Microarchitectural Side Channels SIG Status Review

Presented by: Allison Randal and Ronan Lashermes, Chairs

Tech Chairs Review

Scope

The microarchitecture may be vulnerable to information leakage, notably through resource sharing. Via side-channel leakage where the victim application, by modifying any state in the processor, may expose secret information when the attacker can observe state changes. Or via covert-channel leakage where the attacker tries to actively exfiltrate information.

The Microarchitecture Side Channel (uSC) SIG will analyse -- on an ongoing basis -- the literature, propose and develop the RISC-V strategy to prevent microarchitectural information leakage, with an initial focus on timing side channels. Solutions of interest include microarchitectural purges, microstructure tagging, leakage resilient functionalities, preventing read-only architectural leakage (e.g., performance counters).

The SIG will discuss and propose recommendations on how to evolve the compliance model to include extensions with no functional side effects.

The SIG will develop one or more TG Charters that define one or more of the following items: written documentation, threat models, prototype implementations, toolchain support, and compliance suite for a RISC-V side channel leakage extension(s) or specification(s).

Summary

- SIG Charter: https://github.com/riscv-admin/uarch-side-channels/blob/main/CHARTER.md
- Gap Analysis: https://docs.google.com/presentation/d/1xbsY1zZelNQaqmMNVvfwYm8Q_RAB1TnEgd1QlFYkRj4/edit?usp=sharing
- Strategy to address Gaps:

What is the current status?

- Initial focus of SIG => First TG on fence.t with first draft proposal.
- Currently exploring next steps for the SIG

What assumptions have you made?

- Cf next slides: « Context and vocabulary » (summarized/simplified to fit the slot)

On whom are you dependent or who is dependent on you?

- No interdependencies with other SIG/TG
- Focusing on security-minded application processors for now

The outlook for completing this SIG is: No end in sight! It remains a rapidly evolving topic.

Context and vocabulary - 1

- **Communication across security boundaries inside microarchitecture**
  - Covert channels / side channels (they differ in threat models).
  - => Microarchitectural Data Sampling (MDS): exfiltrate a secret with a channel.
  - We restrict the scope of the discussions to timing channels.

- **ANY microarchitectural state can support a covert / side channel**
  - Classics: caches, branch prediction, performance counters, ...
  - But also : cache controllers, port contention, really any FSM...
  - Ease of exploitation varies, is not well known.

Context and vocabulary - 2

- **Spectre attacks: because of speculation, architectural and microarchitectural control flow may differ.**
  - The control flow inside the microarchitecture may be arbitrary => akin to a fault injection.
  - This control flow may be controlled by an attacker to trigger a load reading a secret.
- Speculation sources => Spectre variants *(some examples below)*

Branch direction prediction	Branch destination prediction	STL aliasing prediction	Value prediction
Spectre-PHT	Spectre-BTB	Spectre-STL	String Comparison Overrun
	Spectre-RSB		Zero Dividend Injection

Context and vocabulary - 3

- Speculation sources => Spectre variants
- Covert channels => Spectre subvariants

Spectre exploit = speculation + covert/side channel

Common misconceptions:

- “We just need a secure cache” => This is one covert channel among many.
- “Spectre is an implementation issue” => the purpose of speculation is to allow deviation from architectural control flow.

Aside:

- Meltdown does not use speculation => easier fix: check permissions first !
- There are other speculation-based vulnerabilities (random e.g.: GhostKnight = speculative RowHammer).
- Still lots of SIG-related vulnerabilities out there. This month: GhostRace, ZenHammer, GoFetch…

Gap Analysis - Covert/side channels

- **Gap:** Restricting microarchitectural sharing: only possible across architectural security boundaries (privileges, ASID, VMID).
  - E.g. branch predictors state must be flushed or statically partitioned by privilege level.
  - E.g. branch predictors state must be flushed (or partitioned) when switching to a new ASID.
  - Complex and not flexible. Performance is an issue.

- A way to prevent **covert channels**: fence.t (purpose of currently launching uSCRI-TG)

Gap Analysis

- **Gap:** Selective speculation

Software solutions

❑ Retpoline, Speculative Load Hardening (SLH), ...

❑ Are they implemented in GCC, LLVM ?

❑ Holistic security of these schemes is doubtful.

❑ <u>**HUUUUUGE**</u> performance penalty.

Hardware solutions

Stop speculation before 3. Detect sequences :

1. *Speculation:* A speculation triggering instruction.
2. *Acquisition:* A secret value is loaded (= can be accessed).
3. *Disclosure:* The secret value is source to a leaking instruction.

- Self-reported performance measurements are not reliable.
- Actual security put into questions when verified with formal methods (e.g. what if you switch steps 1 and 2).

Explore solutions => speculation barriers, serializing instructions, compiler patches, ... (ongoing academic work, not enough feedback yet on security/performance trade-off). Imho, we can do better than our competitors.

Gap Analysis

- **Gap:** What techniques, when do you use these techniques, what security policies ?
  - Performance costs seem to be prohibitive (need investigation).
  - E.g. Does your last build use SLH ? Yes/no, is it adapted to the situation ? Who decides, on what criteria ?
  - The topic is a lot more complex than the simplified picture in these slides. (See writeup.)

- **Microarchitectural security guide** to promote best practices.

Prioritization list of Actions

1. Fence.t => finalize TG, present first proposal to tech chairs
2. Microarchitecture security guide => non normative
3. Fence.spec => future TG ?, research paper planned for end of 2024

Open Issues

- Lots of unknowns: performance costs, implementation complexity, security adequacy, ...
- We are seeking feedback from industrial partners.

We are meeting April 9th, Security HC open slot (odd week)