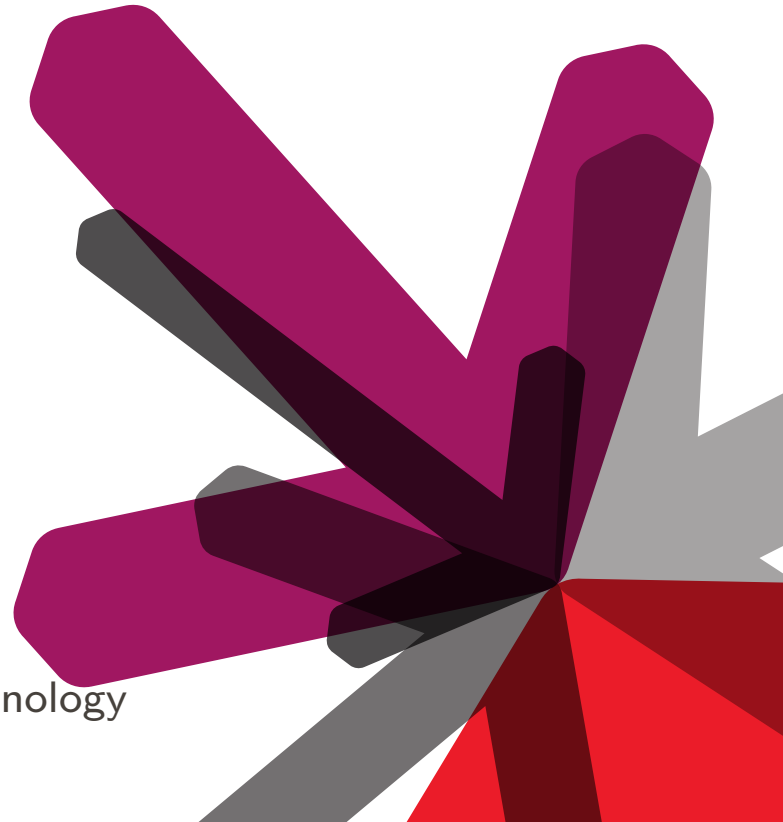
Hochschule **RheinMain**
University of Applied Sciences
Wiesbaden Rüsselsheim

SOFTWARE BASED SIDE - CHANNEL ATTACKS ON CPUS

Their history and how we behaved

Last update: August 8, 2018



Harald Heckmann

Computer Science – Smart Systems for Man and Technology
RheinMain University

CONTENTS

INTRODUCTION

OVERVIEW OF ATTACKS

OVERVIEW

Table: Overview of attacks

Attack	**Date**	**Aim**
Remote attacks [...]	2003	side-channel (sc) on network
Prime+Probe	2005	analyse cache/memory access
Pred. secr. keys [...]	2006	sc attack on BPU (passive/active)
Cache-Games [...]	2011	sc attack on cache (single core)
Flush+Reload	2013	low-noise cross vm L3 cache sc
Evict+Reload	2015	architecture independent cache sc
Meltdown	2018	exploit out-of-order exec. (x86)
Spectre (v1 and v2)	2018	exploit speculative exec. (branch)
SgxSpectre	2018	break intels "secure" enclave
BranchScope	2018	exploit PHT instead of BTB
Spectre-NG	2018	new variants to exploit spec. exec.
TLBleed	2018	successful attack on TLB

ATTACKS IN DETAIL

ATTACKS IN DETAIL

Contents:

- Flush+Reload (2013)
- Evict+Reload (2015)
- Meltdown (2018)
- Spectre (2018)

FLUSH+RELOAD

Crucial knowledge:

- Shared pages
- Cache hierarchy
- Address translation
- Timing side-channel attacks
- X86 instructions

FLUSH+RELOAD

Given: We can deduce secrets from execution paths
Setting: one victim process, one attacker process, executed on x86 architecture, page sharing enabled

FLUSH+RELOAD

1. map victim code into own memory (abuse shared pages)
2. determine relevant text segment addresses
3. let victim execute it's code until it reaches the code of interest
4. flush the relevant cachelines using the instruction "clflush" (address known through page sharing)
5. let the victim execute some lines of code
6. access the relevant addresses and measure the timing
7. reconstruct execution path, derive secrets

FLUSH+RELOAD

Weaknesses:

- Uses x86 instructions
- Oblivious to address diversification (e.g. ASLR)

EVICT+RELOAD

Crucial knowledge:

- Pages in detail
- Cache in detail
- Timing side-channel attacks

EVICT+RELOAD

Given: We know the cache layout.
Problems: LLC cache is physically addressed, modern LLC caches is sliced with the help of a secret hash function.
Setting: Cache inclusiveness is given and large pages are enabled

EVICT+RELOAD

1. create buffer B containing memory at least twice the size of the LLC cache.
2. create two sets:
   - Conflict set - contains enough addresses to evict the complete LLC
   - Eviction sets - contains enough addresses to clear one specific set in the LLC
3. access memory location A, you know in which set it is stored, but not in which slice
4. load every address in the conflict set. Access A. If A was not evicted, add it to the conflict set
5. Test for every remaining address B in the buffer: access B, access every address in the conflict set. Remove addresses one by one from the conflict set and retry the test. If the test fails, the removed address is required to evict that specific set in each slice.

EVICT+RELOAD

Weaknesses:

- Large page size required
- Cache inclusiveness property required

MELTDOWN

Crucial knowledge:

- Out-of-order execution
  - pages
- cache hierarchy
- flush+reload

MELTDOWN

Given: We execute this attack on x86 architecture
Setting: two processes in own control: One attacker and one spy

exeption()
array[x * pagesize]

MELTDOWN

Weaknesses:

- Relies on a permission check bug in Intel CPUs
  - Relies on flush+reload
- Relies on the fact that the whole physical memory is mapped into kernel address space

SPECTRE

Crucial knowledge:

- BPU (BTB)
- Speculative execution
  - Evict+Reload
  - Prime+Probe
- optionally Flush+Reload

SPECTRE

Given: The CPU supports speculative execution
Setting: A victim process which contains code following a specific pattern, an attacker process

SPECTRE V1

$$\text{if (x} > \text{bounds\_lower \&\& x} < \text{bounds\_upper)}$$
$$\text{array\_to\_probe[array[x} * 256]]$$

SPECTRE

Weaknesses:

- Memory loads during speculative execution
- Those inherited from Flush+Reload or Evict+Reload and Prime+Probe

MITIGATIONS

FLUSH+RELOAD

Weaknesses:

- Uses x86 instructions
- Oblivious to address diversification (e.g. ASLR)
  - (Shared pages)

EVICT+RELOAD

Weaknesses:

- Large page size required
- Cache inclusiveness property required

MELTDOWN

Weaknesses:

- Relies on a permission check bug in Intel CPUs
- Relies on flush+reload
- Relies on the fact that the whole physical memory is mapped into kernel address space

SPECTRE

Weaknesses:

- Memory loads during speculative execution
- Those inherited from Flush+Reload or Evict+Reload and Prime+Probe

SPECTRE V1

Kernel function: array_index_mask_nospec

SPECTRE V2

Retpoline

REACTION AND AFTERMATH

ORGANIZATION

1. What were the warning signals given by the researchers?
2. How did manufacturer of CPUs react to the publication of such attacks?
3. Have special communication channels next to papers been used to sensitize journalists or computer users to the problem?

WHAT WERE THE WARNING SIGNALS GIVEN BY THE RESEARCHERS?

From "Remote Timing Attacks are Practical" (2003):
"Our experiments show that, counter to current belief, the timing attack is effective when carried out between machines separated by multiple routers. Similarly, the timing attack is effective between two processes on the same machine and two Virtual Machines on the same computer.

WHAT WERE THE WARNING SIGNALS GIVEN BY THE RESEARCHERS?

From the paper containing the "Prime+Probe" attack (2005):
"At the system level, cache state analysis is of concern in essentially any case where process separation is employed in the presence of malicious code. Beyond the demonstrated case of encrypted filesystems, this includes many multi-user systems, as well as web browsing and DRM applications. [...] the leakage also occurs in no cryptographic systems and may thus leak sensitive information directly."

WHAT WERE THE WARNING SIGNALS GIVEN BY THE RESEARCHERS?

From the paper "Predicting secret keys via Branch Prediction" (2006):

"this paper has identified the branch prediction capability of modern microprocessors as a new security risk [...] The practical results from our experiments should be encouraging to think about efficient and secure software mitigations for this kind of new side-channel attacks."

WHAT WERE THE WARNING SIGNALS GIVEN BY THE RESEARCHERS?

From the paper containing the "Flush+Reload" attack (2013):
"The technique is generic and can be used to monitor other software. It can be used to devise other types of attacks on cryptographic software."

WHAT WERE THE WARNING SIGNALS GIVEN BY THE RESEARCHERS?

From the paper containing the "Evict+Reload" attack (2015):

"[...] we believe that our attack is eminently practical, and as such presents a real threat against keys used by cloud-based services"

WHAT WERE THE WARNING SIGNALS GIVEN BY THE RESEARCHERS?

From the paper "Meltdown" (2018):
"The fact that hardware optimizations can change the state of microarchitectural elements, and thereby imperil secure software implementations, is known since more than 20 years [20]. Both industry and the scientific community so far accepted this as a necessary evil for efficient computing. Today it is considered a bug when a cryptographic algorithm is not protected against the microarchitectural leakage introduced by the hardware optimizations."

WHAT WERE THE WARNING SIGNALS GIVEN BY THE RESEARCHERS?

From the paper containing the "Spectre" attack (2018):
"As the attack involves currently undocumented hardware effects [...] there is currently no way to know whether a particular code construction is, or is not, safe across today's processors – much less future designs. A great deal of work lies ahead."

HOW DID MANUFACTURER OF CPUS REACT TO THE PUBLICATION OF SUCH ATTACKS?

- they took this problem serious
  - develop microcode
  - work close with OS developers
  - showed clear interest to close those leaks

HOW DID MANUFACTURER OF CPUS REACT TO THE PUBLICATION OF SUCH ATTACKS?

- The users are at the second position - after money
  - no recall or compensation for the damages
  - they have continued to develop features after warnings (e.g. speculative execution)

HAVE SPECIAL COMMUNICATION CHANNELS NEXT TO PAPERS BEEN USED TO SENSITIZE [...]

1. conferences
2. journals
3. forum discussions
4. news services (like heise.de in germany)

$\rightarrow$ most computer users are not reached

CONCLUSION

CONCLUSION

$\rightarrow$ We need open-source hardware (long term security)

CONCLUSION

$\rightarrow$ We need "fence" kernel functions (short term security)

QUESTIONS

- Send questions to: Harald.b.Heckmann@student.hs-rm.de
- This latex template is based on the HSRM theme from Benjamin Weiss
- The HSRM theme is licensed with GNU Public License