Software-based Microarchitectural Attacks: What do we learn from Meltdown and Spectre?

Daniel Gruss
March 26, 2019
Graz University of Technology
You realize it is something big when...

- it is in the news, all over the world
- you get a Wikipedia article in multiple languages
- there are comics, including xkcd
- you get a lot of Twitter followers after Snowden mentioned you
You realize it is something big when...

- it is in the *news*, all over the world
You realize it is something big when...

- it is in the news, all over the world
- you get a Wikipedia article in multiple languages
- there are comics, including xkcd
- you get a lot of Twitter follower after Snowden mentioned you

Daniel Gruss — Graz University of Technology
You realize it is something big when...

• it is in the news, all over the world
• you get a Wikipedia article in multiple languages
• there are comics, including xkcd
• you get a lot of Twitter followers after Snowden mentioned you

Daniel Gruss — Graz University of Technology
The Fallout

You realize it is something big when...

- it is in the news, all over the world
- you get a Wikipedia article in multiple languages
- there are comics, including xkcd
- you get a lot of Twitter followers after Snowden mentioned you

Daniel Gruss — Graz University of Technology
You realize it is something big when...

- it is in the news, all over the world
- you get a Wikipedia article in multiple languages
- there are comics, including xkcd
- you get a lot of Twitter follower after Snowden mentioned you

SECURITY FLAW REVEALED

<table>
<thead>
<tr>
<th>Company</th>
<th>Price</th>
<th>Change</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Intel (Prev)</td>
<td>45.26</td>
<td>-1.59</td>
<td>-3.39%</td>
</tr>
<tr>
<td>Intel (After Hours)</td>
<td>44.85</td>
<td>-0.41</td>
<td>-0.91%</td>
</tr>
</tbody>
</table>

SHROUT: ISSUE NOT UNIQUE TO INTEL, BUT IT'S AFFECTED THE MOST
You realize it is something big when:

- it is in the news, all over the world
- you get a Wikipedia article in multiple languages
- there are comics, including xkcd
- you get a lot of Twitter followers after Snowden mentioned you

Daniel Gruss | Graz University of Technology
You realize it is something big when...

- it is in the news, all over the world
- you get a Wikipedia article in multiple languages
Kernel page-table isolation

From Wikipedia, the free encyclopedia
(Redirected from KAISER)

"KPTI" redirects here. For other uses, see KPTI (disambiguation).

Kernel page-table isolation (KPTI or PTI,[1] previously called KAISER)[2][3] is a Linux kernel feature that mitigates the Meltdown security vulnerability (affecting mainly Intel's x86 CPUs)[4] and improves kernel hardening against attempts to bypass kernel address space layout randomization (KASLR). It works by better isolating user space and kernel space memory.[5][6] KPTI was merged into Linux kernel version 4.15,[7] and backported to Linux kernels 4.14.11, 4.9.75, 4.4.110.[8][9][10] Windows[11] and macOS[12] released similar updates. KPTI does not address the related Spectre vulnerability.[13]
You realize it is something big when...

- it is in the news, all over the world
- you get a Wikipedia article in multiple languages
- there are comics, including xkcd
- you get a lot of Twitter followers after Snowden mentioned you

Meltdown is a hardware vulnerability affecting Intel x86 microprocessors and some ARM-based microprocessors.\[1\][2][3] It allows a rogue process to read all memory, even when it is not authorized to do so.

Meltdown affects a wide range of systems. At the time of disclosure, this included all devices running any but the most recent and patched versions of iOS,\[4\] Linux,\[5][6\], macOS,\[4\] or Windows. Accordingly, many servers and cloud services were impacted,\[7\] as well as a potential majority of smart devices and embedded devices using ARM-based processors (mobile devices, smart TVs and others), including a wide range of networking equipment. A purely software workaround to Meltdown has been assessed as slowing computers between 5 and 30 percent in certain specialized workloads,\[8\] although companies responsible for software correction of the exploit are reporting minimal impact from general benchmark testing.\[9\]

Meltdown was issued a Common Vulnerabilities and Exposures ID of CVE-2017-5754\[2\], also known as Rogue Data Cache Load,\[2\] in January 2018. It was disclosed in conjunction with another exploit, Spectre, with which it shares some, but not all characteristics. The Meltdown and Spectre vulnerabilities are considered "catastrophic"
The Fallout

You realize it is something big when...

- it is in the news, all over the world
- you get a Wikipedia article in multiple languages
- there are comics, including xkcd
- you get a lot of Twitter followers after Snowden mentioned you

Spectre (security vulnerability)

From Wikipedia, the free encyclopedia

Spectre is a vulnerability that affects modern microprocessors that perform branch prediction. On most processors, the speculative execution resulting from a branch misprediction may leave observable side effects that may reveal private data to attackers. For example, if the pattern of memory accesses performed by such speculative execution depends on private data, the resulting state of the data cache constitutes a side channel through which an attacker may be able to extract information about the private data using a timing attack.

Two Common Vulnerabilities and Exposures IDs related to Spectre, CVE-2017-5753 and CVE-2017-5715 (bounds check bypass) and CVE-2017-5715d (branch target injection), have been issued. JIT engines used for JavaScript were found vulnerable. A website can read data stored in the browser for another website, or the browser’s memory itself.

Several procedures to help protect home computers and related devices from the Spectre (and Meltdown) security vulnerabilities have been published. Spectre patches have been reported to significantly slow down performance, especially on older computers; on the newer 8th generation Core platforms, benchmark performance drops of 2–14 percent have been measured. Meltdown patches may also produce performance loss. On January 18, 2018, unwanted reboots, even for newer Intel chips, due to
You realize it is something big when...

- it is in the *news*, all over the world
- you get a *Wikipedia* article in multiple languages
- there are *comics*, including xkcd
The Fallout

You realize it is something big when:

• it is in the news, all over the world
• you get a Wikipedia article in multiple languages
• there are comics, including xkcd
• you get a lot of Twitter followers after you mentioned them.

The Phantom Trolley isn’t supposed to touch anyone, but it turns out you can still use it to do stuff, and it can drive through walls.

That sounds bad. Honestly, I’ve been assuming we were doomed ever since I learned about rowhammer.

What’s that?

If you toggle a row of memory cells on and off really fast, you can use electrical interference to flip nearby bits and—do I just suck at computers?

Yup. Especially shared ones.

So you’re saying the cloud is full of phantom trolleys armed with hammers.

...Yes, that is exactly right.

Okay, ill, uh... install updates? Good idea.
You realize it is something big when...

- it is in the news, all over the world
- you get a Wikipedia article in multiple languages
- there are comics, including xkcd
- you get a lot of Twitter followers after Snowden mentioned you

Daniel Gruss — Graz University of Technology
You realize it is something big when...

- it is in the news, all over the world
- you get a Wikipedia article in multiple languages
- there are comics, including xkcd
- you get a lot of Twitter follower after Snowden mentioned you
The Fallout

You realize it is something big when...

• it is in the news, all over the world
• you get a Wikipedia article in multiple languages
• there are comics, including xkcd
• you get a lot of Twitter follower after Snowden mentioned you

Edward Snowden
@Snowden

You may have heard about @Intel's horrific #Meltdown bug. But have you watched it in action? When your computer asks you to apply updates this month, don't click "not now."

(via spectreatack.com & @misc0110)

Daniel Gruss — Graz University of Technology
Revolutionary concept!

Store your food at home, never go to the grocery store during cooking.

Can store ALL kinds of food.

ONLY TODAY INSTEAD OF $1,300

ORDER VIA PHONE: +555 12345
printf("%d", i);
printf("%d", i);
printf("%d", i);
printf("%d", i);
printf("%d", i);
CPU Cache

printf("%d", i);
printf("%d", i);

Cache miss

Request
printf("%d", i);
printf("%d", i); [Cache miss] Request [Response]
printf("%d", i);
printf("%d", i);
printf("%d", i);
printf("%d", i);
printf("%d", i);

Cache miss
Cache hit

Request
Response
CPU Cache

```c
printf("%d", i);
printf("%d", i);
```

Cache miss

Cache hit

DRAM access, slow

Request

Response

Daniel Gruss — Graz University of Technology
CPU Cache

- **DRAM access, slow**
- **Cache miss**

```c
printf("%d", i);
```

- **Cache hit**

```
printf("%d", i);
```

- **No DRAM access, much faster**

```
printf("%d", i);
```

- **Request**
- **Response**

Daniel Gruss — Graz University of Technology
Flush+Reload

Shared Memory

ATTACKER

VICTIM

flush
access

flush
access
Flush+Reload

ATTACKER
flush
access

Shared Memory

VICTIM
cached
access

cached

Daniel Gruss — Graz University of Technology
Flush+Reload

ATTACKER

flush

Shared Memory

VICTIM

access

access
Flush+Reload

ATTACKER

flush

access

Shared Memory

access

VICTIM
Flush+Reload

ATTACKER

Shared Memory

VICTIM

flush
access

access
Flush+Reload

Shared Memory

ATTACKER

flush
access

VICTIM

access
Flush+Reload

ATTACKER

Shared Memory

VICTIM

flush

access

flush

access

Shared Memory

Daniel Gruss — Graz University of Technology
Flush+Reload

ATTACKER

flush

access

fast if victim accessed data,
slow otherwise

Shared Memory

VICTIM

access

4

Daniel Gruss — Graz University of Technology
Memory Access Latency

Number of accesses

Access time [CPU cycles]

Cache Hits
Cache Misses

Daniel Gruss — Graz University of Technology
Hello from the other side (DEMO):
Video streaming over cache covert channel
Back to Work
7. Serve with cooked and peeled potatoes
Wait for an hour
Wait for an hour
1. Wash and cut vegetables

2. Pick the basil leaves and set aside

3. Heat 2 tablespoons of oil in a pan

4. Fry vegetables until golden and softened
1. Wash and cut vegetables

2. Pick the basil leaves and set aside

3. Heat 2 tablespoons of oil in a pan

4. Fry vegetables until golden and softened

Parallelize Dependency
int width = 10, height = 5;

float diagonal = sqrt(width * width
                        + height * height);
int area = width * height;

printf("Area %d x %d = %d\n", width, height, area);
```c
int width = 10, height = 5;

float diagonal = sqrt(width * width + height * height);
int area = width * height;

printf("Area %d x %d = %d\n", width, height, area);
```
char data = *(char*)0xffffffff81a000e0;
printf("%c\n", data);
char data = *(char*)0xffffffff81a000e0;
printf("%c\n", data);

segfault at fffffffff81a000e0 ip
  00000000000400535  
    sp 00007ffce4a80610 error 5 in reader
Adapted code

*(volatile char*) 0;
array[84 * 4096] = 0; // unreachable
Flush+Reload over all pages of the array

Access time [cycles]

Page
Flush+Reload over all pages of the array

This also works on AMD and ARM!
• Combine the two things

```c
char data = *(char*)0xfffffffff81a000e0;
array[data * 4096] = 0;
```
... Meltdown actually works.
I SHIT YOU NOT
THERE WAS KERNEL MEMORY ALL OVER THE TERMINAL
CAN YOU
ENHANCE THAT
meltdown@meltdown ~/ppm2 % taskset 1 ./imgdump 0x375a00000 14919 > output.flif
Reading from 0xffff880375a00000
Leaking Passwords from your Password Manager

Daniel Gruss — Graz University of Technology
AND IN OTHER NEWS...

WE'RE ALL DOOMED, SANDRA.
HOW ABOUT THE WEATHER?
Not so fast...
Take the kernel addresses...

- Kernel addresses in user space are a problem
Take the kernel addresses...

- Kernel addresses in user space are a problem
- Why don’t we take the kernel addresses...
...and remove them and remove them if not needed?
...and remove them if not needed?

• User accessible check in hardware is not reliable
Kernel Address Isolation to have Side channels Efficiently Removed
Kernel Address Isolation to have Side channels Efficiently Removed
KAISER Patches

• Our patch
• Adopted in Linux
• Adopted in Windows
• Adopted in OSX/iOS
• Now in every computer

Daniel Gruss — Graz University of Technology
Our patch

Adopted in Linux

Adopted in Windows

Adopted in OSX/iOS

now in every computer
KAISER Patches

- Our patch
- Adopted in Linux

- Adopted in Windows
KAISER Patches

• Our patch
• Adopted in Linux

• Adopted in Windows

• Adopted in OSX/iOS
KAISER Patches

• Our patch
• Adopted in Linux

• Adopted in Windows

• Adopted in OSX/iOS

→ now in every computer

Daniel Gruss — Graz University of Technology
»A table for 6 please«
Speculative Cooking
» A table for 6 please «
index = 0;

char* data = "textKEY";

if (index < 4)
    LUT[data[index] * 4096]
else
    0
index = 0;

char* data = "textKEY";

if (index < 4)
    then
        Prediction
        LUT[data[index] * 4096]
    else
        0
index = 0;

char* data = "textKEY";

if (index < 4)
    then
        LUT[data[index] * 4096]
    else
        Speculate
        Prediction
        0
index = 0;

char* data = "textKEY";

if (index < 4)
    then
        LUT[data[index] * 4096]
    else
        Prediction

0
index = 1;

char* data = "textKEY";

if (index < 4)
    Prediction
else
    LUT[data[index] * 4096]

0
index = 1;

char* data = "textKEY";

if (index < 4)
    LUT[data[index] * 4096]
else
    0
index = 1;

char* data = "textKEY";

if (index < 4)

then

Speculate

LUT[data[index] * 4096]

else

Prediction

0

Daniel Gruss — Graz University of Technology
index = 1;

char* data = "textKEY";

if (index < 4)
{
    LUT[data[index] * 4096]
}
else
{
    0
}
index = 2;

char* data = "textKEY";

if (index < 4)

then

LUT[data[index] * 4096]

else

0
index = 2;

char* data = "textKEY";

if (index < 4)

then

LUT[data[index] * 4096]

else

0

Prediction
index = 2;

char* data = "textKEY";

if (index < 4)
then
LUT[data[index] * 4096]
else
0
index = 2;

char* data = "textKEY";

if (index < 4)
then
LUT[data[index] * 4096]
else
Prediction

0
index = 3;

char* data = "textKEY";

if (index < 4)
    LUT[data[index] * 4096]
else
    0
index = 3;

char* data = "textKEY";

if (index < 4)

then

Prediction

else

LUT[data[index] * 4096]

0
index = 3;

char* data = "textKEY";

if (index < 4)
then

LUT[data[index] * 4096]

else

Prediction

0
index = 3;

char* data = "textKEY";

if (index < 4)
    then
    LUT[data[index] * 4096]
    else
    Prediction
    0

Daniel Gruss — Graz University of Technology
index = 4;

char* data = "textKEY";

if (index < 4)
    Prediction
  then
    LUT[data[index] * 4096]
  else
    0
index = 4;

char* data = "textKEY";

if (index < 4)
then
Prediction

LUT[data[index] * 4096]

else

0
index = 4;
char* data = "textKEY";

if (index < 4)
  Speculate
  then
  LUT[data[index] * 4096]
  Prediction
else
  0
index = 4;

char* data = "textKEY";

if (index < 4)
then
LUT[data[index] * 4096]
else
Prediction
Execute
0
index = 5;

char* data = "textKEY";

if (index < 4)

then

LUT[data[index] * 4096]

else

Prediction

0

Daniel Gruss — Graz University of Technology
index = 5;

char* data = "textKEY";

if (index < 4)

then

LUT[data[index] * 4096]

else

Prediction

0

Daniel Gruss — Graz University of Technology
Spectre (variant 1)

```c
index = 5;

char* data = "textKEY";

if (index < 4)
  Speculate
  then
    LUT[data[index] * 4096]
  else
    Prediction
    0
else
  Prediction
  0
```

Daniel Gruss — Graz University of Technology
index = 5;

data = "textKEY";

if (index < 4)
    then
        Prediction
        LUT[data[index] * 4096]
else
    Execute
index = 6;

char* data = "textKEY";

if (index < 4)
    Prediction
else
    LUT[data[index] * 4096]  0
index = 6;

char* data = "textKEY";

if (index < 4)

then

LUT[data[index] * 4096]

else

0

Prediction
Spectre (variant 1)

```c
index = 6;

char* data = "textKEY";

if (index < 4)

LUT[data[index] * 4096]

then

Speculate

else

Prediction

0
```

Daniel Gruss — Graz University of Technology
index = 6;

char* data = "textKEY";

if (index < 4) then
    LUT[data[index] * 4096]
else
    Prediction

Execute

0
BLOCKCHAIN
Computer Architecture Today

Informing the broad computing community about current activities, advances and future directions in computer architecture.

Let’s Keep it to Ourselves: Don’t Disclose Vulnerabilities

by Gus Uht on Jan 31, 2019 | Tags: Opinion, Security

CONTRIBUTE

Editor: Alvin R. Lebeck
Associate Editor: Vijay Janapa Reddi

Contribute to Computer Architecture Today
We have ignored microarchitectural attacks for many many years:
We have ignored microarchitectural attacks for many many years:

- attacks on crypto
We have ignored microarchitectural attacks for many many years:

- attacks on crypto → “software should be fixed”
What do we learn from it?

We have ignored microarchitectural attacks for many many years:

- attacks on crypto → “software should be fixed”
- attacks on ASLR
What do we learn from it?

We have ignored microarchitectural attacks for many many years:

- attacks on crypto → “software should be fixed”
- attacks on ASLR → “ASLR is broken anyway”
What do we learn from it?

We have ignored microarchitectural attacks for many many years:

- attacks on crypto → “software should be fixed”
- attacks on ASLR → “ASLR is broken anyway”
- attacks on SGX and TrustZone
We have ignored microarchitectural attacks for many many years:

- attacks on crypto → “software should be fixed”
- attacks on ASLR → “ASLR is broken anyway”
- attacks on SGX and TrustZone → “not part of the threat model”
We have ignored microarchitectural attacks for many many years:

- attacks on crypto → “software should be fixed”
- attacks on ASLR → “ASLR is broken anyway”
- attacks on SGX and TrustZone → “not part of the threat model”
- Rowhammer attacks
We have ignored microarchitectural attacks for many many years:

- attacks on crypto → “software should be fixed”
- attacks on ASLR → “ASLR is broken anyway”
- attacks on SGX and TrustZone → “not part of the threat model”
- Rowhammer attacks → “only affects cheap sub-standard modules”
We have ignored microarchitectural attacks for many many years:

- attacks on crypto → “software should be fixed”
- attacks on ASLR → “ASLR is broken anyway”
- attacks on SGX and TrustZone → “not part of the threat model”
- Rowhammer attacks → “only affects cheap sub-standard modules”

→ for years we solely optimized for performance
After learning about a side channel you realize:
After learning about a side channel you realize:

- the side channels were documented in the Intel manual
When you read the manuals...

After learning about a side channel you realize:

- the side channels were documented in the Intel manual
- only now we understand the implications
What do we learn from it?

Motor Vehicle Deaths per Year (normalized by US population)

Daniel Gruss — Graz University of Technology
Conclusions

A unique chance to

- for the security industry and academia to grow up
- find good trade-offs between security and performance, efficiency, and complexity
- like the health sector, learn to cope with diseases
Software-based Microarchitectural Attacks: What do we learn from Meltdown and Spectre?

Daniel Gruss
March 26, 2019
Graz University of Technology