The entire rebuttal form cannot exceed 2 pages, regardless of the number of words.

1. Application overview

Applicant name: Mathijs De Kremer
Title of the research proposal: Spectrebuster: proofing leakage-free speculative execution

Overall assessment

The reviewers provide valid concerns and interesting insights. Fundamental issues that are raised concern scalability and question the composability of leakage contracts. Practical issues that the reviewers have concern the feasibility of the planning and predict hardware vendors not willing to adopt the ideal of a hardware-software contract based ecosystem.

Detailed rebuttal (max. 1,000 words)

Issue-1: Combining leakage-free parts does not necessarily give a leakage free system.
Answer-1:
This will be one of the challenges of defining a composable notion of leakage-freedom. It is important to realize that we will not attempt to make all hardware components leakage-free. Instead, we will capture the type and amount of leakage per component, combine this for all components and use this to define a contract to which software must adhere to be secure.

Issue-2: Doubts about scalability to large scale systems.
Answer-2:
Both hardware and software will be susceptible to state explosion. On the hardware side we solve this by introducing a composable notion of leakage freedom. On the software side we aim to keep the problem tractable by observing that the window for a speculative vulnerability is small and always starts with a branching instruction. Therefore, we only have to verify the instruction sequences that fulfill these requirements.

Issue-3: Not all stakeholders are identified.
Answer-3:
The reviewers indicate that end users, software researchers, and Linux kernel maintainers are not named as stakeholders. However, end users and the scientific community are explicitly named in the knowledge utilization section. Linux kernel maintainers are implicitly named in the research plan section.

Issue-4: Hardware vendors may sacrifice security for performance.
Answer-4:
End users will not accept hardware vendors blindly letting go of mitigations for speculative vulnerabilities. Mitigating severe problems is most efficient in hardware. With Spectrebuster we aim to equip the community to better reason about the tradeoff between security and performance when it comes to speculative execution.

Issue-5: Missing risk mitigation.
Answer-5:
RA remarks that we should mitigate the risk of not finding Spectre gadgets in the Linux kernel. I propose that we try the tool on existing, known Spectre gadgets if we can not find Spectre gadgets in the kernel. Additionally RC likes to see risk mitigation for RT-1. If we fail to define a usable notion of composable leakage freedom, we will instead only reason about leakage freedom of an entire system and if necessary limit ourselves to small, theoretical examples.

Issue-6: Time underestimated.
Answer-6:
RA remarks that one year each is not enough for RT-2 and RT-3. In response, I propose to extend RT-2 with one year if necessary and work on it in parallel with RT-3. RT-3 is a dependency for RT-4 which will have to be postponed if one year is not enough for RT-3. I expect this to not be a problem since RT-4 and RT-5 can be completed together in one year, as also mentioned by RC.

Issue-7: Long term vision relies on hardware vendors publishing leakage contracts, with possibly sensitive information.
Answer-7:
RC remarks that hardware vendors might be reluctant to cooperate with software developers and publish their leakage contracts. This will ultimately depend on how much information about the hardware can be deduced from a leakage contract and how much performance hardware vendors have to sacrifice by not publishing leakage contracts and therefore having to better secure their hardware.

Issue-8: RT-4 and RT-5 are not different enough.
Answer-8:
The difference between RT-4 and RT-5 lies in developing a gadget scanner and being able to apply the gadget scanner in scanning for Spectre gadget in a real-world codebase. This step is not trivial, since real-world codebases are big and we will have to employ optimizations and heuristics to being able to scan the Linux kernel.

Issue-9: How will you capture unknown leakage channels?
Answer-9:
The model will only incorporate known sources of leakage because we can not know what we do not know. It is therefore indeed important to realize that we can only prove speculation safety for the model that we define. However, the sources of leakage will be fundamental. Examples of fundamental leakage sources are time and micro-architectural state. We do not expect new fundamental leakage sources to appear often.

Issue-10: It is unclear how data artifacts will be released.
Answer-10:
Data artifacts will have the form of leakage contracts. The exact format of a leakage contract will be defined in RT-3. We would like to publish these data as open-source artifacts but this depends on whether these data contain protected intellectual property.

Statement by the applicant

I have completed this form truthfully, on my own, meeting all page- and word-count requirements.

Name: Mathijs De Kremer
Place: Limmen
Date: December, 12, 2023