Schwarz, M., Yarom, Y.: Spectre Attacks: Exploiting Speculative Execution. In: 40th IEEE Symposium on Security and Privacy (S&P'19)
7. Watson, R.N.M., Neumann, P.G., Woodruff, J., Roe, M., Almatary, H., Anderson, J., Baldwin, J., Barnes, G., Chisnall, D., Clarke, J., Davis, B., Eisen, L., Filardo, N.W., Fuchs, F.A., Grisenthwaite, R., Joannou, A., Laurie, B., Markettos, A.T., Moore, S.W., Murdoch, S.J., Nienhuis, K., Norton, R., Richardson, A., Rugg, P., Sewell, P., Son, S., Xia, H.: CHERI ISA (Version 9). Tech. rep., CTSRD, University of Cambridge (2023), https://www.cl.cam.ac.uk/techreports/UCAM-CL-TR-987.html
8. Watson, R.N.M., Woodruff, J., Roe, M., Moore, S.W., Neumann, P.G.: CHERI: Notes on the Meltdown and Spectre Attacks. Tech. rep., CTSRD, University of Cambridge (2018)
9. Yarom, Y., Falkner, K.: FLUSH+RELOAD: a High Resolution, Low Noise, L3 Cache Side-Channel Attack. In: USENIX Security (2014)