Meltdown, Spectre, ZombieLoad

Daniel Gruss

May 16, 2019

Graz University of Technology

The Fallout

You realize it is something big when...

The Fallout

You realize it is something big when...

- it is in the news, all over the world


WASHINGTON, D.C.
FOX
BUSINESS
WASHINGTON, D.C.
WINTER STORM
NEWS
› ALERT
INTEL REVEALS DESIGN FLAW THAT
COULD ALLOW HACKERS TO ACCESS DATA
FOX
BUSINESS
NETWORK
@FOXBUSINESS


DEVELOPING STORY
COMPUTER CHIP FLAWS IMPACT BILLIONS OF DEVICES
LIVE
CNN
DAX ▲164.69
NEWS STREAM


GLOBAL
COMPUTER CHIP SCARE
The bugs are known as 'Spectre' and 'Meltdown'
BBC WORLD NEWS
£:HK$ 10.58
EURO:£ 0.891


SECURITY FLAW REVEALED
Intel (Prev)
45.26 -1.59 [-3.39%]
Intel (After Hours)
44.85 -0.41 [-0.91%]
CAPITAL CONNECTION
SHROUT: ISSUE NOT UNIQUE TO INTEL, BUT IT'S AFFECTED THE MOST
CNBC


ABO SHOP AKADEMIE JOBS MEHR
E-PAPER AUDIO APPS ARCHIV ANMELDEN
ZEIT ONLINE
Suche
Politik Gesellschaft Wirtschaft Kultur Wissen Digital Campus Arbeit Entdecken Sport ZEITmagazin mehr
#D18
AUS DER SERIE
Was bewegt
Daniel Gruss
Der Kernschmelzer
Daniel Gruss hat eine schwere Sicherheitslücke in Computerchips entdeckt. Warum gelingt dem Informatiker, woran die Hersteller scheitern?
Von Jens Tönnesmann
7. März 2018, 16:46 Uhr / Editiert am 9. März 2018, 20:11 Uhr / 26 Kommentare
AUS DER
ZEIT NR. 11/2018
DIE ZEIT

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

Not logged in Talk Contributions Create account Log in

WIKIPEDIA
The Free Encyclopedia

Article Talk | Read Edit View history | Search Wikipedia

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]

Contents [hide]

1 Background on KAISER
2 Meltdown vulnerability and KPTI
3 Implementation
4 References

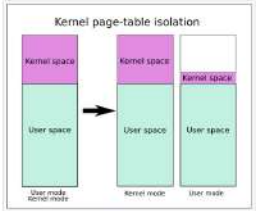
Main page
Contents
Featured content
Current events
Random article
Donate to Wikipedia
Wikipedia store

Interaction

Help
About Wikipedia
Community portal
Recent changes
Contact page

Tools

What links here
Related changes



One set of page table for use in kernel mode includes both kernel-space and user-space. The second set of page table for use in user mode

Not logged in Talk Contributions Create account Log in

WIKIPEDIA
The Free Encyclopedia

Article Talk | Read Edit View history | Search Wikipedia

Meltdown (security vulnerability)

From Wikipedia, the free encyclopedia

Main page
Contents
Featured content
Current events
Random article
Donate to Wikipedia
Wikipedia store

Interaction
Help
About Wikipedia
Community portal
Recent changes
Contact page

Tools
What links here
Related changes
Upload file

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, 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 logo used by the team that discovered the vulnerability

Not logged in Talk Contributions Create account Log in

WIKIPEDIA
The Free Encyclopedia

Article Talk Read Edit View history Search Wikipedia

Spectre (security vulnerability)

From Wikipedia, the free encyclopedia

Main page
Contents
Featured content
Current events
Random article
Donate to Wikipedia
Wikipedia store

Interaction
Help
About Wikipedia
Community portal
Recent changes
Contact page

Tools
What links here
Related changes
Upload file

Spectre is a vulnerability that affects modern microprocessors that perform branch prediction.[1][2][3] 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.[4][5][6]

A logo created for the vulnerability, featuring a ghost with a branch

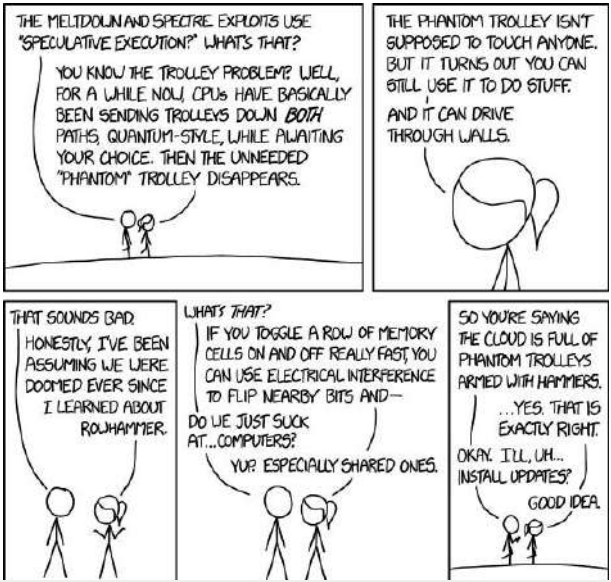
Two Common Vulnerabilities and Exposures IDs related to Spectre, CVE-2017-5753 (bounds check bypass) and CVE-2017-5715 (branch target injection), have been issued.[7] JIT engines used for JavaScript were found vulnerable. A website can read data stored in the browser for another website, or the browser's memory itself.[8]

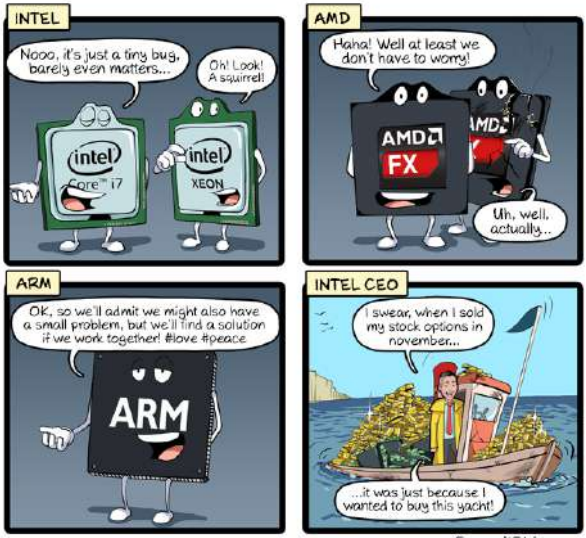
Several procedures to help protect home computers and related devices from the Spectre (and Meltdown) security vulnerabilities have been published.[9][10][11][12] Spectre patches have been reported to significantly slow down performance, especially on older computers; on the newer 8th generation Core platforms, benchmark performance drops of 2–14 percent have been measured.[13] Meltdown patches may also produce performance loss.[5][14][15] On January 18, 2018, unwanted reboots, even for newer Intel chips, due to

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

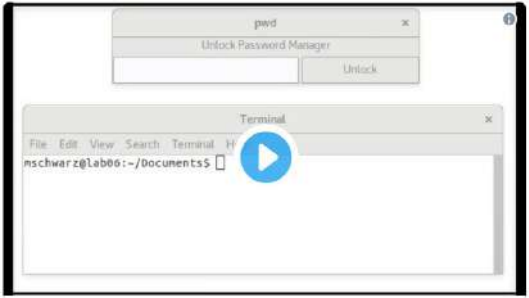

THE MELTDOWN AND SPECTRE EXPLOITS USE "SPECULATIVE EXECUTION?" WHAT'S THAT?
YOU KNOW THE TROLLEY PROBLEM? WELL, FOR A WHILE NOW, CPUs HAVE BASICALLY BEEN SENDING TROLLEYS DOWN BOTH PATHS, QUANTUM-STYLE, WHILE AWAITING YOUR CHOICE. THEN THE UNNEEDED "PHANTOM" TROLLEY DISAPPEARS.
THE PHANTOM TROLLEY ISN'T 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 WE 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. I'LL, UH... INSTALL UPDATES?
GOOD IDEA.


INTEL
Nooo, it's just a tiny bug, barely even matters...
Oh! Look! A squirrel!
intel
Core™ i7
intel
XEON
AMD
Haha! Well at least we don't have to worry!
AMD
FX
Uh, well, actually...
ARM
OK, so we'll admit we might also have a small problem, but we'll find a solution if we work together! #love #peace
ARM
INTEL CEO
I swear, when I sold my stock options in november...
...it was just because I wanted to buy this yacht!
CommitStrip.com

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 spectreattack.com & @misc0110)

23:37 - 4. Jan. 2018

152 6.547 6.512

amazon.com
Prime+Probe

ANOTHER FLIP IN THE ROW

ROWHAMMER.JS

FANTASTIC TIMERS

AND WHERE TO FIND THEM

HIGH-RESOLUTION MICROARCHITECTURAL ATTACKS IN JAVASCRIPT

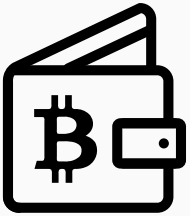
JavaScript zero

REAL JavaScript AND ZERO SIDE-CHANNEL ATTACKS

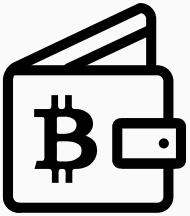
National Geographic


breaking bitcoin
ABOUT
AGENDA
TICKETS
VENUE
SPONSORS
CONTACT US
BREAKING BITCOIN
Conference
Paris 9-10 Sep 2017

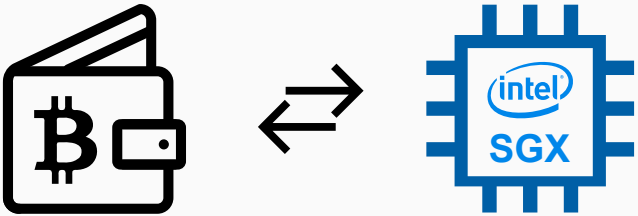
Stealing Bitcoins?



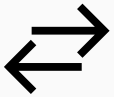
Stealing Bitcoins?

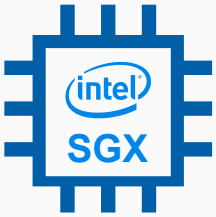


Stealing Bitcoins?

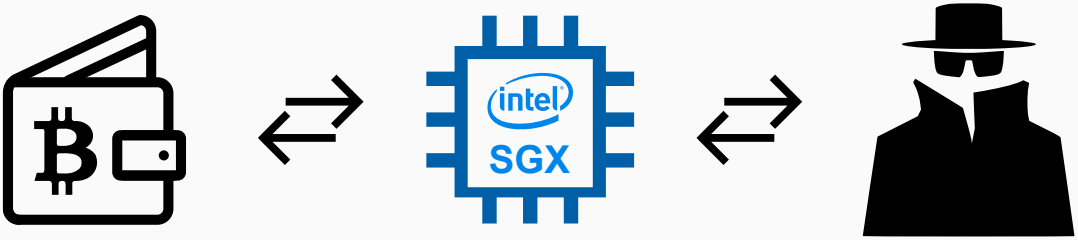


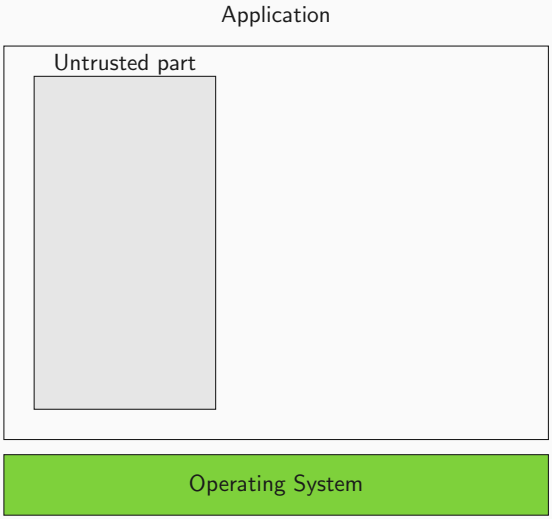
Stealing Bitcoins?

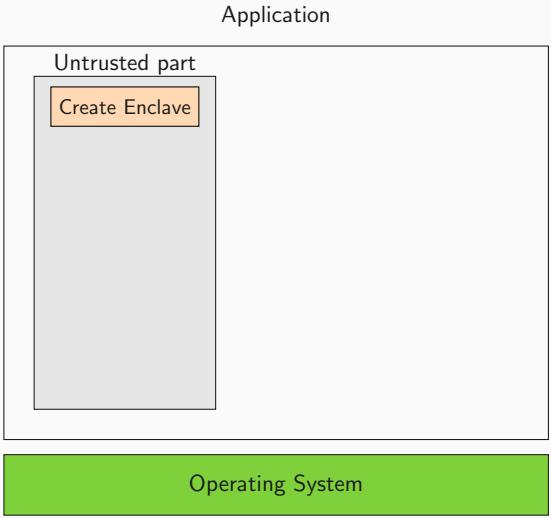


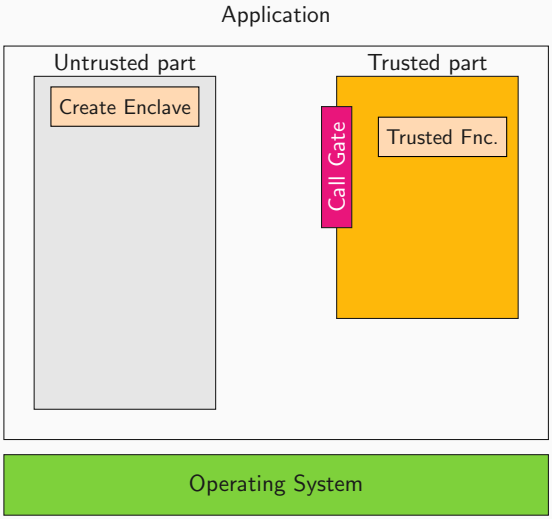


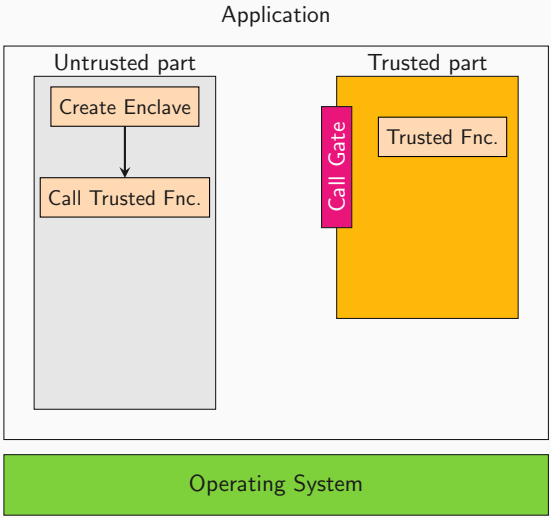
Stealing Bitcoins?

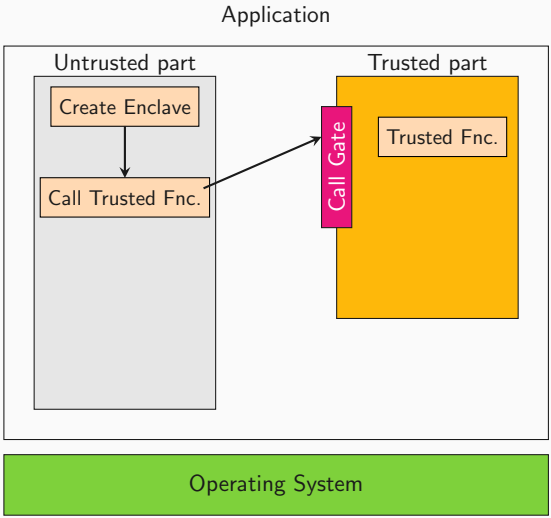


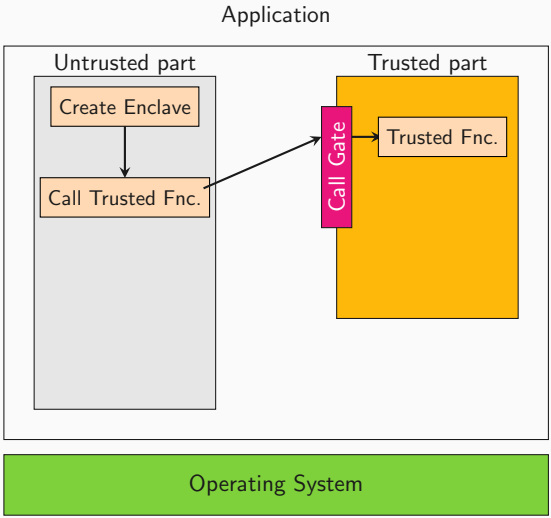

Application
Untrusted part
Operating System

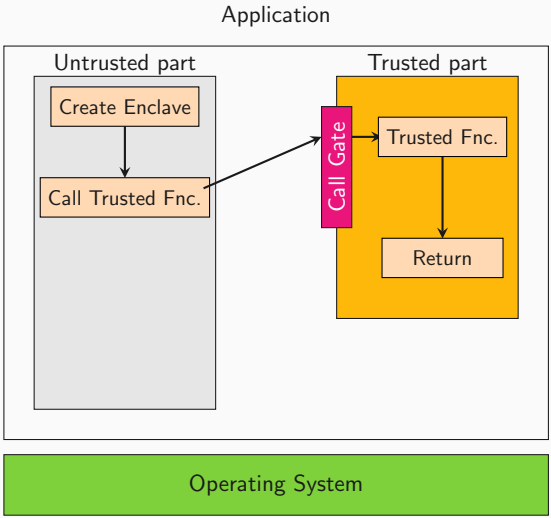

Application
Untrusted part
Create Enclave
Operating System

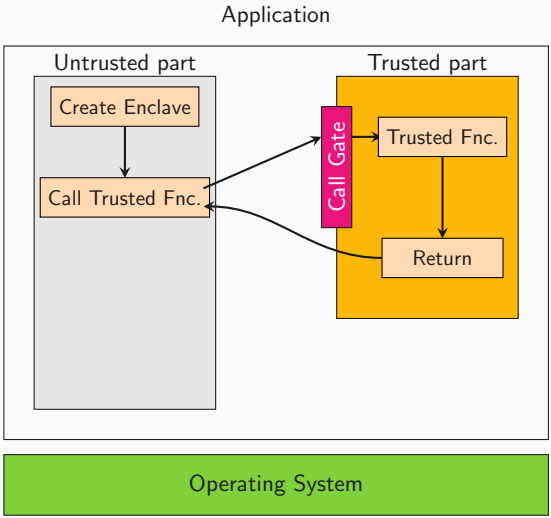

Application
Untrusted part
Create Enclave
Trusted part
Call Gate
Trusted Fnc.
Operating System

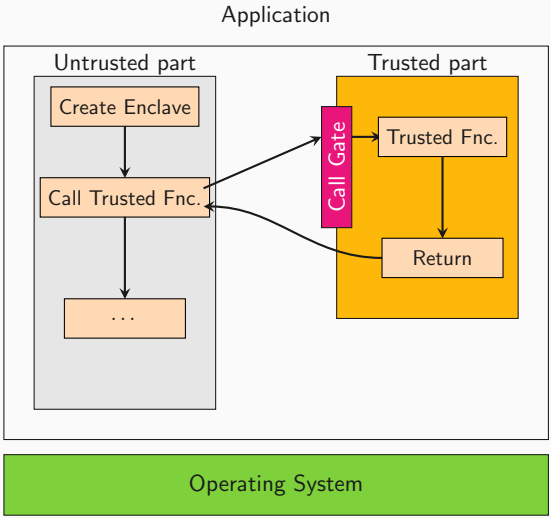

Application
Untrusted part
Create Enclave
Call Trusted Fnc.
Trusted part
Call Gate
Trusted Fnc.
Operating System

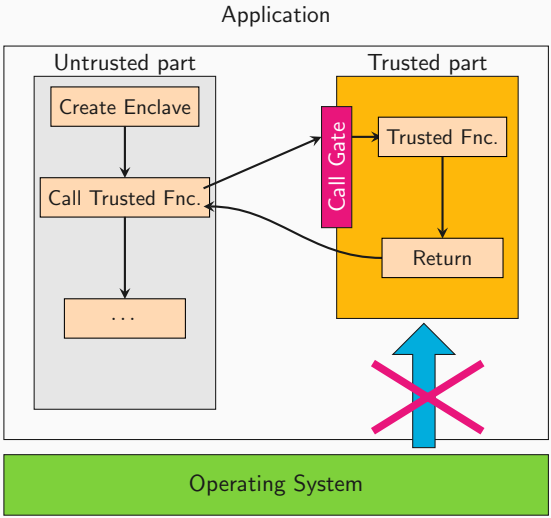

Application
Untrusted part
Trusted part
Create Enclave
Call Trusted Fnc.
Call Gate
Trusted Fnc.
Operating System


Application
Untrusted part
Trusted part
Create Enclave
Call Trusted Fnc.
Call Gate
Trusted Fnc.
Operating System


Application
Untrusted part
Trusted part
Create Enclave
Call Trusted Fnc.
Call Gate
Trusted Fnc.
Return
Operating System


Application
Untrusted part
Trusted part
Create Enclave
Call Trusted Fnc.
Call Gate
Trusted Fnc.
Return
Operating System


Application
Untrusted part
Trusted part
Create Enclave
Call Trusted Fnc.
...
Call Gate
Trusted Fnc.
Return
Operating System


Application
Untrusted part
Trusted part
Create Enclave
Call Trusted Fnc.
...
Call Gate
Trusted Fnc.
Return
Operating System

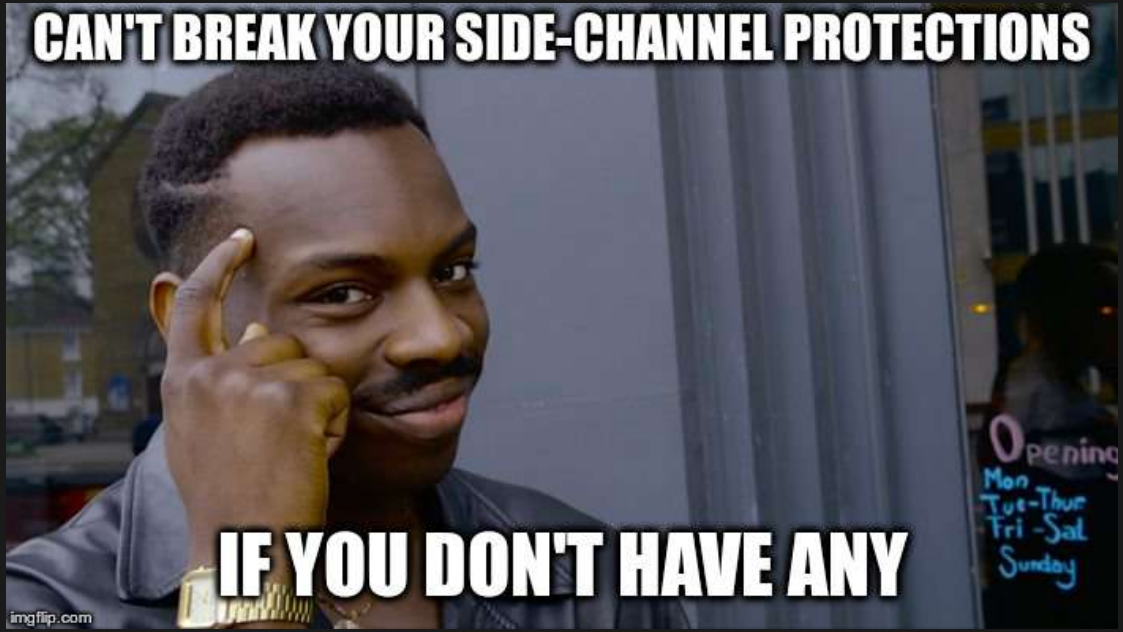
Protection from Side-Channel Attacks

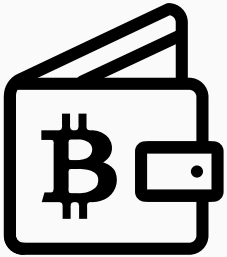
Protection from Side-Channel Attacks

Intel SGX does not provide explicit protection from side-channel attacks.

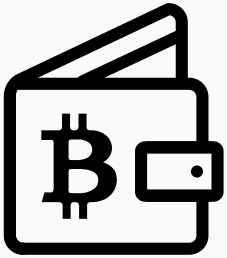
Protection from Side-Channel Attacks

Intel SGX does not provide explicit protection from side-channel attacks. It is the enclave developer's responsibility to address side-channel attack concerns.


CAN'T BREAK YOUR SIDE-CHANNEL PROTECTIONS
IF YOU DON'T HAVE ANY
Opening
Mon
Tue-Thur
Fri-Sat
Sunday
imgflip.com



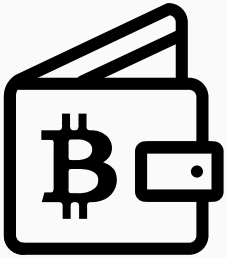
- Ledger SGX Enclave for blockchain applications
- BitPay Copay Bitcoin wallet
- Teechain payment channel using SGX



- Ledger SGX Enclave for blockchain applications
- BitPay Copay Bitcoin wallet
- Teechain payment channel using SGX

Teechain

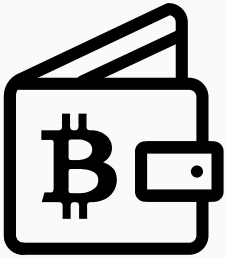
[...] We assume the TEE guarantees to hold



- Ledger SGX Enclave for blockchain applications
- BitPay Copay Bitcoin wallet
- Teechain payment channel using SGX

Teechain

[...] We assume the TEE guarantees to hold and do not consider side-channel attacks [5, 35, 46] on the TEE.

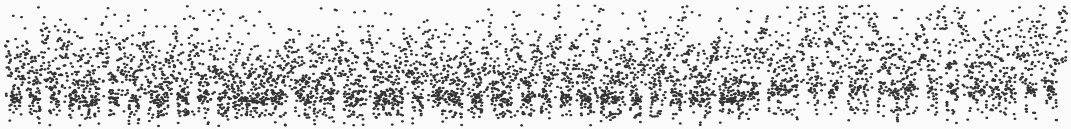


- Ledger SGX Enclave for blockchain applications
- BitPay Copay Bitcoin wallet
- Teechain payment channel using SGX

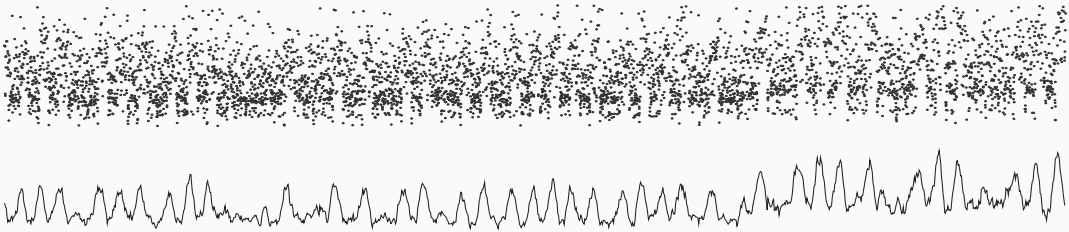
Teechain

[...] We assume the TEE guarantees to hold and do not consider side-channel attacks [5, 35, 46] on the TEE. Such attacks and their mitigations [36, 43] are outside the scope of this work. [...]

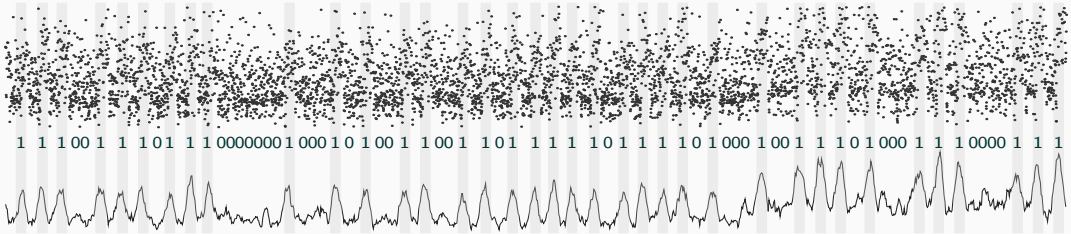
Raw Prime+Probe trace...

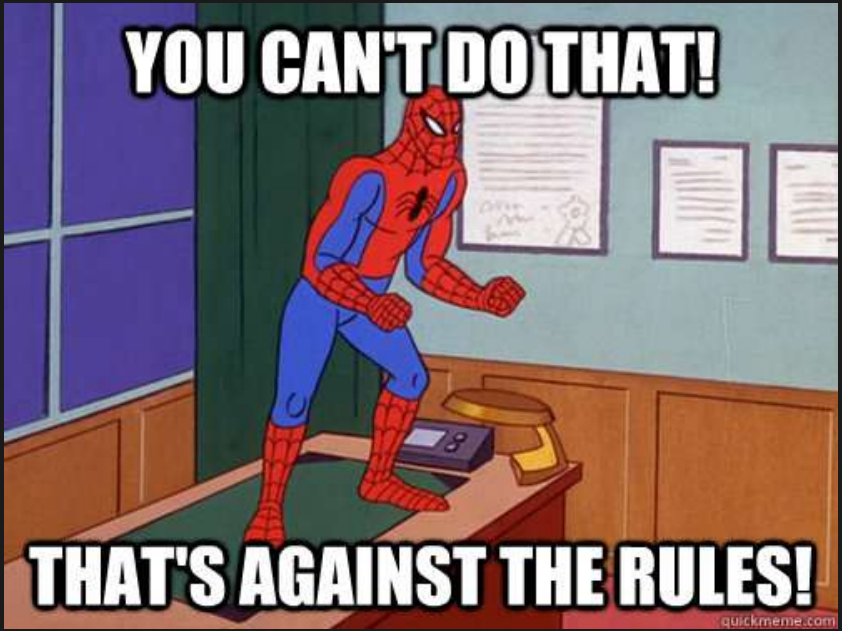


...processed with a simple moving average...



...allows to clearly see the bits of the exponent




YOU CAN'T DO THAT!
THAT'S AGAINST THE RULES!
quickmeme.com


WANT TO DISCUSS THREAT MODELS NOW?
38
imgflip.com

1337 4242

FOOD CACHE

Revolutionary concept!

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

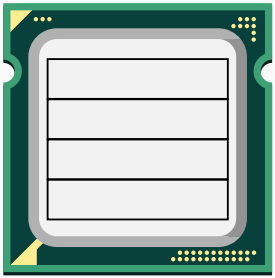
Can store **ALL** kinds of food.

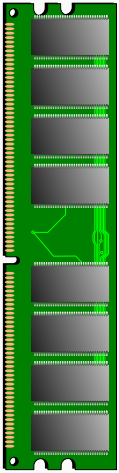
ONLY TODAY INSTEAD OF ~~$1,300~~

$1,299

ORDER VIA PHONE: +555 12345

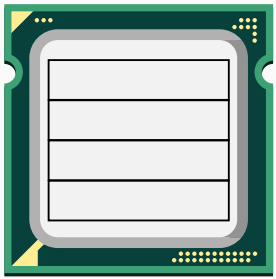
```
printf("%d", i);
printf("%d", i);
```

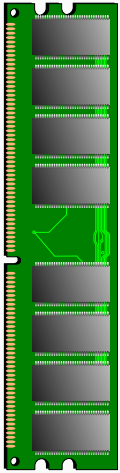




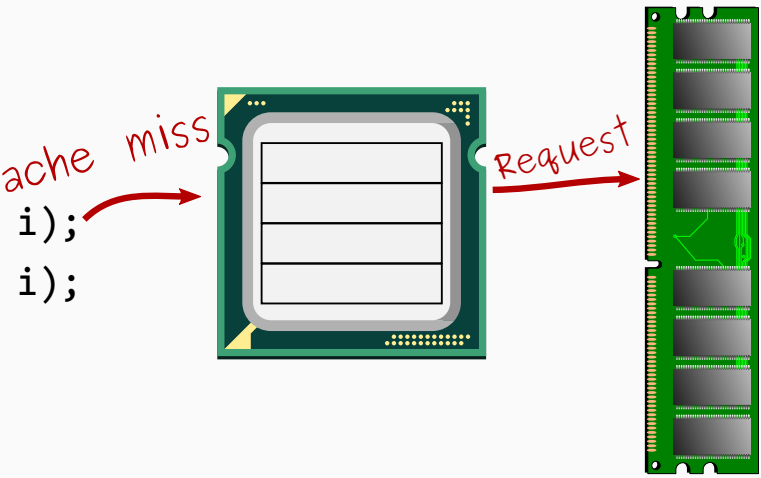
```
printf("%d", i);
printf("%d", i);
```

Cache miss

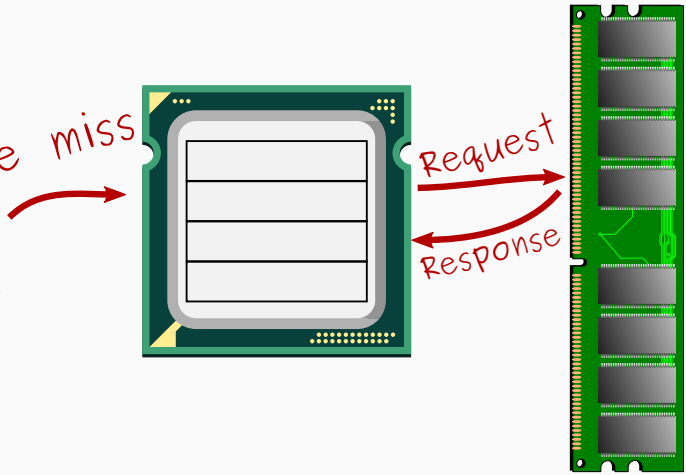




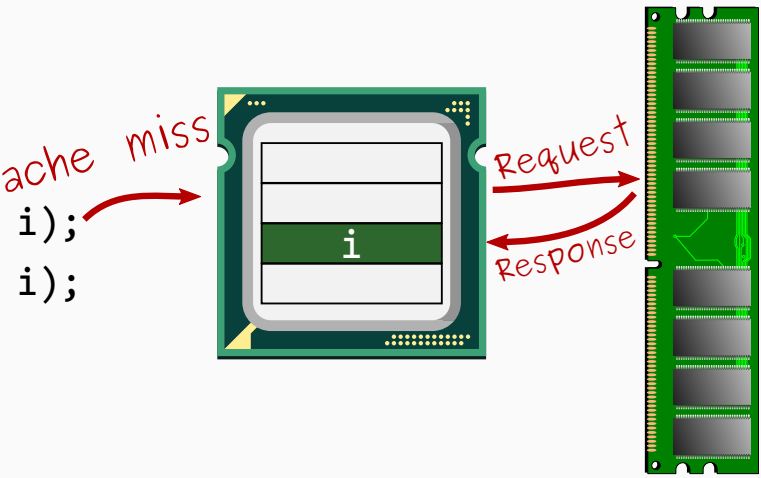
```
printf("%d", i);
printf("%d", i);
```

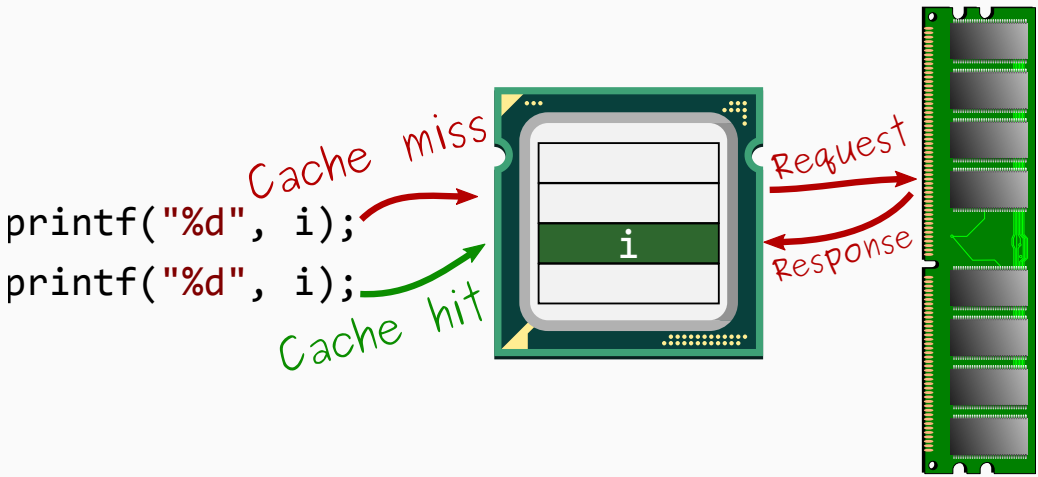


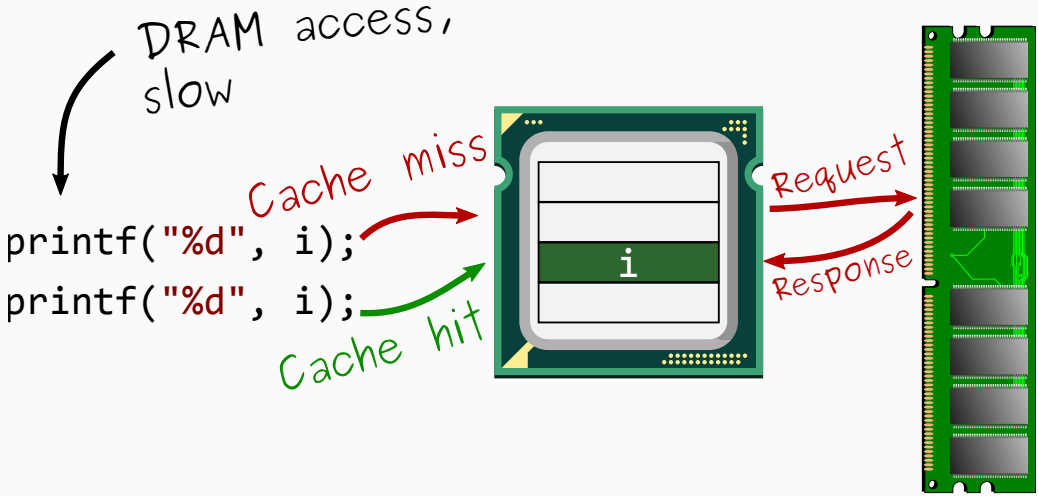
```
printf("%d", i);
printf("%d", i);
```

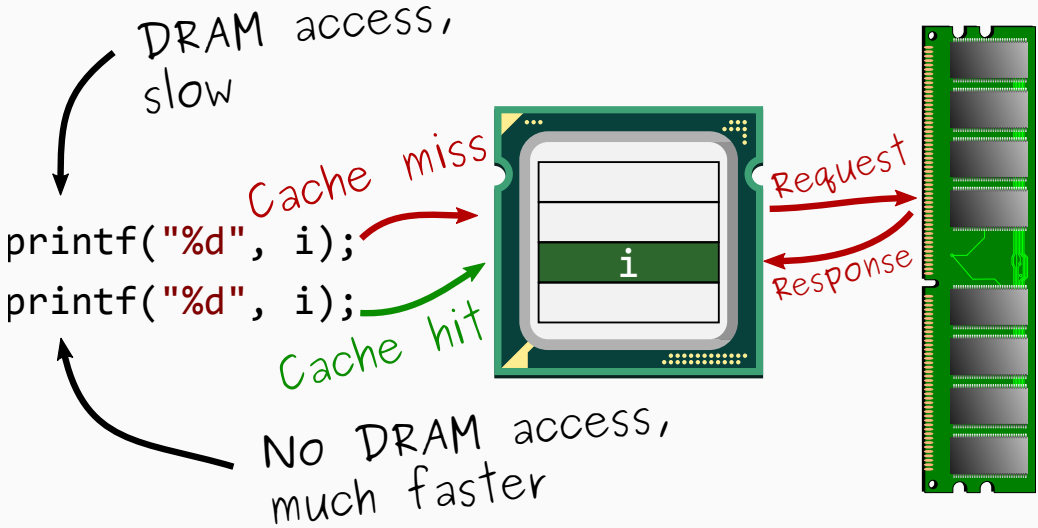


```
printf("%d", i);
printf("%d", i);
```




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


DRAM access, slow
printf("%d", i);
printf("%d", i);
Cache miss
Cache hit
i
Request
Response

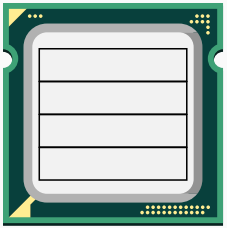

DRAM access, slow
printf("%d", i);
printf("%d", i);
Cache miss
Cache hit
No DRAM access, much faster
i
Request
Response

Shared Memory

ATTACKER

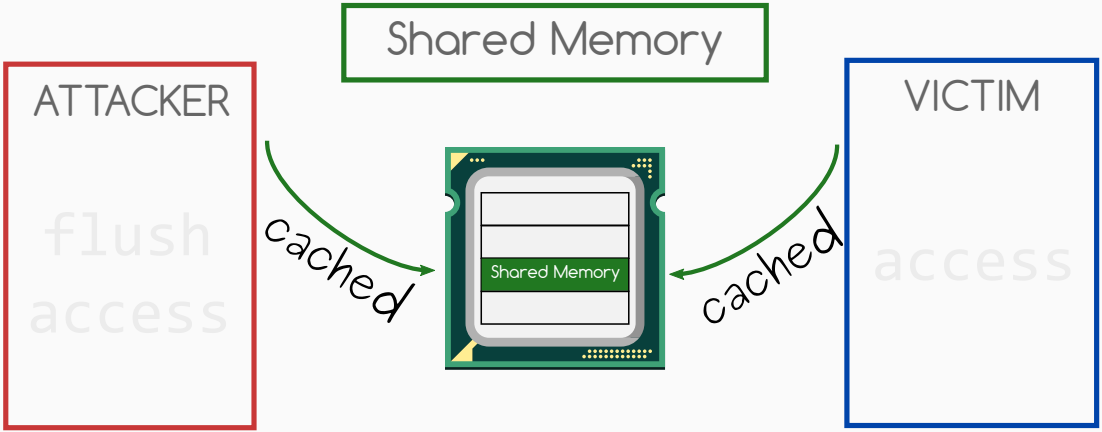
flush

access



VICTIM

access

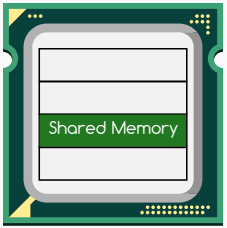

Shared Memory
ATTACKER
flush
access
cached
Shared Memory
cached
VICTIM
access

Shared Memory

ATTACKER

flush

access



VICTIM

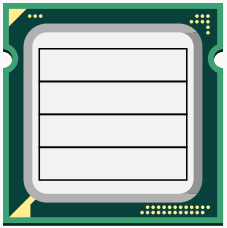
access

Shared Memory

ATTACKER

flush

access



VICTIM

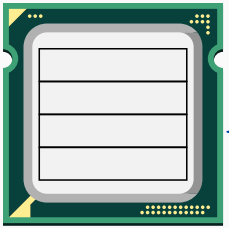
access

Shared Memory

ATTACKER

flush

access



VICTIM

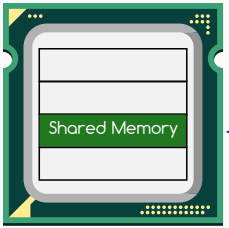
access

Shared Memory

ATTACKER

flush

access



VICTIM

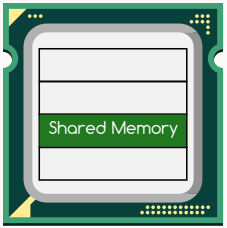
access

Shared Memory

ATTACKER

flush

access



VICTIM

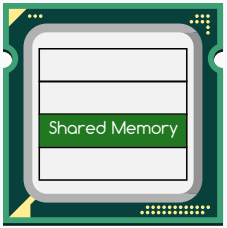
access

Shared Memory

ATTACKER

flush

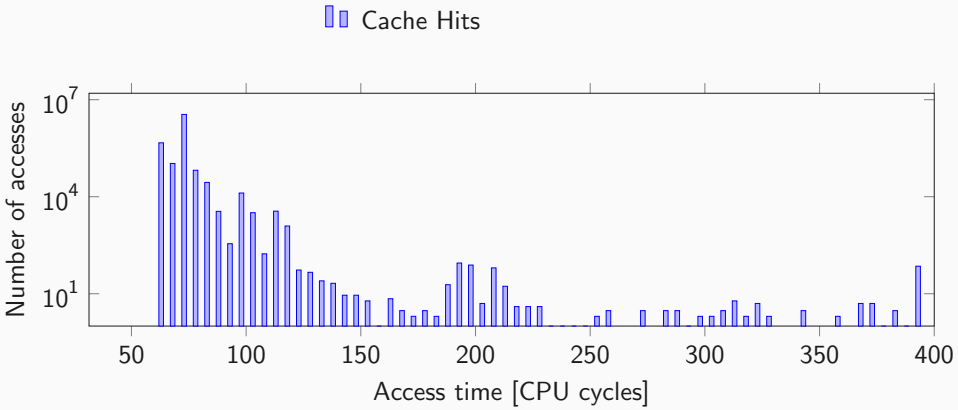
access

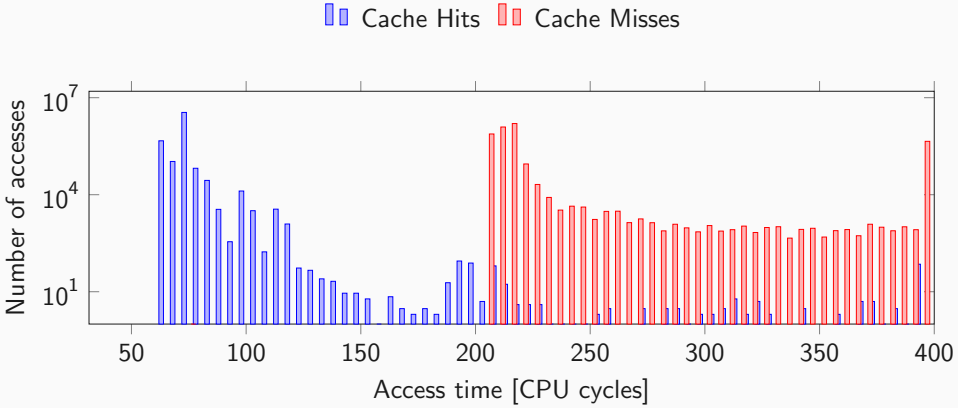


VICTIM

access

fast if victim accessed data,
slow otherwise


Cache Hits
Number of accesses
10^7
10^4
10^1
50
100
150
200
250
300
350
400
Access time [CPU cycles]


Cache Hits
Cache Misses
Number of accesses
10^7
10^4
10^1
50
100
150
200
250
300
350
400
Access time [CPU cycles]

Back to Work

7. Serve with cooked and peeled potatoes

Wait for an hour

Wait for an hour

LATENCY

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

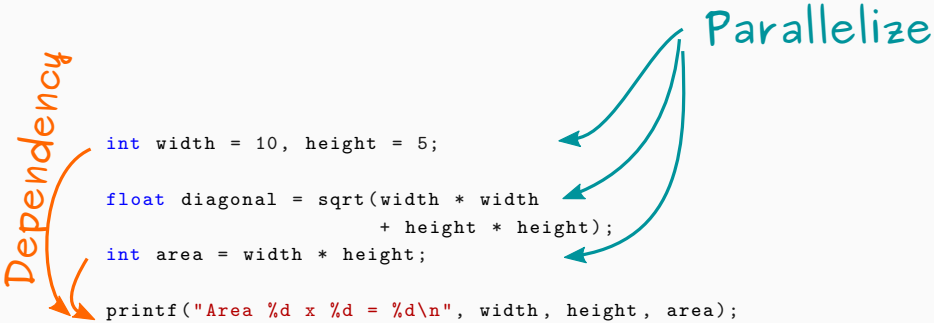
4. Fry vegetables until golden and softened

Dependency

Parallelize

Out-of-order Execution

```
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);
```


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);

```
char data = *(char*)0xffffffff81a000e0;
printf("%c\n", data);
```

```
char data = *(char*)0xffffffff81a000e0;
printf("%c\n", data);
```

```
segfault at ffffffff81a000e0 ip
   0000000000400535
      sp 00007ffce4a80610 error 5 in reader
```

Adapted code

```
*(volatile char*)0;
array[84 * 4096] = 0; // unreachable
```

Flush+Reload over all pages of the array

Flush+Reload over all pages of the array

This also works on AMD and ARM!

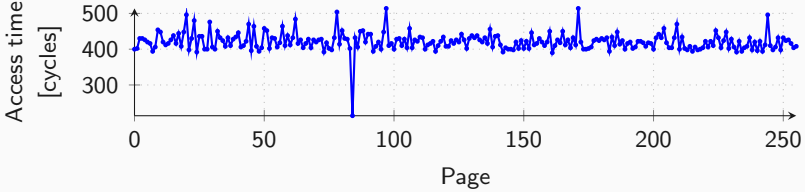
- Combine the two things

```
char data = *(char*)0xffffffff81a000e0;
array[data * 4096] = 0;
```

- Combine the two things

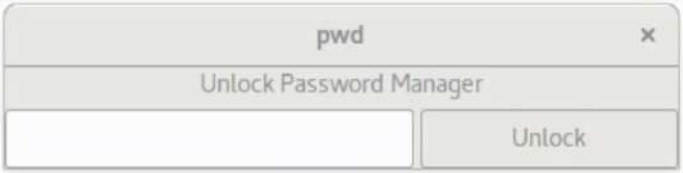
```
char data = *(char*)0xffffffff81a000e0;
array[data * 4096] = 0;
```

Flush+Reload again...



... Meltdown actually works.


I SHIT YOU NOT
THERE WAS KERNEL MEMORY ALL OVER THE TERMINAL

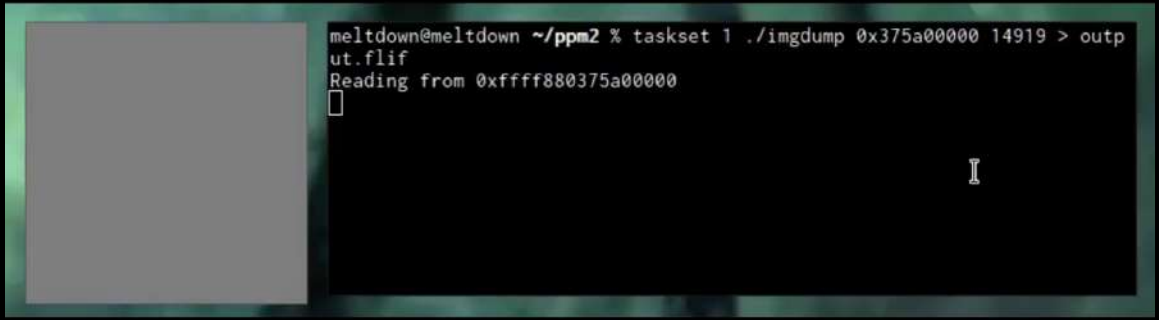


Terminal

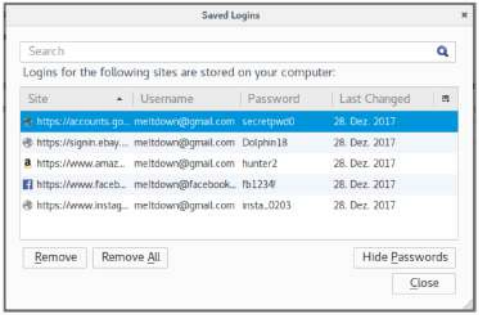
File Edit View Search Terminal Help

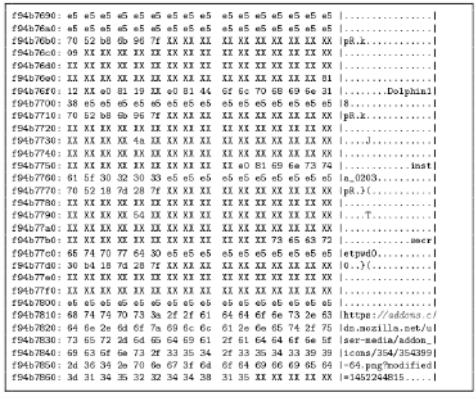
```
mschwarz@lab06:~/Documents$
```


CAN YOU
ENHANCE THAT


meltdown@meltdown ~/ppm2 % taskset 1 ./imgdump 0x375a00000 14919 > outp
ut.flif
Reading from 0xffff880375a00000

Leaking Passwords from your Password Manager






AND IN OTHER NEWS...
MELTDOWN
WE'RE ALL DOOMED, SANDRA.
HOW ABOUT THE WEATHER?
COMEDY CENTRAL

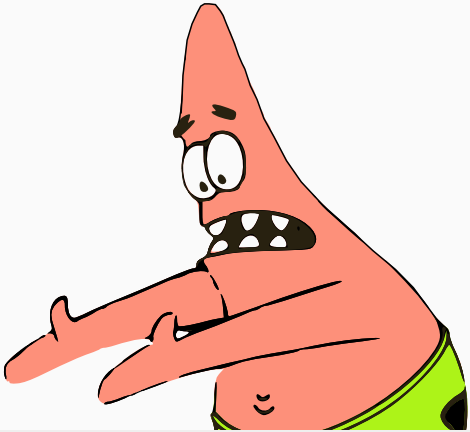
Not so fast...

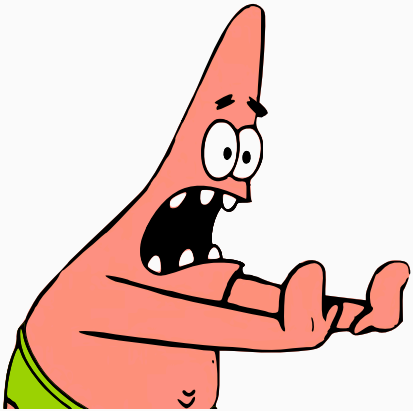
Take the kernel addresses...

- Kernel addresses in user space are a problem

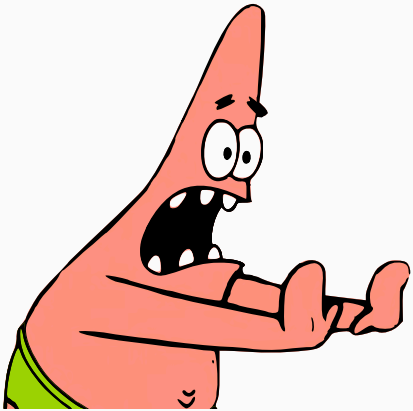
Take the kernel addresses...

- Kernel addresses in user space are a problem
- Why don't we take the kernel addresses...





- ...and remove them if not needed?



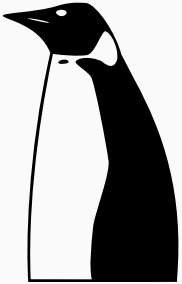
- ...and remove them if not needed?
- User accessible check in hardware is not reliable

Kernel **A**ddress **I**solation to have **S**ide channels **E**fficiently **R**emoved

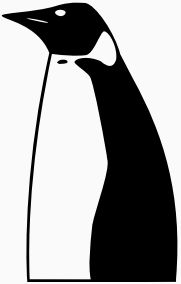
KAISER /ˈkʌɪzə/

1. [german] Emperor, ruler of an empire
2. largest penguin, emperor penguin

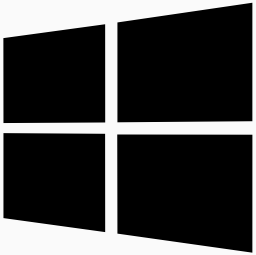
Kernel **A**ddress **I**solation to have **S**ide channels **E**fficiently **R**emoved



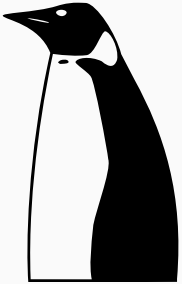
- Our patch
- Adopted in Linux



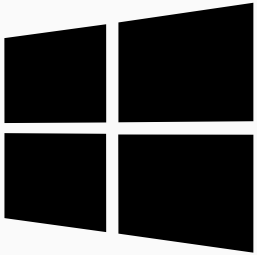
- Our patch
- Adopted in Linux



- Adopted in Windows

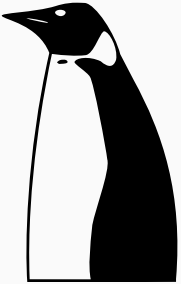


- Our patch
- Adopted in Linux

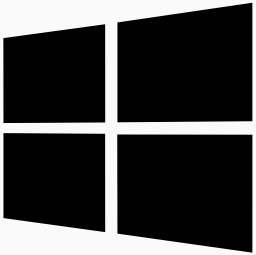


- Adopted in Windows

- Adopted in OSX/iOS



- Our patch
- Adopted in Linux

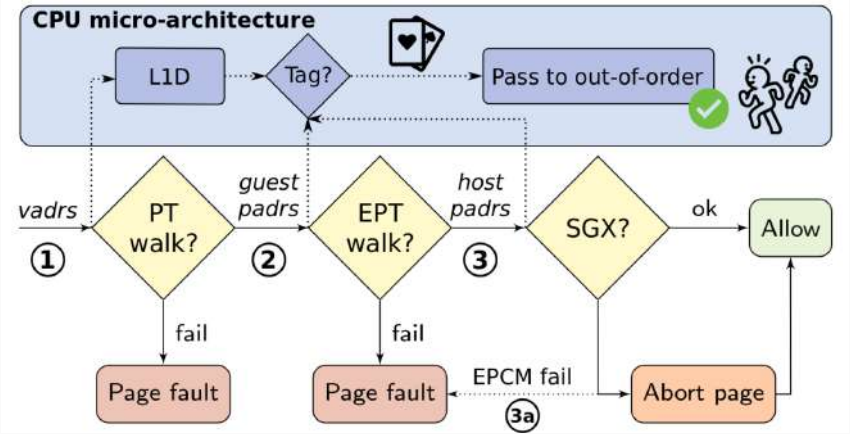


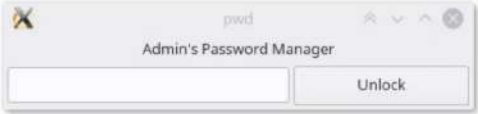
- Adopted in Windows

- Adopted in OSX/iOS

→ **now in every computer**

Problem solved?


CPU micro-architecture
L1D
Tag?
Pass to out-of-order
vadrs
PT walk?
guest padrs
EPT walk?
host padrs
SGX?
ok
Allow
1
2
3
fail
fail
Page fault
Page fault
EPCM fail
3a
Abort page



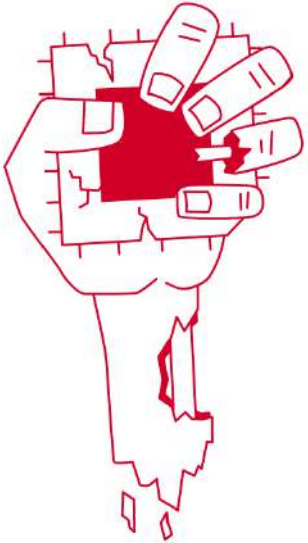
QEMU

```
Booting from ROM...
early console in extract_kernel
input_data: 0x0000000001e0a276
input_len: 0x00000000003d48f8
output: 0x0000000001000000
output_len: 0x00000000011bc258
kernel_total_size: 0x0000000000dee000
booted via startup_32()
Physical KASLR using RDTSC...
Virtual KASLR using RDTSC...

Decompressing Linux... Parsing ELF... Performing relocations... done.
Booting the kernel.
```

```
Run reader <pfn> [<cache miss threshold>] to leak hypervisor data from the L1

/ #
```


SO YOU ARE TELLING ME
YOU CAN DUMP THE MEMORY STORED IN L1?
imgflip.com


WHAT IF I TOLD YOU
YOU CAN LEAK THE ENTIRE MEMORY
imgflip.com



ZombieLoad Attack

Processors leak sensitive data and keys while accessing them.

After Meltdown, Spectre, and Foreshadow we discovered more critical vulnerabilities in modern processors. The ZombieLoad attack allows **stealing sensitive data and keys** while the computer accesses them. While programs normally only see their own data, a malicious program can exploit the fill buffers to get hold of secrets currently processed by other running programs.

The attack does not only work on **personal computers**, but can also be exploited in the **cloud.**

DOWNLOAD

CITE | TRY

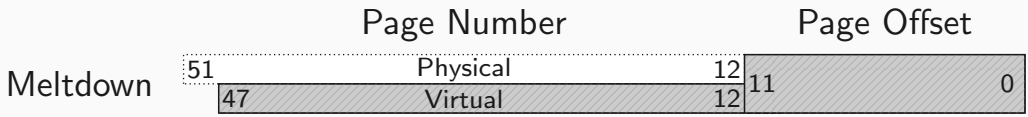
UPDATE 14.05.2019 19:00 Uhr | c't Magazin

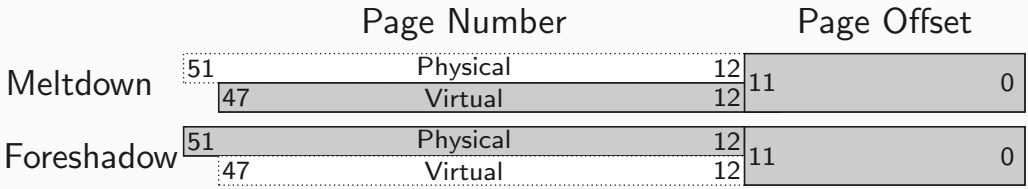
ZombieLoad: Neue Sicherheitslücken in Intel-Prozessoren

Bei vielen bisherigen Core-i- und Xeon-Prozessoren kann Malware Daten laufender Prozesse belauschen, wenn sie auf demselben Kern läuft.

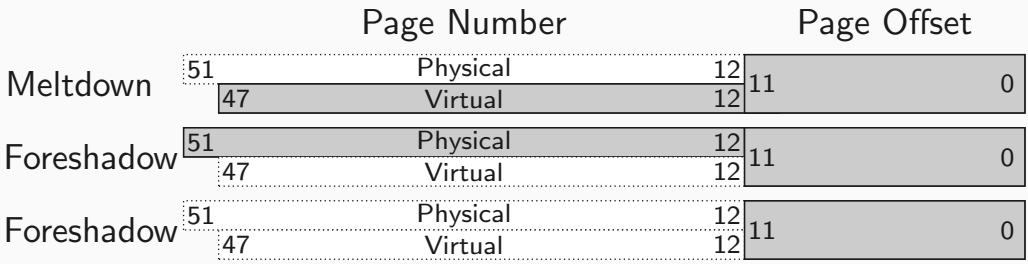
Von Christof Windeck

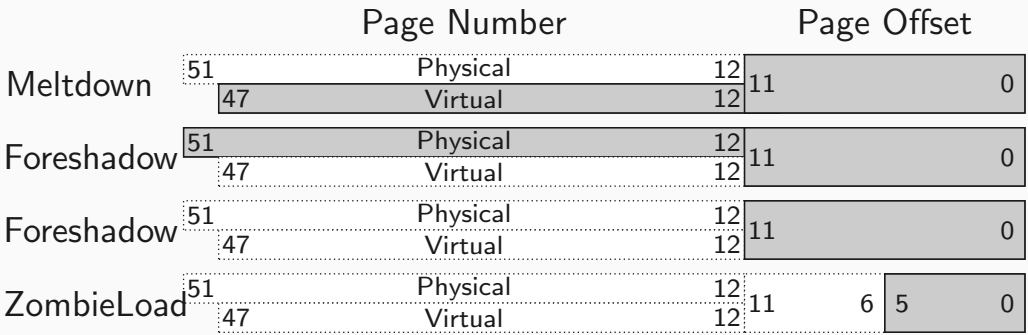

tagesschau
Daniel Gruss
Technische Universität Graz

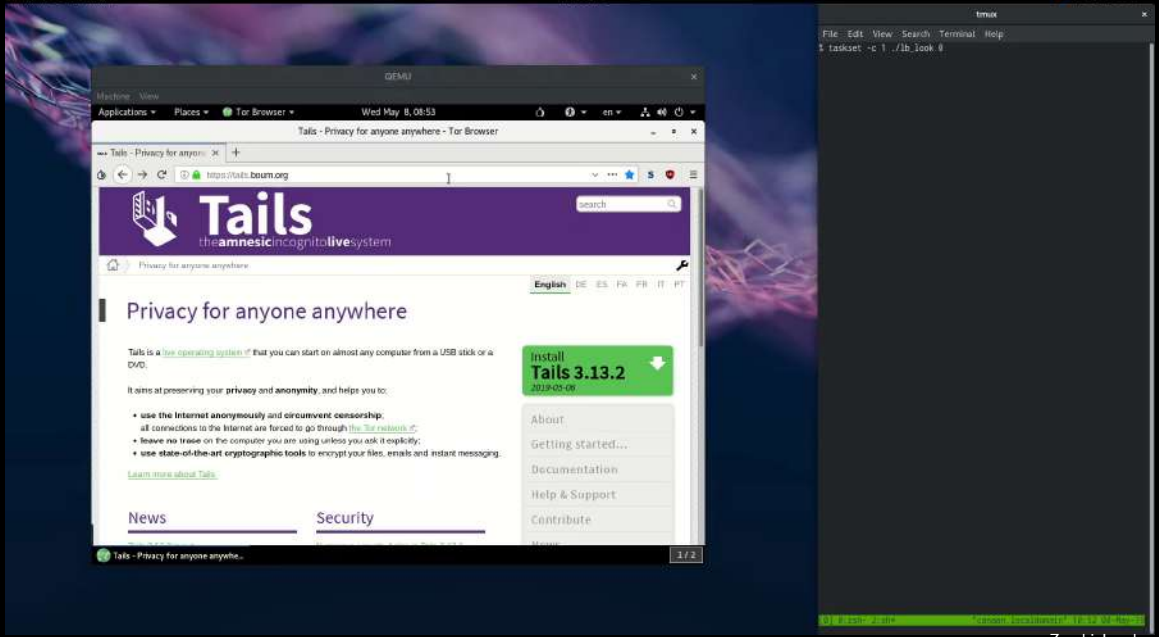

Page Number
Page Offset
Meltdown
51
Physical
12
11
0
47
Virtual
12


Page Number
Page Offset
Meltdown
51
Physical
12
47
Virtual
12
11
0
Foreshadow
51
Physical
12
47
Virtual
12
11
0

ZombieLoad Comparison




Page Number
Page Offset
Meltdown
51
Physical
12
47
Virtual
12
11
0
Foreshadow
51
Physical
12
47
Virtual
12
11
0
Foreshadow
51
Physical
12
47
Virtual
12
11
0
ZombieLoad
51
Physical
12
47
Virtual
12
11
6
5
0


Activities
Terminal
Wed 10:53
tmux
File Edit View Search Terminal Help
$ taskset -c 1 ./lb_look 0
QEMU
Machine View
Applications
Places
Tor Browser
Wed May 8, 08:53
en
Tails - Privacy for anyone anywhere - Tor Browser
Tails - Privacy for anyone
https://tails.boum.org
Tails
the amnesic incognito live system
search
Privacy for anyone anywhere
English DE ES FA FR IT PT
Privacy for anyone anywhere
Tails is a live operating system that you can start on almost any computer from a USB stick or a DVD.
It aims at preserving your privacy and anonymity, and helps you to:
use the Internet anonymously and circumvent censorship;
all connections to the Internet are forced to go through the Tor network;
leave no trace on the computer you are using unless you ask it explicitly;
use state-of-the-art cryptographic tools to encrypt your files, emails and instant messaging.
Learn more about Tails
Install
Tails 3.13.2
2019-05-08
About
Getting started...
Documentation
Help & Support
Contribute
News
Security
Tails - Privacy for anyone anywhe...
1 / 2
ZombieLoad

Zombieload: Grazer Forscher entdeckten gravierende Lücken bei Intel-Prozessoren

14. Mai 2019, 19:00

foto: lunghammer / tu graz

Die an der TU Graz forschenden oritz Lipp, Michael Schwarz und Daniel Gruss (v.l.) haben neue Sicherheitslücken in Intel-Prozessoren gefunden.

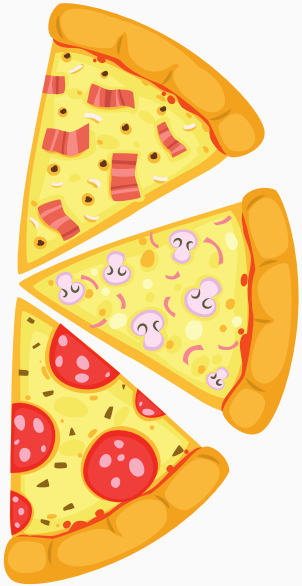
Prozessoren der Jahre 2012 bis 2018 betroffen – Neue Updates werden notwendig

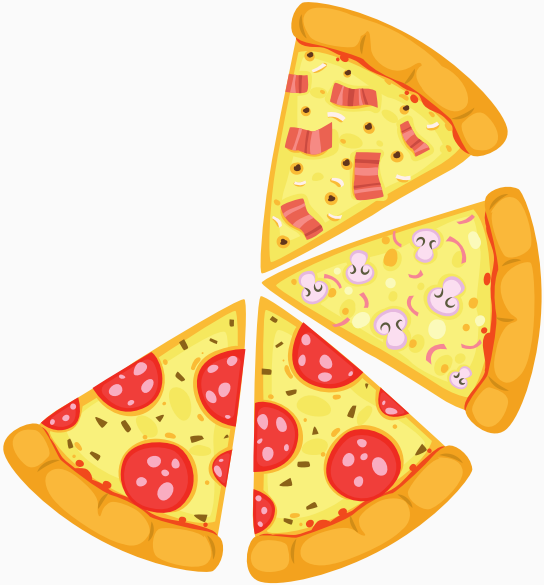
Zwei weitere Angriffsmethoden, um Daten aus Computersystemen auslesen zu können, haben IT-Experten der TU Graz gemeinsam mit einem internationalen Team entdeckt. Betroffen sind alle von Intel entwickelten Prozessoren, die zwischen 2012 und Anfang 2018 herges wurden, teilte die TU Graz am Dienstag mit. Intel wurde informiert und h bereits mit Sicherheitspatches reagiert.

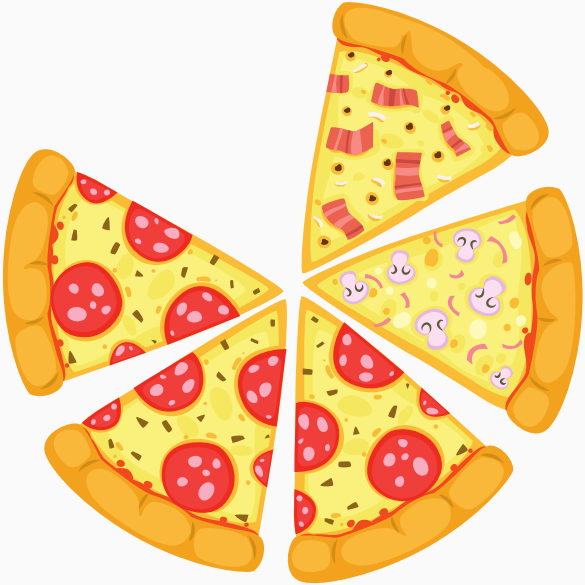
Patches gegen Meltdown und Spectre schützen nicht

"ZombieLoad" und "Store-to-Leak Forwarding" haben die


PIZZA
SPECIAL RECIPES








»A table for 6 please«

Speculative Cooking


»A table for 6 please«


PIZZA
SPECIAL RECIPES


PIZZA
SPECIAL RECIPES

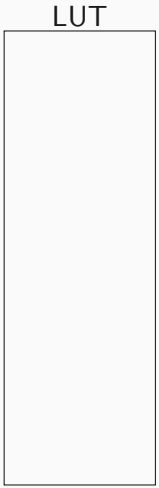

Pizza


PIZZA
SPECIAL RECIPES

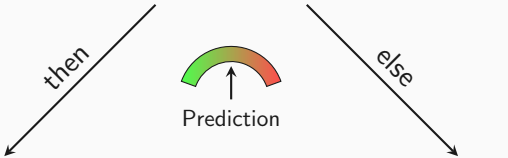

Pizza


PIZZA
SPECIAL RECIPES


Pizza

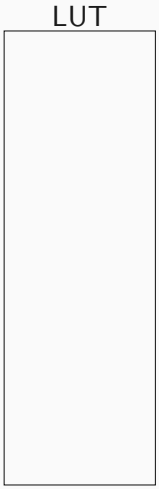



```
index = 0;

char* data = "textKEY";

if (index < 4)
```



`LUT[data[index] * 4096]`

0



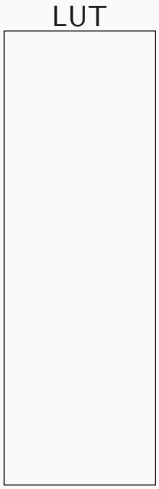

```
index = 0;
char* data = "textKEY";
if (index < 4)
```

then

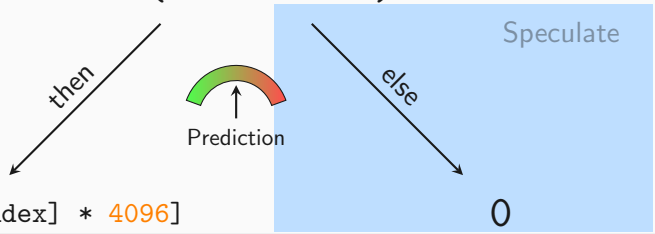
else

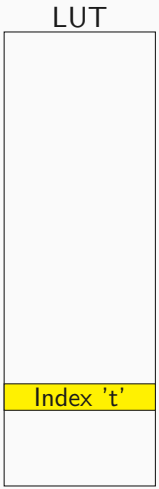

```
LUT[data[index] * 4096]
```

0




```
index = 0;
char* data = "textKEY";
if (index < 4)
```






```
index = 0;

char* data = "textKEY";

if (index < 4)
```

Execute

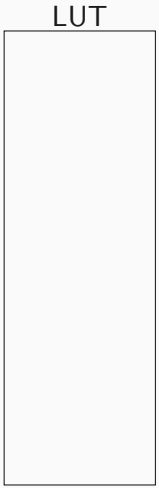
then

else

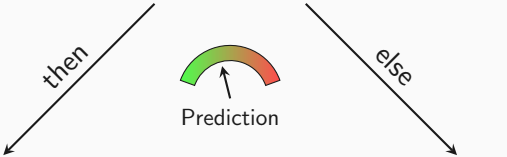
Prediction

```
LUT[data[index] * 4096]
```

0

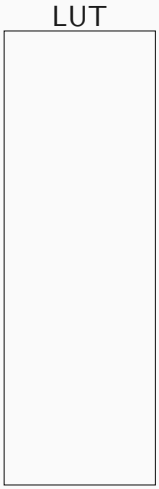



```
index = 1;

char* data = "textKEY";

if (index < 4)
```



`LUT[data[index] * 4096]`

0



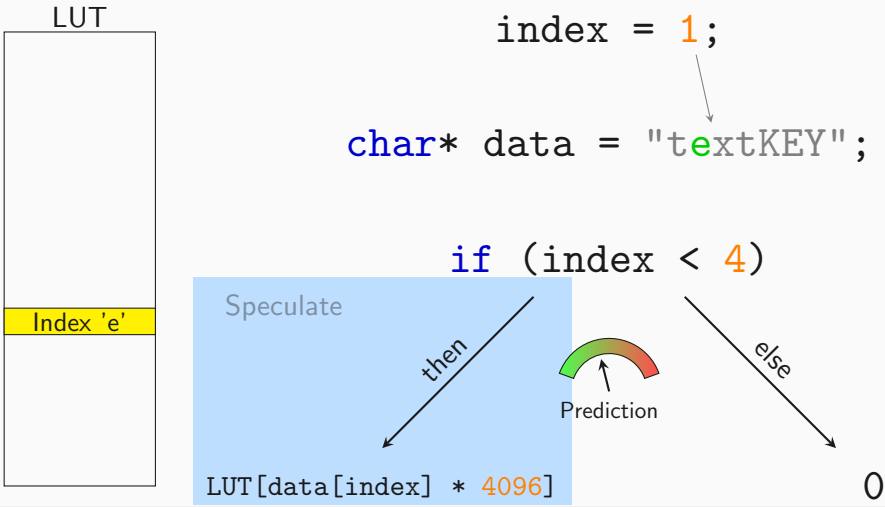

```
index = 1;
char* data = "textKEY";
if (index < 4)
```

then

else


```
LUT[data[index] * 4096]
```

0


LUT
Index 'e'
index = 1;
char* data = "textKEY";
if (index < 4)
Speculate
then
Prediction
else
LUT[data[index] * 4096]
0


```
index = 1;

char* data = "textKEY";

if (index < 4)
```

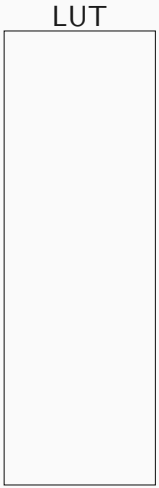
then

Prediction

else

```
LUT[data[index] * 4096]
```

0



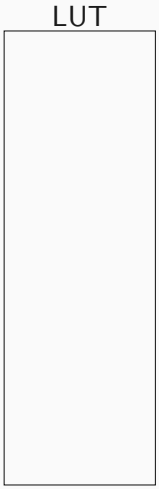

```
index = 2;

char* data = "textKEY";

if (index < 4)
```

then

else

```
LUT[data[index] * 4096]
```

0



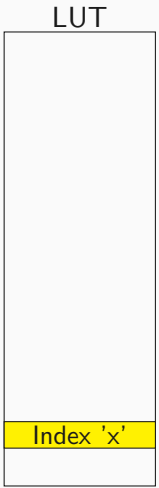

```
index = 2;
char* data = "textKEY";
if (index < 4)
```

then

else


```
LUT[data[index] * 4096]
```

0




```
index = 2;

char* data = "textKEY";

if (index < 4)
```

Speculate

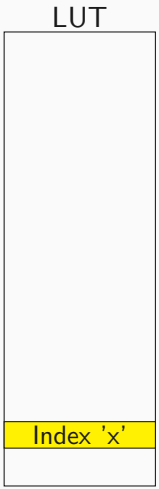
then

Prediction

else

```
LUT[data[index] * 4096]
```

0




```
index = 2;

char* data = "textKEY";

if (index < 4)
```

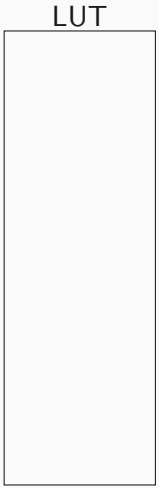
then

Prediction

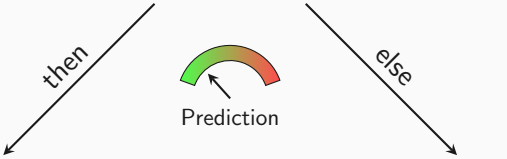
else

`LUT[data[index] * 4096]`

0

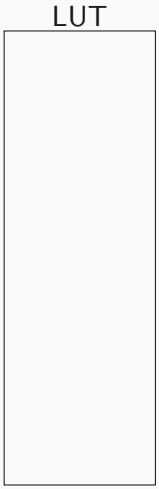



```
index = 3;

char* data = "textKEY";

if (index < 4)
```



`LUT[data[index] * 4096]`

0




```
index = 3;
```

```
char* data = "textKEY";
```

```
if (index < 4)
```

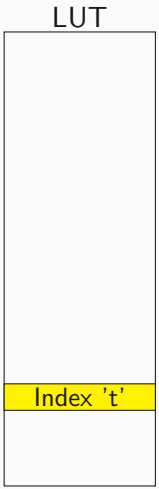
then

Prediction

else

```
LUT[data[index] * 4096]
```

0




```
index = 3;

char* data = "textKEY";

if (index < 4)
```

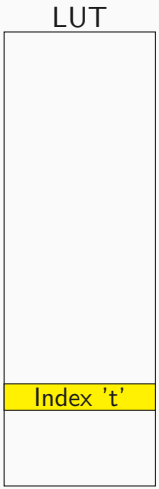
Speculate

then

else


```
LUT[data[index] * 4096]
```

0




```
index = 3;

char* data = "textKEY";

if (index < 4)
```

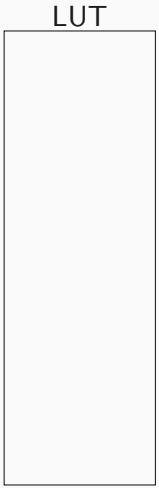
then

LUT[data[index] * 4096]

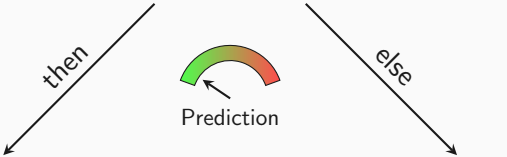
Prediction

else

0

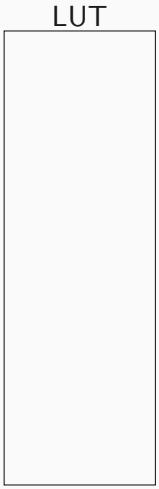



```
index = 4;

char* data = "textKEY";

if (index < 4)
```



`LUT[data[index] * 4096]`

0



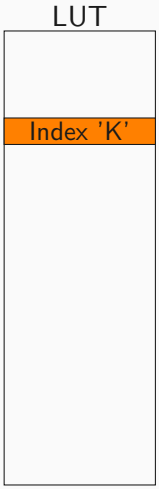

```
index = 4;

char* data = "textKEY";

if (index < 4)
```

then

else


```
LUT[data[index] * 4096]
```

0




```
index = 4;
char* data = "textKEY";
if (index < 4)
```

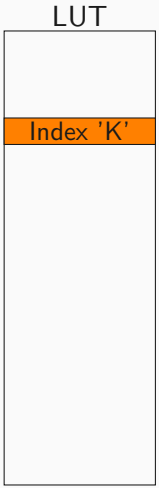
Speculate

then

else


```
LUT[data[index] * 4096]
```

0




```
index = 4;
```

```
char* data = "textKEY";
```

```
if (index < 4)
```

then

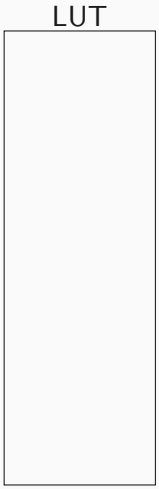
Prediction

else

Execute

```
LUT[data[index] * 4096]
```

0




```
index = 5;

char* data = "textKEY";

if (index < 4)
```

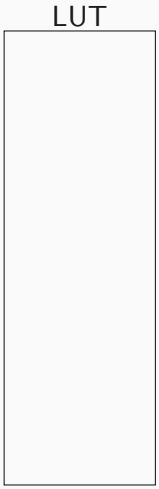
then

Prediction

else

`LUT[data[index] * 4096]`

0




```
index = 5;
```

```
char* data = "textKEY";
```

```
if (index < 4)
```

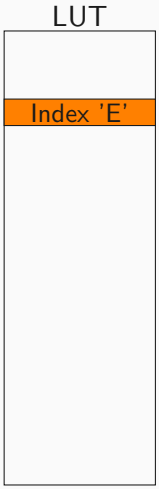
then

Prediction

else

```
LUT[data[index] * 4096]
```

```
0
```




```
index = 5;
char* data = "textKEY";
if (index < 4)
```

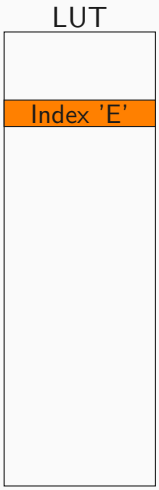
Speculate

then

else


```
LUT[data[index] * 4096]
```

0




```
index = 5;
```

```
char* data = "textKEY";
```

```
if (index < 4)
```

then

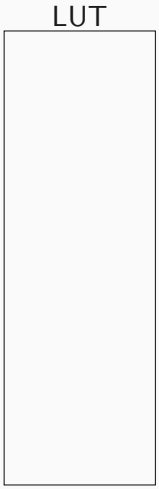
Prediction

else

Execute

```
LUT[data[index] * 4096]
```

0



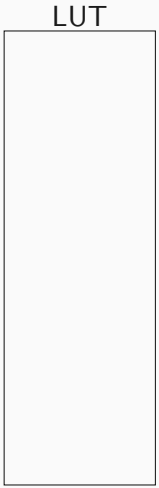

```
index = 6;

char* data = "textKEY";

if (index < 4)
```

then

else

`LUT[data[index] * 4096]`

0




```
index = 6;

char* data = "textKEY";

if (index < 4)
```

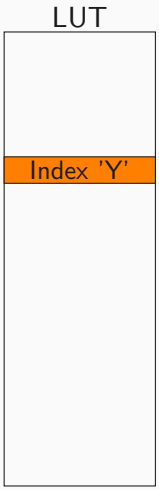
then

Prediction

else

```
LUT[data[index] * 4096]
```

0




```
index = 6;

char* data = "textKEY";

if (index < 4)
```

Speculate

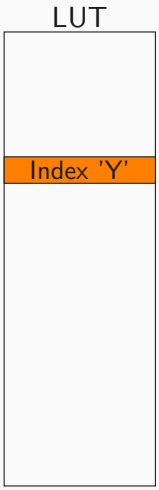
then

else

Prediction

`LUT[data[index] * 4096]`

0




```
index = 6;

char* data = "textKEY";

if (index < 4)
```

then

else

Execute

Prediction

```
LUT[data[index] * 4096]
```

0

NetSpectre:

NetSpectre:

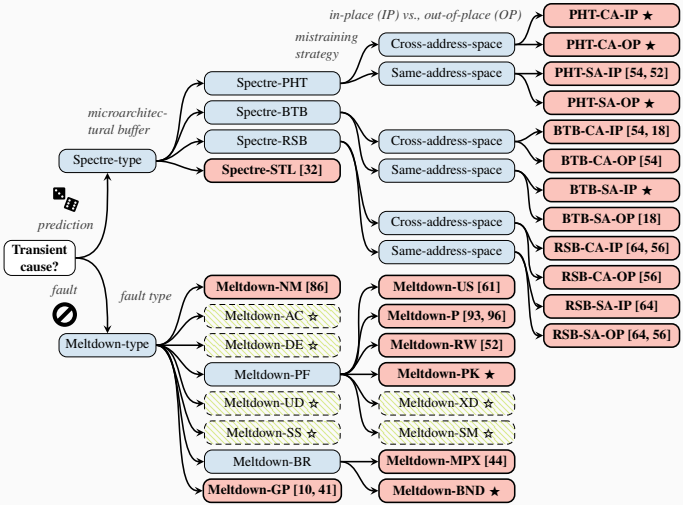
- completely remote - we just send network requests

NetSpectre:

- completely remote - we just send network requests
- leak around 15–60 bit per second

NetSpectre:

- completely remote - we just send network requests
- leak around 15–60 bit per second
- no attacker code on target system


Transient cause?
prediction
fault
Spectre-type
Meltdown-type
microarchitectural buffer
fault type
Spectre-PHT
Spectre-BTB
Spectre-RSB
Spectre-STL [32]
Meltdown-NM [86]
Meltdown-AC ☆
Meltdown-DE ☆
Meltdown-PF
Meltdown-UD ☆
Meltdown-SS ☆
Meltdown-BR
Meltdown-GP [10, 41]
mistraining strategy
in-place (IP) vs., out-of-place (OP)
Cross-address-space
Same-address-space
Cross-address-space
Same-address-space
Cross-address-space
Same-address-space
PHT-CA-IP ★
PHT-CA-OP ★
PHT-SA-IP [54, 52]
PHT-SA-OP ★
BTB-CA-IP [54, 18]
BTB-CA-OP [54]
BTB-SA-IP ★
BTB-SA-OP [18]
RSB-CA-IP [64, 56]
RSB-CA-OP [56]
RSB-SA-IP [64]
RSB-SA-OP [64, 56]
Meltdown-US [61]
Meltdown-P [93, 96]
Meltdown-RW [52]
Meltdown-PK ★
Meltdown-XD ☆
Meltdown-SM ☆
Meltdown-MPX [44]
Meltdown-BND ★

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

Ignorance is bliss?

Computer science:

Computer science:

- computer engineering

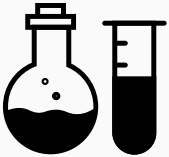
Where in Computer Science are we?

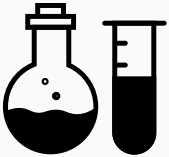
Computer science:

- computer engineering
- philosophy

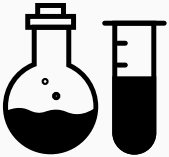
Computer science:

- computer engineering
- philosophy
- artificial science



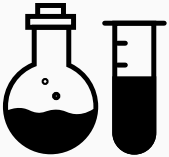


"The Sciences of the Artificial" (Herbert A. Simon, 1969)



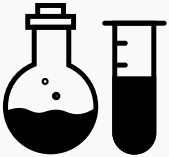
"The Sciences of the Artificial" (Herbert A. Simon, 1969)

- Natural sciences: studying something natural



"The Sciences of the Artificial" (Herbert A. Simon, 1969)

- Natural sciences: studying something natural
- Artificial science: studying something artificial (something man-made) as if it was something natural



"The Sciences of the Artificial" (Herbert A. Simon, 1969)

- Natural sciences: studying something natural
- Artificial science: studying something artificial (something man-made) as if it was something natural

→ A consequence of complexity

We have ignored microarchitectural attacks for many many years:

What do we learn from it?

We have ignored microarchitectural attacks for many many years:

- attacks on crypto

What do we learn from it?

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

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 → "not part of the threat model"

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 → "not part of the threat model"
- Rowhammer attacks

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 → "not part of the threat model"
- Rowhammer attacks → "only affects cheap sub-standard modules"

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 → "not part of the threat model"
- Rowhammer attacks → "only affects cheap sub-standard modules"

→ for years we solely optimized for performance

- The complexity of the systems we built is too large to fully understand them
- We need to study man-made systems like nature to find flaws
- We need good and adjustable trade-offs between security and performance, efficiency, and complexity
- Learn from nature, Learn to cope with diseases

Meltdown, Spectre, ZombieLoad

Daniel Gruss

May 16, 2019

Graz University of Technology