Compiler Engineering Project Report - Mitigating Spectre and Meltdown

Devansh Agarwal: ES16BTECH11009
Sidharth Mohla: ES16BTECH11020

1 Problem Statement

In this project a study on Spectre and Meltdown vulnerabilities was to be made, and an LLVM based solution that detects and mitigates Spectre and Meltdown vulnerabilities was to be developed. Since for Meltdown many OS patches have already been released, major focus was on Spectre.

1.1 Spectre and meltdown

1.1.1 Spectre

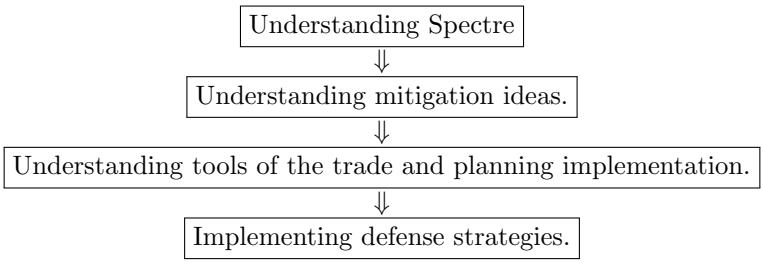
Spectre is a vulnerability that allows attackers to access arbitrary locations in memory by tricking the processor to *speculatively* load the data into the cache and then performing a timing attack on the cache to get the information piece by piece .
As of Nov. 2018, more than 9 variants of this class of bugs have been reported, although no known attacks have been discovered exploiting these bugs.

1.1.2 Meltdown

Meltdown also works in a similar fashion. Here, the attacker tries to read privileged data, which obviously fails but results in data being cached, which can thus be extracted using the same similar timing attack. It has been patched in most modern OS.

2 Approach

Since the inception, we have come afar. The following time line gives a general idea of what went in the project.



- Understanding Spectre:
  In depth understanding of spectre and meltdown, side channel attacks and running POV on different processor platforms (AMD x86, Intel x64 and ARM).Reading Papers/Reports:
  - oo7: Low-overhead Defense against Spectre Attacks via Binary Analysis
  - Spectre attacks: Exploiting speculative execution.
  - Paul Kocher. Spectre mitigations in Microsoft's C/C++ compiler
  - Microsoft community. C++ developer guidance for speculative execution side channels.
  - Interprocedural Taint Analysis for LLVM bitcode.
- Understanding mitigation strategies:
  Understanding flaws of oo7, understanding test cases in depth, Mitigation strategies like : Insert lfence, nop, garbage instructions, pointer hardening.
- Understanding tools of the trade and planning implementation:
  Learning about LLVM MIR and CodeGen infrastructure, generating and understanding MIR code for spectre, using clang for Static Analysis (exploring the inbuilt taint analysis checker), running that on spectre test cases, tried to use tools like Phaser for spectre detection but wasn't able to make it work. Thinking about advantages and disadvantages of writing pass in MIR or normal IR.
- Implementing defense strategies:
  Writing a modular LLVM pass for IR that detects and fixes the vulnerabilities, making reports on how LLVM has inbuilt checks for spectre and associated problems, project closure.

3 Experimental Setup

Primary Setup

- An LLVM pass written from scratch that detects and mitigates Spectre.
- The 15 test cases by Paul Kocher made into compilable code.
- Extra test cases to display branch fencing in non-Spectre branches.
- LLVM 8 or 7 build with clang and llvm-dis also built.
- An Intel/AMD processor powered machine.
- Location of pass and commands to run are in the private github repository.

Secondary Setup

- Running Spectre POV.
- Running Meltdown POV.
- These POV can be run by following the instructions on Github.

4 Conclusion

- A Modular pass was created which works for all the 15 test cases by Paul Kocher. Additionally, apart from detection the transformation pass is able to add fences wherever needed in the IR and thus also in the binary generated (verified by generating assembly for the binary) for all the cases, which is equivalent to removing the vulnerability.All the test cases are made into compilable code and fenced files for all the test cases is generated and put on github. Additional test cases are also written. In the pass the initial taints are limited to the ones generated by scanf currently but other sources can be added without changing other parts of the code. Currently some non spectre branches are also being flagged in future work a tighter bound can be implemented.
- Various Google documents are also created which contain information(or where to find it) about spectre meltdown and other relevant stuff(MIR, taint analysis etc.). Also a doc which has all the 15 test cases by Paul Kocher is also made.

.

5 Links to references

5.1 Spectre and meltdown videos

Spectre explanation video by Scott Manley
Spectre demo video
Video: Computerphile Spectre explanation
Stanford Seminar - Exploiting modern microarchitectures: Meltdown, Spectre and other hardware attacks video
Meltdown and Spectre with Matt Godbolt video
Matt Godbolt explaining his presentation on Spectre and Meltdown
Paul Kocher talk on exploiting Speculative execution
Video explaining cache side channel attack

5.2 Papers, blog entries and code samples

Meltdown official site
Meltdown paper
Spectre paper
oo7 low overhead defense against Spectre attacks via binary analysis
Interprocedural taint analysis for LLVM-bitcode
Paul Kocher blog on Spectre Mitigations in Microsoft's C/C++ Compiler
Google Project Zero
Meltdown PoC
Spectre PoC
x86 Taint analysis

5.3 MIR basic tutorials

MIR language reference
YAML documentation

The LLVM Target-Independent Code Generator Infrastructure
YAML in one video
MIR canon Improving code diff. using canonical transformation Video

5.4 Basic Static analysis with clang

Clang checker developer manual
Clang experimental checkers
P. Goldsborough Build useful tools with clang video The Clang AST tutorial video
Clang Static Analysis
Forum resource for taint analysis inbuilt in clang

5.5 Materials generated during project

Github page
Google drive folder