OUTLINE

1. Introduction
2. Background
3. KASLR
4. KAISER
5. KPTI
6. Conclusion
INTRODUCTION
INTRODUCTION

Meltdown: new attack, January 2018
INTRODUCTION

Meltdown: new attack, January 2018

KPTI: Kernel Page Table Isolation
→ patch against Meltdown
INTRODUCTION

Meltdown: new attack, January 2018

KPTI: Kernel Page Table Isolation
→ patch against Meltdown

KAISER: Kernel Address Isolation to have Side channels Efficiently Removed
→ original concept
→ prevent side-channel attacks against KASLR
INTRODUCTION

Meltdown: new attack, January 2018

KPTI: Kernel Page Table Isolation
→ patch against Meltdown

KAISER: Kernel Address Isolation to have Side channels Efficiently Removed
→ original concept
→ prevent side-channel attacks against KASLR

KASLR: Kernel Address Space Layout Randomization
BACKGROUND
VIRTUAL ADDRESS SPACE

Context switch: switch between two processes

CR3 update

4k memory page

entry

PML4

PDP

PD

PT
Context switch: switch between two processes → CR3 update
## VIRTUAL ADDRESS SPACE

User - kernel space switch:
Kernel space is mapped into every user process
→ No CR3 update
User - kernel space switch:
Kernel space is mapped into every user process
→ No CR3 update

TLB: Translation Lookaside Buffer
→ Cache for Page Table Entries
→ Performance increase
MELTDOWN

Out-of-Order Execution:
→ Improves performance

Approach:

1. Inaccessible kernel memory is loaded → exception
2. Out-of-order execution of following code
3. Content of accessed kernel memory is leaked through cache side-channel

→ Entire physical memory can be read
KASLR
Kernel Address Space Layout Randomization

Randomize placement of kernel at boot time
→ To secure the kernel address information

Attacks:
Double Page Fault Attack,
Intel TSX-based Attack,
Prefetch Side-Channel Attack
DOUBLE PAGE FAULT ATTACK

Allocated: page belongs to the address space
Accessible: right access privilege

1. Access inaccessible kernel memory
2. First page fault
   2.1 Page allocated → cached
   2.2 Page not allocated → not cached
3. Second page fault
   3.1 Cached → less time
   3.2 Not cached → more time
4. Learning if kernel memory is allocated
DOUBLE PAGE FAULT ATTACK

Allocated: page belongs to the address space
Accessible: right access privilege

1. Access inaccessible kernel memory
2. First page fault
   2.1 Page allocated → cached
   2.2 Page not allocated → not cached
3. Second page fault
   3.1 Cached → less time
   3.2 Not cached → more time
4. Learning if kernel memory is allocated

→ KASLR is dead.
KAISER
### KAISER

**Kernel Address Isolation to have Side channels Efficiently Removed**

Published in July 2017
Prevent side-channel attacks against KASLR

→ Isolate user address and kernel address space
Kernel Address Isolation to have Side channels Efficiently Removed

Published in July 2017
Prevent side-channel attacks against KASLR

→ Isolate user address and kernel address space

Before Meltdown:

→ Kernel space mapped in user space
→ Protected through permission bits in translation tables
DIFFERENT MODELS

(a) Regular OS
DIFFERENT MODELS

(a) Regular OS

User memory  not mapped
context switch

switch address space

not mapped  Kernel memory

(b) Stronger Kernel Isolation

User memory  not mapped
context switch

switch address space

not mapped  Kernel memory
DIFFERENT MODELS

(a) Regular OS

(b) Stronger Kernel Isolation

(c) KAISER
PARTITIONING

CR3 + 0x1000
User

CR3
Kernel

PML4 User

PML4 Kernel

CR3[12]=1

CR3[12]=0
1. Minimizing the Kernel Address Space Mapping for context switch some locations need to be mapped
1. Minimizing the Kernel Address Space Mapping for context switch some locations need to be mapped → Interrupts, exceptions, system calls
1. Minimizing the Kernel Address Space Mapping for context switch some locations need to be mapped → Interrupts, exceptions, system calls

2. Efficient and Secure TLB Management more address space switches ⇒ more TLB flushes
1. Minimizing the Kernel Address Space Mapping for context switch some locations need to be mapped → Interrupts, exceptions, system calls

2. Efficient and Secure TLB Management more address space switches ⇒ more TLB flushes → PCIDs

Process context identifiers
→ Each TLB-entry marked with process id
Double Page Fault Attack

![Graph showing execution time in cycles for unmapped and mapped cases with and without KAISER.]

- **Unmapped**:
  - No KAISER: 14,621 cycles
  - KAISER: 12,307 cycles

- **Mapped**:
  - No KAISER: 14,621 cycles
  - KAISER: 12,282 cycles

The graph illustrates the execution time in cycles for different scenarios involving unmapped and mapped pages, with and without KAISER.
KPTI
<table>
<thead>
<tr>
<th>Outline</th>
<th>Introduction</th>
<th>Background</th>
<th>KASLR</th>
<th>KAISER</th>
<th>KPTI</th>
<th>Conclusion</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Kernel Page Table Isolation

Initial Linux patch by Dave Hansen
Kernel Page Table Isolation

Initial Linux patch by Dave Hansen

→ PCIDs
→ Trampoline functions
Kernel Page Table Isolation

Initial Linux patch by Dave Hansen

→ PCIDs
→ Trampoline functions

Current Status:
Linux: 4.15
Windows: 17035
MacOs: 10.13.2
EVALUATION

Syscalls, interrupts and exceptions
→ Performance loss can vary heavily: ~5% ≥30%
EVALUATION

Syscalls, interrupts and exceptions
→ Performance loss can vary heavily: ~5% ≥30%

lseek
no kaiser: 5.2 M/s
kaiser+ pcid: 3.0 M/s
kaiser+nopcid: 2.2 M/s
CONCLUSION
CONCLUSION

→ Best short-term solution

→ Performance loss varies heavily

→ New hardware or microcode update?
Gruss, Daniel and Lipp, Moritz and Schwarz, Michael and Fellner, Richard and Maurice, Clementine and Mangard, Stefan

»KASLR is Dead: Long Live KASLR«
Springer International Publishing, 2017

Lipp, Moritz and Schwarz, Michael and Gruss, Daniel and Prescher, Thomas and Haas, Werner and Mangard, Stefan and Kocher, Paul and Genkin, Daniel and Yarom, Yuval and Hamburg, Mike

»Meltdown«
ArXiv e-prints, 2018

R. Hund and C. Willems and T. Holz

»Practical Timing Side Channel Attacks against Kernel Space ASLR«
Dave Hansen
»[PATCH 00/30] [v3] KAISER: unmap most of the kernel from
userspace page tables«
https://lwn.net/Articles/738997/, 2017
Accessed: June 1, 2018

Jonathan Corbet
»KAISER: hiding the kernel from user space«
https://lwn.net/Articles/738975/, 2017
Accessed: June 30, 2018

Dave Hansen
»Use global pages with PTI«
https://lwn.net/Articles/750049/, 2018
Accessed: July 1, 2018"
Thank you for your attention.