TODO

TODO

Bachelor Thesis

by

TODO

in fulfillment of requirements for degree

Bachelor of Science (B.Sc.)

submitted to

Rheinische Friedrich-Wilhelms-Universität Bonn

Institut für Informatik IV

Arbeitsgruppe für IT-Sicherheit

in degree course

Computer Science (M.Sc.)

First Supervisor: TODO
University of Bonn

Second Supervisor: TODO
TODO

Sponsor: TODO
University of Bonn

TODO

Abstract

TODO

Contents

1 Introduction

Contrary to older processors, modern Intel CPUs implement a number of optimization techniques that increase their efficiency. One of which is the concept of out-of-order execution, which takes advantage of the mutual independence of instructions that would normally be executed sequentially. This allows a CPU to continue working on other instructions while one is waiting for data from memory, for example. A second optimization technique is called speculative execution and involves the prediction of branches. Rather than waiting for an instruction that determines which branch is taken, the outcome is predicted. With either technique, the CPU might encounter cases where the current CPU state must be rolled back to a previous one. For out-of-order execution, this happens if an instruction raises an exception (illegal memory access, for example). For speculative execution, this happens if a branch is mispredicted. A rollback causes some instructions that are currently being executed (in-flight) to finish, but all instructions that come after the faulting instruction to be executed again.

The disclosure of both Spectre and Meltdown in early 2018 introduced a whole family of vulnerabilities that take advantage of both out-of-order and speculative execution to leak secrets over the processor's caches. And while the performance losses introduced by software and hardware mitigations are measurable, neither vulnerability can be exploited freely on a fully patched system.

As a result, however, the process of trying to exploit one of the vulnerabilities for the sake of learning how they work in detail can be challenging. Apart from the software, which may be obtained by installing an older version of the Linux Kernel that does not implement any mitigations, one must also make sure their CPU is effected by the vulnerabilities and has not yet received any relevant microcode updates from Intel. Often times, this means that a user's personal computer does not meet these requirements.

We design and implement a graphical CPU emulator that supports single-step out-of-order and speculative execution and is vulnerable to select variations of Spectre and Meltdown. While simplified in comparison to real hardware, the emulator allows its users to gain a better understanding of how exactly the two vulnerabilities can be exploited. Furthermore, the user may experiment with (ineffective) mitigations or implement their own microprograms that are executed once rollbacks are completed. We supply example programs that can be run by the emulator and serve both as an entry point for the user as well as the basis of our evaluation.

Firstly, chapter bla bla bla discusses bla bla bla.

Statement of Authorship

I hereby confirm that the work presented in this bachelor thesis has been performed and interpreted solely by myself except where explicitly identified to the contrary. I declare that I have used no other sources and aids other than those indicated. This work has not been submitted elsewhere in any other form for the fulfilment of any other degree or qualification.

TODO

TODO