AFRL-AFOSR-JP-TR-2022-0033

Provable Time Protection for Eliminating Timing Channels

Heiser, Gernot
COMMONWEALTH SCIENTIFIC AND INUSTRIAL RESEARCH ORGANISATION
LIMEEONE AVE
CABERRA, , 2612
AU

05/25/2022
Final Technical Report

Air Force Research Laboratory
Air Force Office of Scientific Research
Asian Office of Aerospace Research and Development
Unit 45002, APO AP 96338-5002

REPORT DOCUMENTATION PAGE

PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ORGANIZATION.

1. REPORT DATE	2. REPORT TYPE	3. DATES COVERED	
20220525	Final	START DATE 20190619	END DATE 20210121

4. TITLE AND SUBTITLE
Provable Time Protection for Eliminating Timing Channels

5a. CONTRACT NUMBER	5b. GRANT NUMBER	5c. PROGRAM ELEMENT NUMBER
	FA2386-19-1-4021	
5d. PROJECT NUMBER	**5e. TASK NUMBER**	**5f. WORK UNIT NUMBER**

6. AUTHOR(S)
Gernot Heiser

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)	8. PERFORMING ORGANIZATION REPORT NUMBER
COMMONWEALTH SCIENTIFIC AND INUSTRIAL RESEARCH ORGANISATION LIMEEONE AVE CABERRA 2612 AU	

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)	11. SPONSOR/MONITOR'S REPORT NUMBER(S)
AOARD UNIT 45002 APO AP 96338-5002	AFRL/AFOSR IOA	AFRL-AFOSR-JP-TR-2022-0033

12. DISTRIBUTION/AVAILABILITY STATEMENT
A Distribution Unlimited: PB Public Release

13. SUPPLEMENTARY NOTES

14. ABSTRACT
The original proposal listed the following four objectives for the project, to be done for the seL4 microkernel: 1. Develop previously propotyped time protection mechanisms into a security model in the seL4 microkernel that integrates with existing information-flow reasoning about seL4; 2. formalise a high-level abstraction of micro-architectural state, i.e. the hardware resources that make execution speed dependent on execution history, and then formalise the operations that affect this state; 3. formalise absence of temporal interference (and thus timing channels) as spatial or temporal partitioning of such state; 4. building on the above, formally prove that time protection mechanisms recently prototyped in seL4 eliminate timing channels by correctly partitioning microarchitectural state. Over the course of the project we made initial progress on the first three objectives. We note that the project is co-funded by the Australian Research Council (ARC) for the years 2019–21.

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES
a. REPORT U	b. ABSTRACT U	c. THIS PAGE U	SAR	6

19a. NAME OF RESPONSIBLE PERSON	19b. PHONE NUMBER *(Include area code)*
ALAN LIN	227-7009

Provable Time Protection for Eliminating Timing Channels

Final Report for AOARD Grant FA2386-19-1-4021

March 23, 2021

1 Background

The original proposal was for a three-year project. AOARD granted one year of funding with options for two further years. A no-cost extension to the end of 2020 was granted in June 2020 to the first funding year due to Covid-related delays in recruiting staff for the project. In September 2020 AOARD exercised the first funding option. In November 2020, before the funding for the first year was fully expended, CSIRO decided to terminate the project and return the Option 1 as well as the remainder of the first year's funding. Effectively, the team performed 90% of one year of the project.

We note that the project is co-funded by the Australian Research Council (ARC) for the years 2019–21.

2 Objectives (from Proposal)

The original proposal listed the following four objectives for the project, to be done for the seL4 microkernel:

1. Develop previously propotyped time protection mechanisms into a security model in the seL4 microkernel that integrates with existing information-flow reasoning about seL4;

2. formalise a high-level abstraction of micro-architectural state, i.e. the hardware resources that make execution speed dependent on execution history, and then formalise the operations that affect this state;

3. formalise absence of temporal interference (and thus timing channels) as spatial or temporal partitioning of such state;

4. building on the above, formally prove that *time protection* mechanisms recently prototyped in seL4 eliminate timing channels by correctly partitioning microarchitectural state.

Over the course of the project we made initial progress on the first three objectives.

3 Progress

We decided to adopt a "breadth-first" approach to achieving our objectives, by first proving the absence of timing channels for a highly abstract (and grossly over-simplified and thus unrealistic) formal model of the hardware and the operating systems's (OS's) mechanisms, and then refine this model in several steps to a realistic model of the hardware. Over the course of the project, we completed formalisation of two of the three OS mechanisms involved (see below for detail) and proved non-interference for the first with some progress on the second.

Our experience so far makes us very confident that it will be possible to achieve the desired proofs at least for the three mechanisms modelled so far. Completing the "shallow" pass will give us confidence that the whole project will succeed.

3.1 Technical Findings and Contributions

3.1.1 Objective 1: TP security model in seL4

The time protection (TP) mechanisms, as previously prototyped, have four components:

1. a kernel clone mechanism that enables partitioning a system down to the kernel level, with no code and only a small number of data structures shared between them;

2. partitioning off-core microarchitectural state (lower-level caches);

3. on a security-domain switch, flushing on-core state that is not partitionable;

4. ensuring deterministic domain-switch latency by prefetching kernel data structures after the flush and padding flush+pre-fetch to a worst-case latency.

We had known from our earlier work that present mainstream processors do not provide suitable mechanisms for achieving all of this, and had planned to perform the project under the assumption that such mechanisms will become available in time. We have since realised that the new, open RISC-V instruction-set architecture (ISA) allows us to accelerate this process, in two ways: (i) by influencing the specification of the RISC-V ISA to ensure future RISC-V processors will have to provide these mechanisms and (ii) prototyping such mechanisms in an existing open-source implementation of the ISA.

We actively pursued both parts. The PI achieved buy-in from the RISC-V Security Standing Committee for the enhancements to the ISA and is actively working with members of the respective technical committees on the specification these mechanisms. While progress here is somewhat slower than hoped, we expect a suitable draft ISA specification sometime this year.

At the same time we partnered with the group of Prof Luca Benini at ETH Zurich, who had developed Ariane, an open-source, 64-bit implementation of RSIC-V. We collaborated on the implementation of the TP support mechanisms in Ariane, and could demonstrate that these are (i) simple and easy to implement, (ii) highly effective in preventing timing channels when used correctly by the operating system and (iii) impose negligible overhead. This work resulted in a Masters thesis at ETH and a workshop publication [Wistoff et al., 2020], a workshop publication and then a conference publication [Wistoff et al., 2021] that won a **Best-Paper Award**.

Making the best of the opportunities provided by RISC-V required porting our mechanisms to the RISC-V version of seL4, which in turn required merging them from the 2-year-old seL4 version they were developed for to the latest kernel. In order to enable later verification, this required changes in the design of the prototype and re-implementation of large parts.

This work has mostly been completed, with the support for kernel cloning operational in the latest version of seL4, without breaking any of the existing verification. This provides the base for achieving Objective 1.

3.1.2 Objective 2: Formalising microarchitectural state

We started work on formalising partitionable state (physically-addressed caches below the on-core L1 caches). Specifically we developed a cache model in Isabelle that captures a notion of cache sets and mapping of addresses to such sets, and from that derives a notion of memory colour.

The cache model tracks individual cache sets, each of which defines a memory colour. It also formally defines a memory separation property that

states what it means for separate security partitions to have access only to disjoint colours. In addition the model also records elapsed execution time for memory operations, as influenced by the cache state (formally, as a function of whether memory accesses hit or miss the cache).

We then formalised non-partitionable hardware state (on-core caches, branch predictors and pre-fetchers etc) as history-dependent state, as well as its resetting to a history-independent state by suitable hardware mechanisms. We model the reset as an operation with a latency that is also history-dependent but with a known upper bound. This then allowed us to reason about the OS padding the latency of the reset to this upper-bound.

We started work on formalising the third component, prefetching of kernel data required for kernel exit. This work is still in progress.

3.1.3 Objective 3: Proving non-interference

We have demonstrated the utility of this model by proving that read accesses of addresses to different colours cannot interfere in the cache. This is the starting point for an eventual proof that memory colouring will partition the cache such that different partitions cannot interfere in the cache.

The formal noninterference property that we prove guarantees that the time taken by memory operations in one partition cannot be affected by another partition, when memory separation holds. Specifically, the current model considers memory read operations and the theorem states that the elapsed time of a read operation depends only on the cache state for the cache sets to which the partition has access, plus that such operations do not affect the state of any cache sets to which the partition does not have access. Absence of execution time interference then follows from memory separation, i.e. when each partition has access to disjoint colours and so to disjoint cache sets.

For non-partitionable state, the model mentioned in Objective 2 is detailed enough to observe that executing instructions changes the state of this cache and thereby can influence the timing of instructions in another protection domain. That is, the model demonstrates the presence of unwanted information flows if no suitable protection mechanism is used. The noninterference property described above is agnostic of the mitigation mechanism or the cause of time differences — it will be violated for any timing difference visible in the timing state of model. This means, we establish the same property as for partionable state, but the proof is different: we show that the mechanism of flushing the L1 cache and padding the flush time to its worst case closes this timing channel.

Like the model, the proof for the pre-fetching mechanism for shared kernel data is progress. The aim is again to re-use the same top-level property, and to establish a reasoning framework that allows us to show that this property holds for the new mechanism.

3.1.4 Objective 4: Proving that time protection prevents timing channels

No progress on this objective, as it is dependent on fully achieving Objectives 2 and 3.

3.1.5 Opportunities arising

The developments around RISC-V, described in Section 3.1.1, open up a new and exciting opportunity: adding a further component to this project, namely investigating how the access-control mechanisms in the RISC-V version of seL4 can be formally verified. This will form the basis for later connecting our time-protection proofs to the verified kernel all the way to the binary. This raises the exciting prospect of an end-to-end proof of freedom from microarchitectural timing channels.

At the same time, there are companies already designing RISC-V processors with our proposed ISA extensions built in, so by the time these proofs are completed, we expect to have compliant hardware for which they are valid.

We were planning to use most of the second-year AORD funding for these access-control proofs, sadly the early termination of the project by CSIRO makes this impossible for now.

3.2 Academic papers or presentations

Gernot Heiser. Time protection: Preventing timing channels. Presentation to the RISC-V Security Standing Committee and Privileged ISA Specification Standing Committee, June 2020.

Gernot Heiser, Toby Murray, and Gerwin Klein. Towards provable timing-channel prevention. *ACM Operating Systems Review*, 54:1–7, August 2020. ISSN 0163-5980. doi: https://doi.org/10.1145/3421473.3421475.

Nils Wistoff, Moritz Schneider, Frank Gürkaynak, Luca Benini, and Gernot Heiser. Prevention of microarchitectural covert channels on an open-source 64-bit RISC-V core. In *Workshop on Computer Architecture Research with RISC-V (CARRV)*, Valencia, Spain, May 2020. ACM.

Nils Wistoff, Moritz Schneider, Frank Gürkaynak, Luca Benini, and Gernot Heiser. Microarchitectural timing channels and their prevention on an open-source 64-bit RISC-V core. In *Design, Automation and Test in Europe (DATE)*, virtual, February 2021. IEEE.

3.3 Patents filed or underway

N/A

3.4 Links to any published code bases or datasets

N/A (yet)

3.5 Meetings with any US universities or labs

N/A

3.6 External funding enabled as a result of this grant

No new external funding, but the project is already co-funded by the ARC.

3.7 Students graduated

Nils Wistoff, Masters student at RWTH Aachen University (Germany) completed his thesis on the project, co-supervised by Prof Luca Benini (ETH Zurich) and Prof Gernot Heiser (UNSW Sydney).