Flush+Reload, Meltdown, Spectre, Rowhammer

Zu Risiken und Nebenwirkungen fragen Sie Ihr Prozessorhandbuch oder Ihren Seitenkanal-Forscher

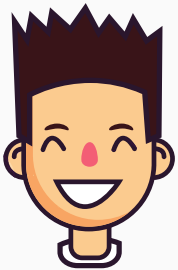
---

Michael Schwarz (@misc0110)

25.05.2018

www.iaik.tugraz.at

Wer bin ich?



Michael Schwarz

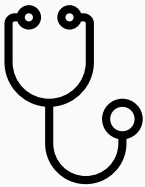
PhD student @ Technische Universität Graz

Fokus auf Seitenkanal Angriffe

@misc0110

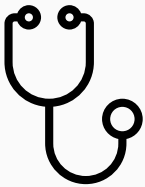
michael.schwarz91@gmail.com

Seitenkanäle



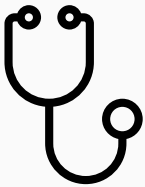
- Seitenkanäle gibt es auch in Software

Seitenkanäle in Software



- Seitenkanäle gibt es auch in Software
- Können auch für Attacken ausgenutzt werden

Seitenkanäle in Software



- Seitenkanäle gibt es auch in Software
- Können auch für Attacken ausgenutzt werden
- Statt Geräusche misst man meistens Zeitunterschiede

Beispiel: PIN Vergleich

- Trivialer Ansatz:
  - Ziffer für Ziffer vergleichen
  - Sobald sich eine Ziffer unterscheidet, wird der Vergleich abgebrochen

```
1 function check_pin($input) {
2   $correct = "1234"
3   for($i = 0; $i < 4; $i++) {
4     if($correct[$i] != $input[$i]) {
5       // Unterschied, Abbruch
6       return false;
7     }
8   }
9   // PIN korrekt
10  return true;
11 }
```

Beispiel: PIN Vergleich

- Messung der Ausführungszeit für verschiedene PINs

PIN	Zeit

Beispiel: PIN Vergleich

- Messung der Ausführungszeit für verschiedene PINs

PIN	Zeit
0000	

Beispiel: PIN Vergleich

- Messung der Ausführungszeit für verschiedene PINs

PIN	Zeit
0000	
1000	

Beispiel: PIN Vergleich

- Messung der Ausführungszeit für verschiedene PINs

PIN	Zeit
0000	
1000	
2000	

Beispiel: PIN Vergleich

- Messung der Ausführungszeit für verschiedene PINs

PIN	Zeit
0000	
1000	
2000	
3000	

Beispiel: PIN Vergleich

- Messung der Ausführungszeit für verschiedene PINs

PIN	Zeit
0000	
1000	
2000	
3000	
...	...

Beispiel: PIN Vergleich

- Messung der Ausführungszeit für verschiedene PINs

PIN	Zeit
0000	
1000	
2000	
3000	
...	...

- Bei korrekter Ziffer wird die nächste Ziffer überprüft → längere Ausführungszeit

Beispiel: PIN Vergleich

- Messung der Ausführungszeit für verschiedene PINs

PIN	Zeit
0000	
1000	
2000	
3000	
...	...

- Bei korrekter Ziffer wird die nächste Ziffer überprüft → längere Ausführungszeit
- Maximal 10 Versuche um die erste Stelle herauszufinden

Beispiel: PIN Vergleich

- Messung der Ausführungszeit für verschiedene PINs

PIN	Zeit

Beispiel: PIN Vergleich

- Messung der Ausführungszeit für verschiedene PINs

PIN	Zeit
1000	

Beispiel: PIN Vergleich

- Messung der Ausführungszeit für verschiedene PINs

PIN	Zeit
1000	
1100	

Beispiel: PIN Vergleich

- Messung der Ausführungszeit für verschiedene PINs

PIN	Zeit
1000	
1100	
1200	

Beispiel: PIN Vergleich

- Messung der Ausführungszeit für verschiedene PINs

PIN	Zeit
1000	
1100	
1200	
1300	

Beispiel: PIN Vergleich

- Messung der Ausführungszeit für verschiedene PINs

PIN	Zeit
1000	
1100	
1200	
1300	
...	...

Beispiel: PIN Vergleich

- Messung der Ausführungszeit für verschiedene PINs

PIN	Zeit
1000	
1100	
1200	
1300	
...	...

- Für jede Ziffer wiederholen

Beispiel: PIN Vergleich

- Messung der Ausführungszeit für verschiedene PINs

PIN	Zeit
1000	
1100	
1200	
1300	
...	...

- Für jede Ziffer wiederholen
- Längste Ausführungszeit verrät die Ziffer

Beispiel: PIN Vergleich

- Für jede Ziffer maximal 10 Messungen

Beispiel: PIN Vergleich

- Für jede Ziffer maximal 10 Messungen
- Bei 4 Stellen: 40 Versuche um PIN zu erraten

Beispiel: PIN Vergleich

- Für jede Ziffer maximal 10 Messungen
- Bei 4 Stellen: 40 Versuche um PIN zu erraten
- Vergleich zu probieren: 10 000 Möglichkeiten

Beispiel: PIN Vergleich

- Für jede Ziffer maximal 10 Messungen
- Bei 4 Stellen: 40 Versuche um PIN zu erraten
- Vergleich zu probieren: 10 000 Möglichkeiten
- Seitenkanal reduziert Anzahl Versuche um Faktor 250

Beispiel: PIN Vergleich

- Viele Funktionen können mit konstanter Laufzeit implementiert werden

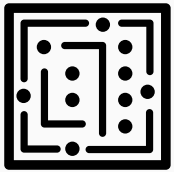
```
1 function check_pin($input) {
2   $correct = "1234";
3   // Annahme: PIN korrekt
4   $ok = true;
5   for($i = 0; $i < 4; $i++) {
6     if($correct[$i] != $input[$i]) {
7       // Unterschied, merken
8       $ok = false;
9     }
10  }
11  return $ok;
12 }
```

- Viele Funktionen können mit konstanter Laufzeit implementiert werden

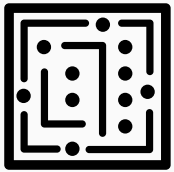
```
1 function check_pin($input) {
2   $correct = "1234";
3   // Annahme: PIN korrekt
4   $ok = true;
5   for($i = 0; $i < 4; $i++) {
6     if($correct[$i] != $input[$i]) {
7       // Unterschied, merken
8       $ok = false;
9     }
10  }
11  return $ok;
12 }
```

- Doch manchmal ist der Seitenkanal in der Hardware

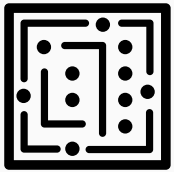
Mikroarchitekturelle Seitenkanäle



- Mikroarchitektur beschreibt die interne Art wie CPUs arbeiten



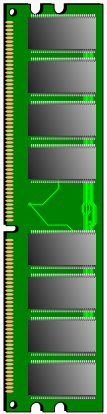
- Mikroarchitektur beschreibt die interne Art wie CPUs arbeiten
- Nicht sichtbar für Benutzer oder Programmierer



- Mikroarchitektur beschreibt die interne Art wie CPUs arbeiten
- Nicht sichtbar für Benutzer oder Programmierer
- Ist großteils nicht dokumentiert und kann nicht direkt beobachtet werden

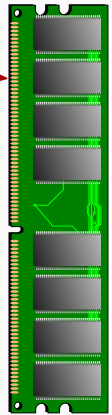
Beispiel: Daten im Speicher (Architektur)

```
$i = 42;
echo $i;
echo $i;
```



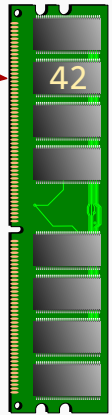
Beispiel: Daten im Speicher (Architektur)

```
$i = 42;
echo $i;
echo $i;
```



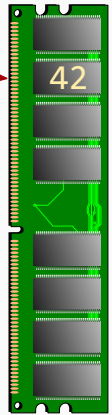
Beispiel: Daten im Speicher (Architektur)

```
$i = 42;
echo $i;
echo $i;
```



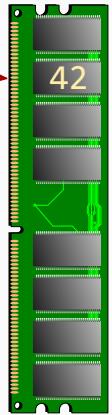
Beispiel: Daten im Speicher (Architektur)

```
$i = 42;
echo $i;
echo $i;
```



Beispiel: Daten im Speicher (Architektur)

```
$i = 42;
echo $i;
echo $i;
```



1337 4242

FOOD CACHE

Revolutionary concept!

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

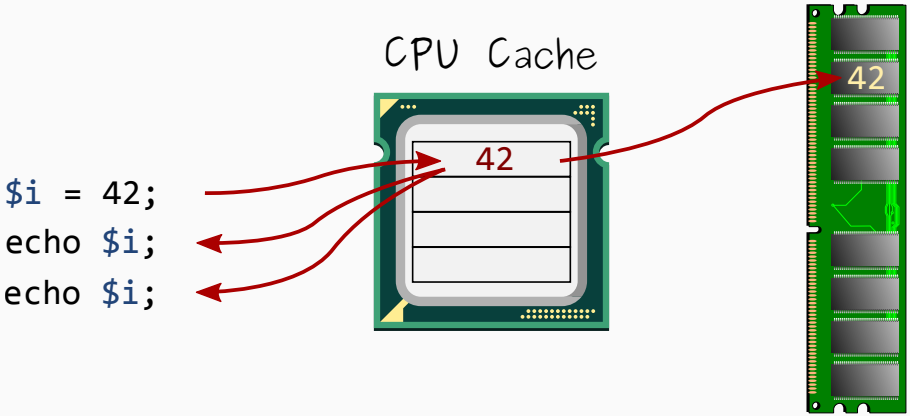
Can store **ALL** kinds of food.

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

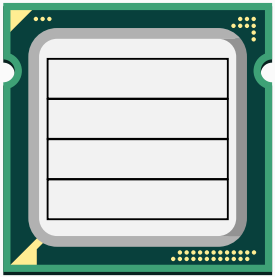
$1,299

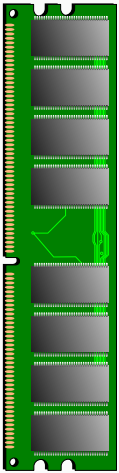
ORDER VIA PHONE: +555 12345

Beispiel: Daten im Speicher (Mikroarchitektur vereinfacht)

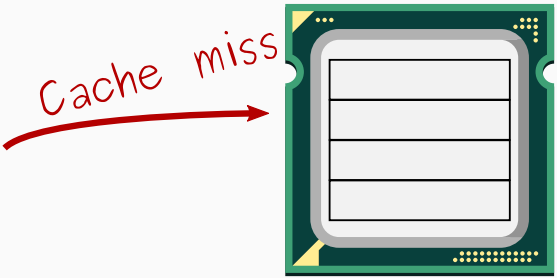


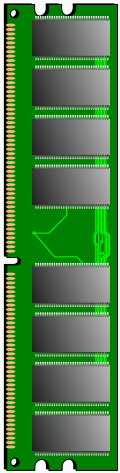
```
echo $i;
echo $i;
```



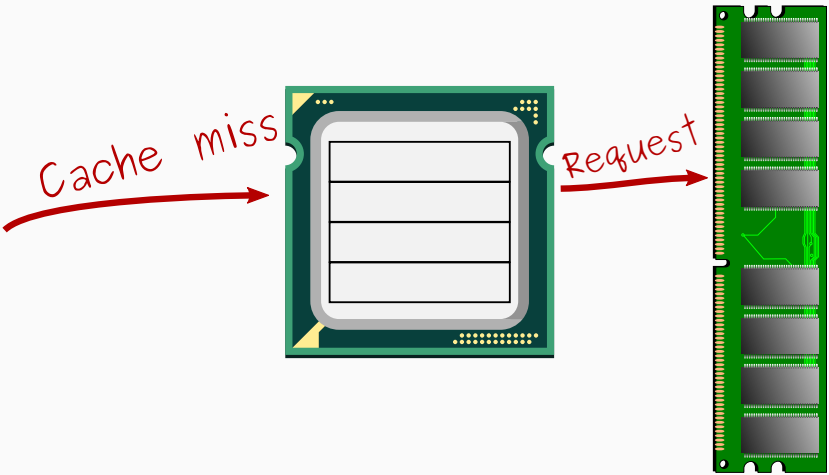


```
echo $i;
echo $i;
```

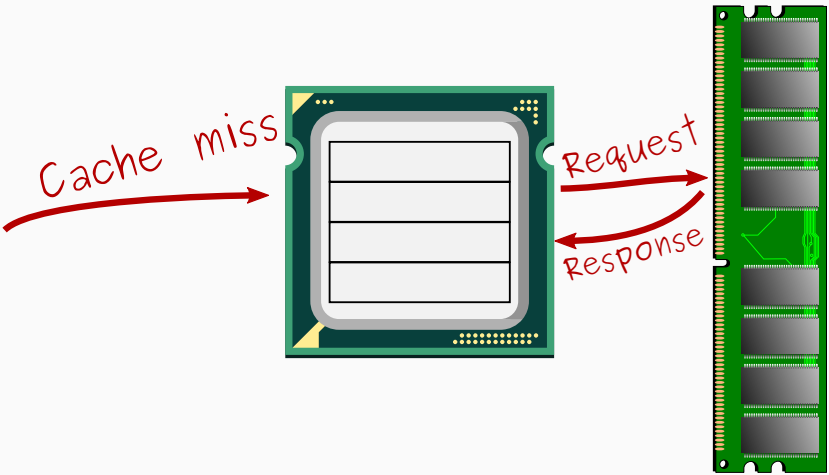




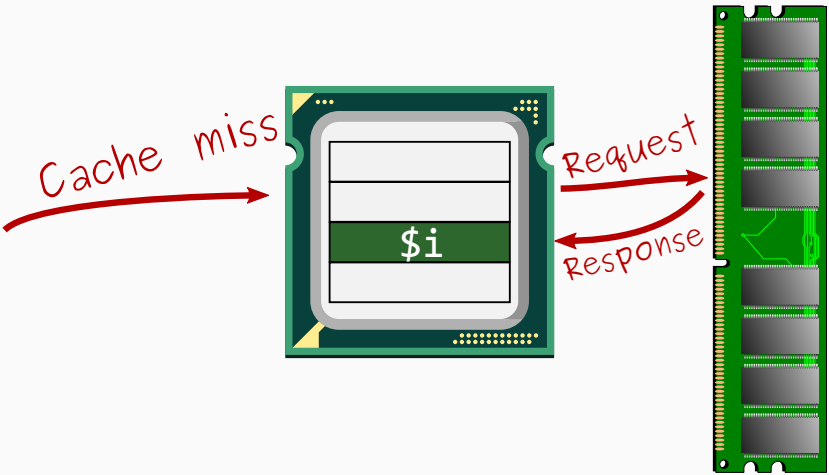
```
echo $i;
echo $i;
```

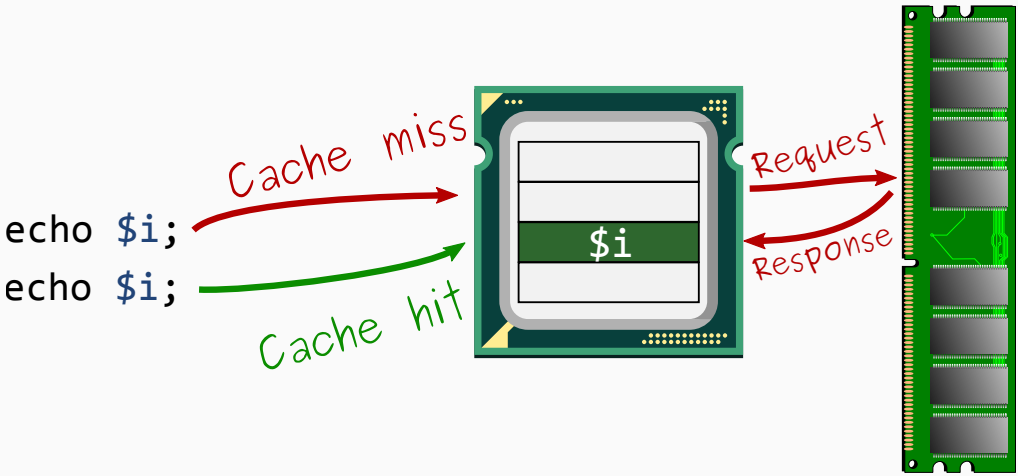


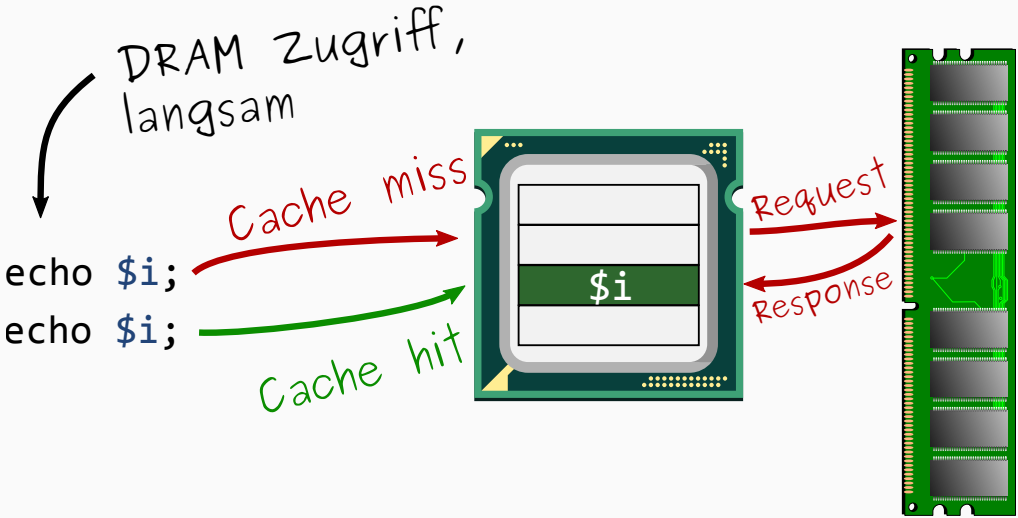
```
echo $i;
echo $i;
```

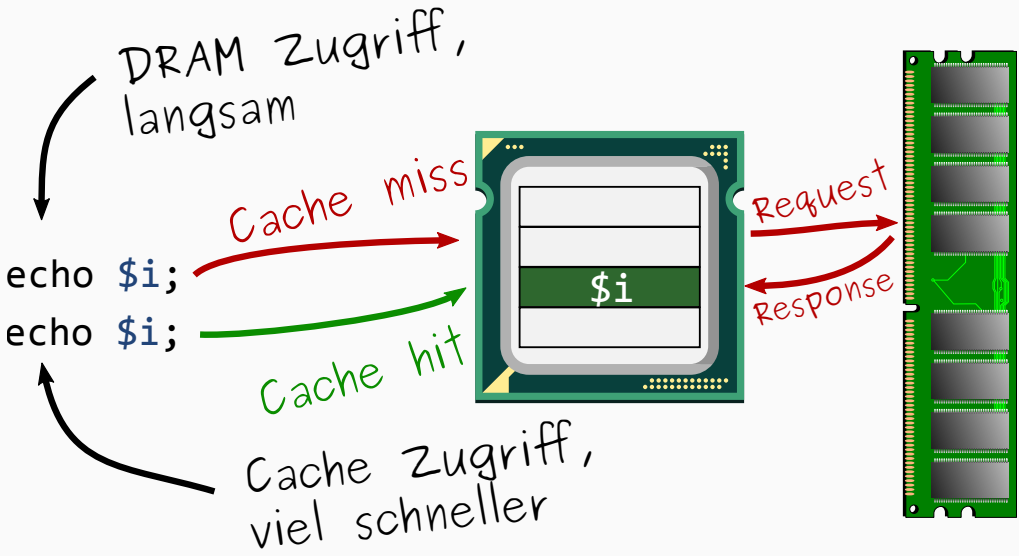


```
echo $i;
echo $i;
```

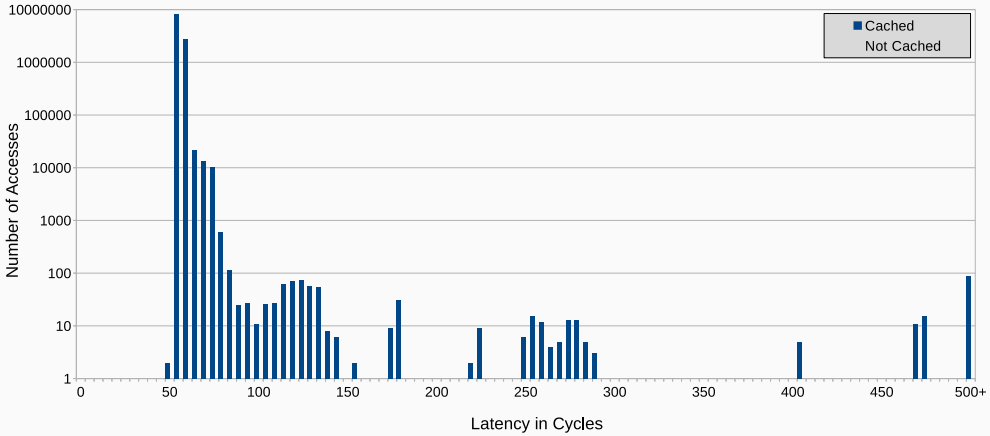



echo $i;
echo $i;
Cache miss
Cache hit
$i
Request
Response

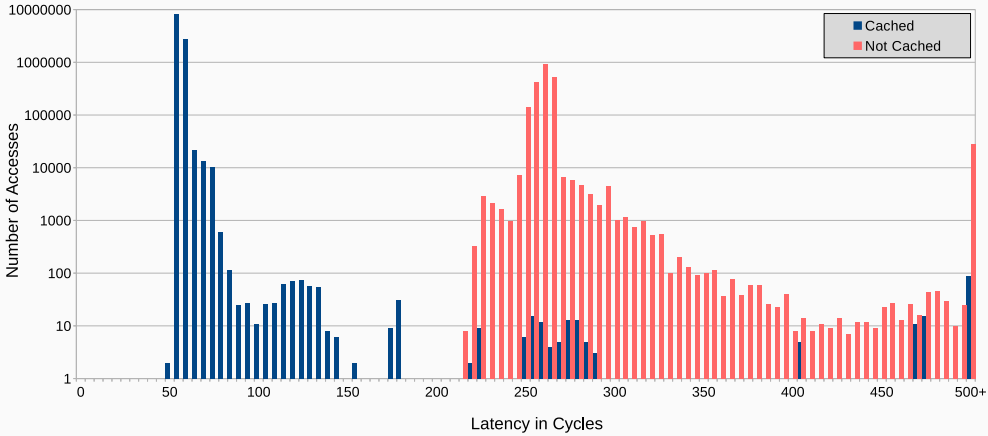

DRAM Zugriff, langsam
Cache miss
Cache hit
echo $i;
echo $i;
$i
Request
Response


DRAM Zugriff,
langsam
Cache miss
echo $i;
echo $i;
Cache hit
Cache Zugriff,
viel schneller
$i
Request
Response

CPU Cache


Cached
Not Cached
Number of Accesses
10000000
1000000
100000
10000
1000
100
10
1
0
50
100
150
200
250
300
350
400
450
500+
Latency in Cycles

CPU Cache


Cached
Not Cached
Number of Accesses
10000000
1000000
100000
10000
1000
100
10
1
0
50
100
150
200
250
300
350
400
450
500+
Latency in Cycles

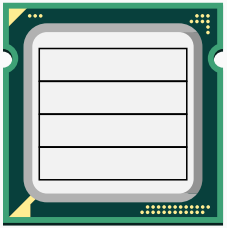
Seitenkanalangriffe

Shared Memory

ATTACKER

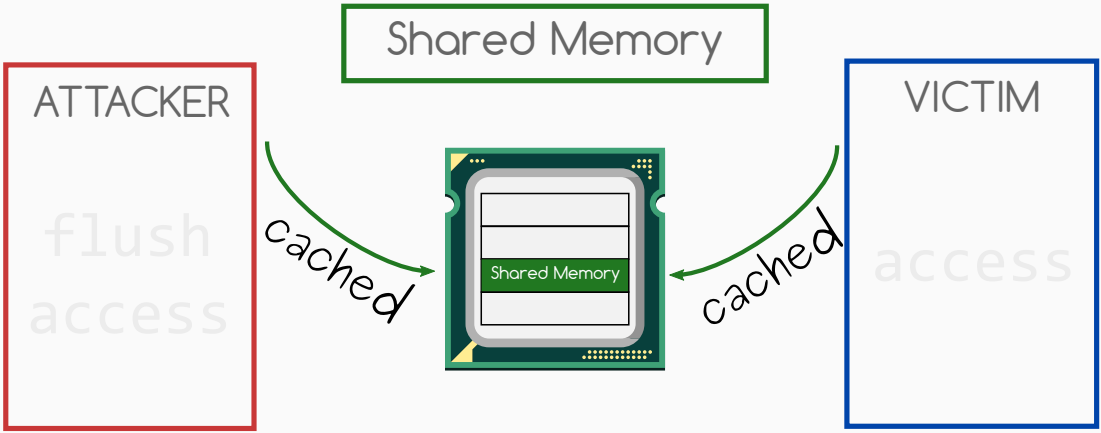
flush

access



VICTIM

access

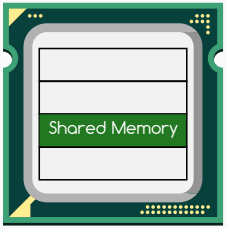

Shared Memory
ATTACKER
flush
access
cached
Shared Memory
cached
VICTIM
access

Shared Memory

ATTACKER

flush

access



VICTIM

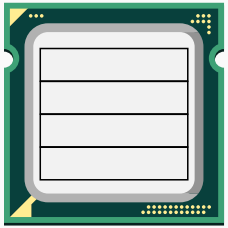
access

Shared Memory

ATTACKER

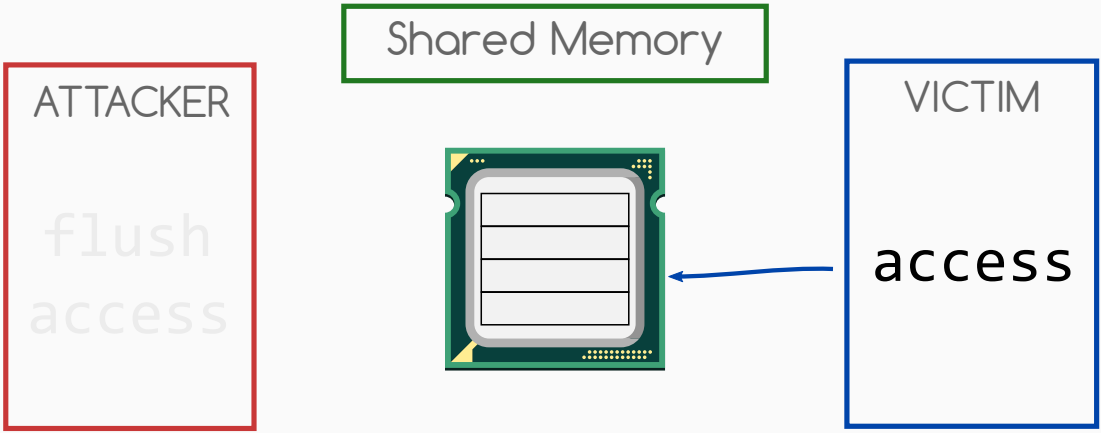
flush

access



VICTIM

access

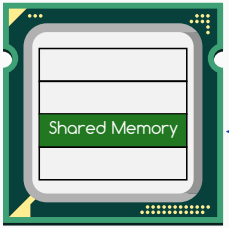

Shared Memory
ATTACKER
flush
access
VICTIM
access

Shared Memory

ATTACKER

flush

access



VICTIM

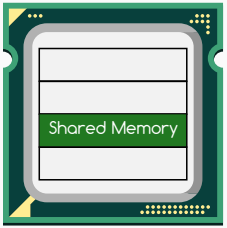
access

Shared Memory

ATTACKER

flush

access



VICTIM

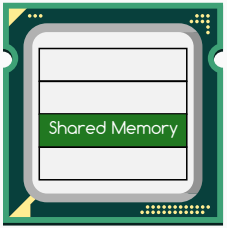
access

Shared Memory

ATTACKER

flush

access

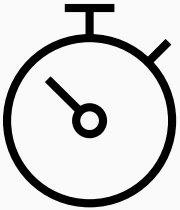


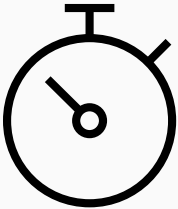
VICTIM

access

schnell, wenn auf die Daten zugegriffen wurde,
sonst langsam

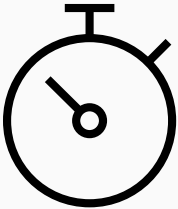
```
1 $array[84] = 0;
```





```
1 $array[84] = 0;
```

- Flush+Reload über alle Indices



```
1 $array[84] = 0;
```

- Flush+Reload über alle Indices

- Zugegriffener Index lädt schneller

- Flush+Reload nutzt den Cache als Seitenkanal

- Flush+Reload nutzt den Cache als Seitenkanal
- Ermöglicht Speicherzugriffe zu beobachten

- Flush+Reload nutzt den Cache als Seitenkanal
- Ermöglicht Speicherzugriffe zu beobachten
- Flush+Reload kann jedoch keine Werte lesen

Meltdown

MELTDOWN

Ein Fehler in Prozessoren, so schwerwiegend, dass...

Ein Fehler in Prozessoren, so schwerwiegend, dass...

- er weltweit in den Nachrichten war


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


FIREEYE CYBER
THREAT MAP
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

Ein Fehler in Prozessoren, so schwerwiegend, dass...

- er weltweit in den Nachrichten war
- es Wikipedia Artikel in mehreren Sprachen gibt


Not logged in Talk Contributions Create account Log in
Article Talk
Read Edit View history
Search Wikipedia
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 (security vulnerability)
From Wikipedia, the free encyclopedia
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"
MELTDOWN
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]

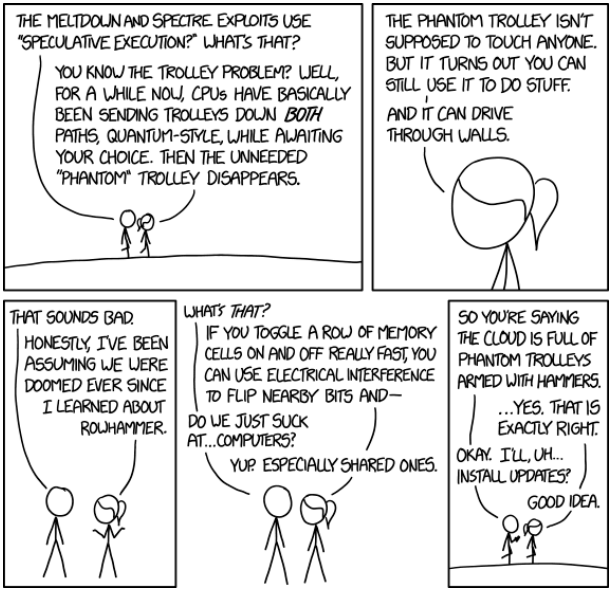
A logo created for the vulnerability, featuring a ghost with a branch

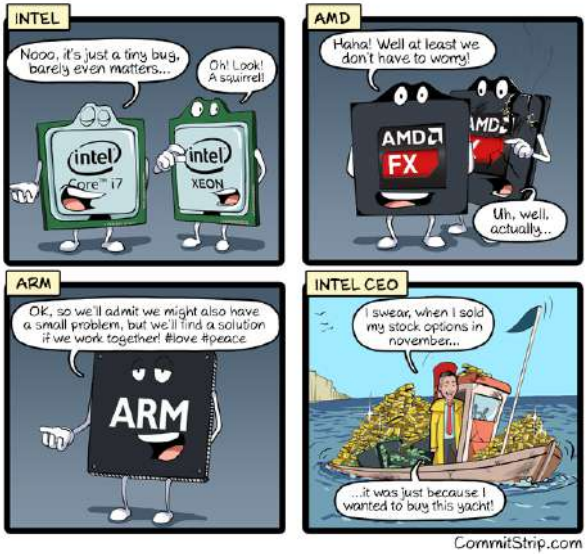
Two Common Vulnerabilities and Exposures IDs related to Spectre, CVE-2017-5753 (bounds check bypass) and CVE-2017-5715 (branch target injection), have been issued.[7] JIT engines used for JavaScript were found vulnerable. A website can read data stored in the browser for another website, or the browser's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

Ein Fehler in Prozessoren, so schwerwiegend, dass...

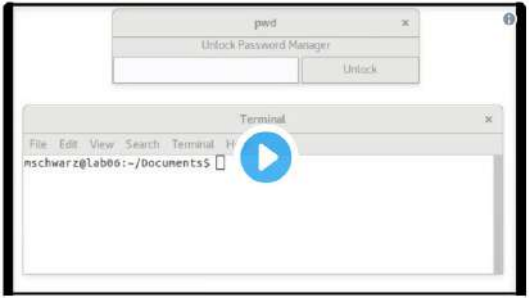
- er weltweit in den Nachrichten war
- es Wikipedia Artikel in mehreren Sprachen gibt
- es sogar Comics darüber gibt


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

Ein Fehler in Prozessoren, so schwerwiegend, dass...

- er weltweit in den Nachrichten war
- es Wikipedia Artikel in mehreren Sprachen gibt
- es sogar Comics darüber gibt
- sogar Snowden auf Twitter darüber schreibt



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

Meltdown

- Ein Fehler in Intel Prozessoren (und manchen ARM Prozessoren)

Meltdown

- Ein Fehler in Intel Prozessoren (und manchen ARM Prozessoren)
- Meltdown kann beliebigen Speicher auslesen

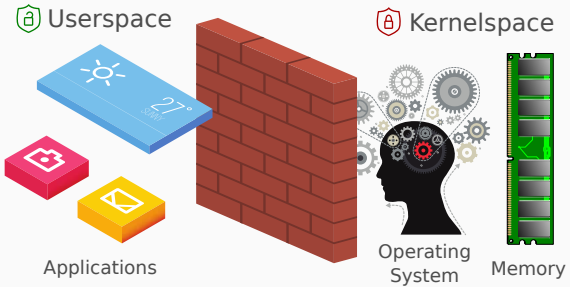
Meltdown

- Ein Fehler in Intel Prozessoren (und manchen ARM Prozessoren)
- Meltdown kann beliebigen Speicher auslesen
- Sowohl Speicher des Betriebssystems, als auch anderer Anwendungen

Wie greift man eigentlich auf fremden Speicher zu?

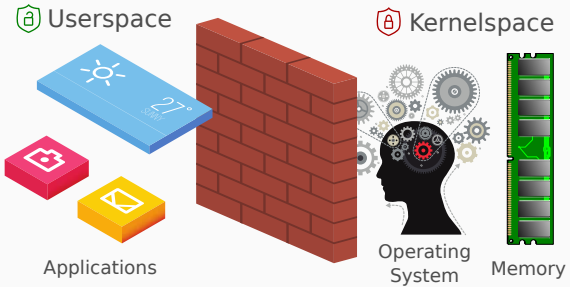
Das Betriebssystem

- Das Betriebssystem ist isoliert von allen anderen Anwendungen



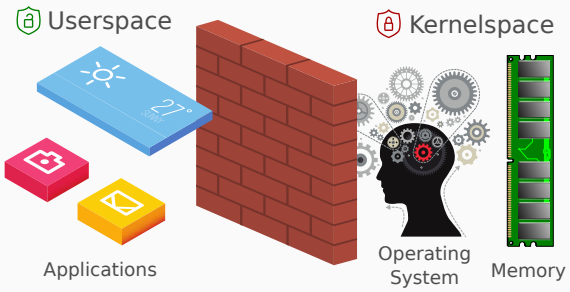
Das Betriebssystem

- Das Betriebssystem ist isoliert von allen anderen Anwendungen
- Diese Isolation wird durch Software und Hardware umgesetzt



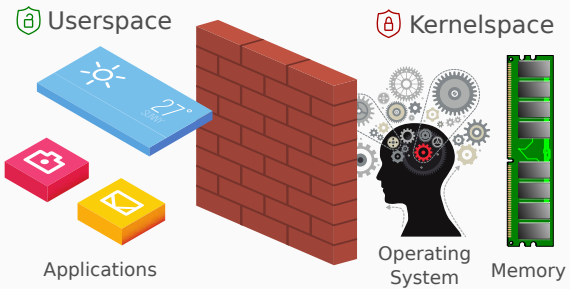
Das Betriebssystem

- Das Betriebssystem ist isoliert von allen anderen Anwendungen
- Diese Isolation wird durch Software und Hardware umgesetzt
- Anwendungen können nicht direkt auf das Betriebssystem zugreifen

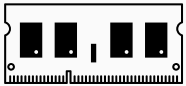


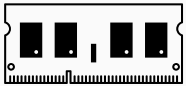
Das Betriebssystem

- Das Betriebssystem ist isoliert von allen anderen Anwendungen
- Diese Isolation wird durch Software und Hardware umgesetzt
- Anwendungen können nicht direkt auf das Betriebssystem zugreifen
- Das Betriebssystem kann allerdings auf jeden Teil des Speichers zugreifen

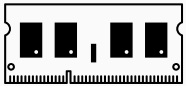


- Für das Betriebssystem ist der Speicher ein großes Array

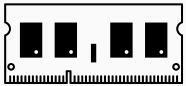




- Für das Betriebssystem ist der Speicher ein großes Array
- Die CPU prüft, ob der Zugriff auf dieses Array wirklich vom Betriebssystem kommt

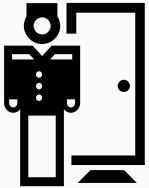


- Für das Betriebssystem ist der Speicher ein großes Array
- Die CPU prüft, ob der Zugriff auf dieses Array wirklich vom Betriebssystem kommt
- Wenn der Zugriff von einer Anwendung kommt, wird diese beendet

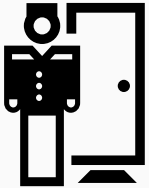


- Für das Betriebssystem ist der Speicher ein großes Array
- Die CPU prüft, ob der Zugriff auf dieses Array wirklich vom Betriebssystem kommt
- Wenn der Zugriff von einer Anwendung kommt, wird diese beendet
- In Programmiersprachen wie C/C++ kann man dies direkt probieren

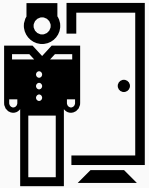
Speicherzugriff



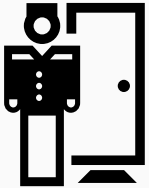
- Der Prozessor prüft, ob der Zugriff erlaubt ist



- Der Prozessor prüft, ob der Zugriff erlaubt ist
- Ja $\rightarrow$ Daten werden geladen



- Der Prozessor prüft, ob der Zugriff erlaubt ist
- Ja $\rightarrow$ Daten werden geladen
- Nein $\rightarrow$ Programm wird beendet



- Der Prozessor prüft, ob der Zugriff erlaubt ist
- Ja $\rightarrow$ Daten werden geladen
- Nein $\rightarrow$ Programm wird beendet
- Wirklich?

```
1 $x = 1 % 0; // Division by Zero
2 // < Programm wird beendet >
3
4 $array[84] = 0;
```

```php
$x = 1 % 0; // Division by Zero
// < Programm wird beendet >

$array[84] = 0;
```

- Flush+Reload über alle Indices

```
1 $x = 1 % 0; // Division by Zero
2 // < Programm wird beendet >
3
4 $array[84] = 0;
```

- Flush+Reload über alle Indices

- Zugegriffener Index lädt schneller...

```
1 $x = 1 % 0; // Division by Zero
2 // < Programm wird beendet >
3
4 $array[84] = 0;
```

- Flush+Reload über alle Indices

- Zugegriffener Index lädt schneller...
- ...obwohl die Zeile nicht ausgeführt wurde

Out-of-order Ausführung

7. Serve with cooked and peeled potatoes

Eine Stunde warten

Eine Stunde warten

LATENZ

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

Abhängigkeit

1. Wash and cut vegetables

2. Pick the basil leaves and set aside

3. Heat 2 tablespoons of oil in a pan

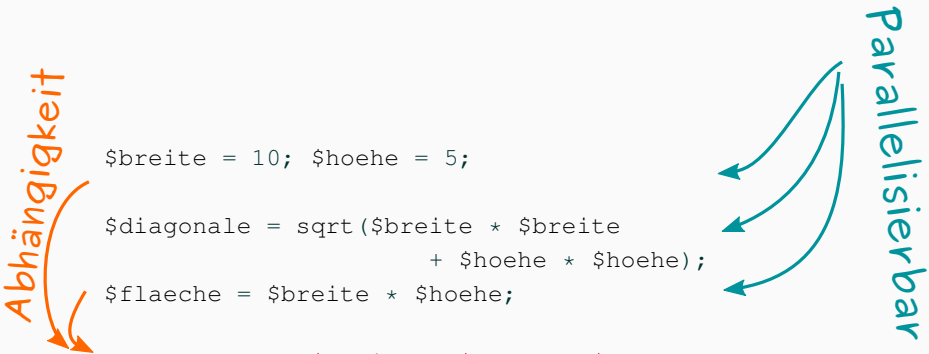
4. Fry vegetables until golden and softened

Parallelisierbar

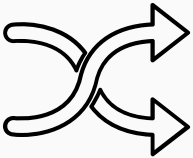
Out-of-order Ausführung

```
$breite = 10; $hoehe = 5;

$diagonale = sqrt($breite * $breite
                      + $hoehe * $hoehe);
$flaeche = $breite * $hoehe;

echo "Flaeche $breite x $hoehe = $flaeche";
```

Out-of-order Ausführung

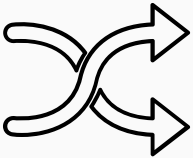



```
$breite = 10; $hoehe = 5;

$diagonale = sqrt($breite * $breite
                        + $hoehe * $hoehe);
$flaeche = $breite * $hoehe;

echo "Flaeche $breite x $hoehe = $flaeche";
```

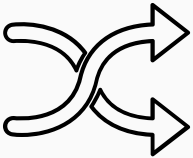


- Code muss nicht in der Reihenfolge ausgeführt werden wie im Programm

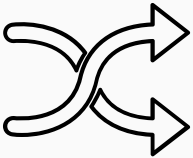
Out-of-order Ausführung



- Code muss nicht in der Reihenfolge ausgeführt werden wie im Programm
- Ausführung muss jedoch immer korrekt sein

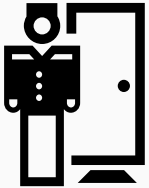


- Code muss nicht in der Reihenfolge ausgeführt werden wie im Programm
- Ausführung muss jedoch immer korrekt sein
- Nicht nur der Fall auf Code-Ebene



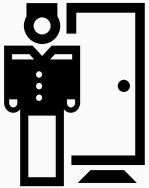
- Code muss nicht in der Reihenfolge ausgeführt werden wie im Programm
- Ausführung muss jedoch immer korrekt sein
- Nicht nur der Fall auf Code-Ebene
- Auch einzelne Programmteile können out-of-order ausgeführt werden

Recap: Speicherzugriff



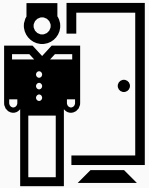
- Der Prozessor prüft, ob der Zugriff erlaubt ist

Recap: Speicherzugriff



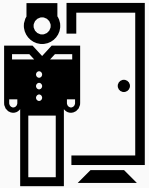
- Der Prozessor prüft, ob der Zugriff erlaubt ist
- Ja $\rightarrow$ Daten werden geladen

Recap: Speicherzugriff



- Der Prozessor prüft, ob der Zugriff erlaubt ist
- Ja $\rightarrow$ Daten werden geladen
- Nein $\rightarrow$ Programm wird beendet

Recap: Speicherzugriff



- Der Prozessor prüft, ob der Zugriff erlaubt ist
- Ja → Daten werden geladen
- Nein → Programm wird beendet
- Kann auch hier die Reihenfolge umgedreht werden?

Out-of-order Ausführung

- Zuerst laden, danach Berechtigung prüfen?

Out-of-order Ausführung

- Zuerst laden, danach Berechtigung prüfen?
- Wer hält das für eine gute Idee?

Out-of-order Ausführung

- Zuerst laden, danach Berechtigung prüfen?
- Wer hält das für eine gute Idee?
- Zumindest Intel hielt es für eine gute Idee

Out-of-order Ausführung

- Zuerst laden, danach Berechtigung prüfen?
- Wer hält das für eine gute Idee?
- Zumindest Intel hielt es für eine gute Idee
- Begründung: Programm wird danach so oder so beendet


intel inside
CRIME SCENE DO NOT CROSS
CRIME SCENE DO NOT CROSS

Ein Wettlauf

- Annahme: Berechtigung wird nach dem Zugriff geprüft

Ein Wettlauf

- Annahme: Berechtigung wird nach dem Zugriff geprüft
- Resultat: Ein kleines Zeitfenster, in dem wir mit den Daten arbeiten können

Ein Wettlauf

- Annahme: Berechtigung wird nach dem Zugriff geprüft
- Resultat: Ein kleines Zeitfenster, in dem wir mit den Daten arbeiten können
- Daten müssen an einen “sicheren” Ort

Ein Wettlauf

- Annahme: Berechtigung wird nach dem Zugriff geprüft
- Resultat: Ein kleines Zeitfenster, in dem wir mit den Daten arbeiten können
- Daten müssen an einen “sicheren” Ort
- Alles Sichtbare (Variablen, Dateien, ...) wird aufgeräumt

- Kann die CPU alle Spuren vom “Tatort” beseitigen?

Spurensuche

- Kann die CPU alle Spuren vom “Tatort” beseitigen?
- Spuren von out-of-order Ausführung bleiben im Cache zurück

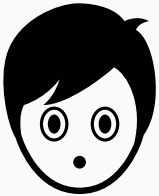
Spurensuche

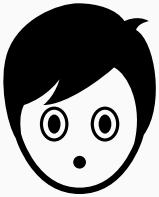
- Kann die CPU alle Spuren vom “Tatort” beseitigen?
- Spuren von out-of-order Ausführung bleiben im Cache zurück
- Cache ist normal nicht sichtbar, da Teil der Mikroarchitektur

Spurensuche

- Kann die CPU alle Spuren vom “Tatort” beseitigen?
- Spuren von out-of-order Ausführung bleiben im Cache zurück
- Cache ist normal nicht sichtbar, da Teil der Mikroarchitektur
- Flush+Reload macht Cache sichtbar

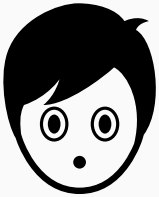
```
1 $wert = $dram[1000]; // Nur im Betriebssystem erlaubt!
2 // < Programm wird beendet >
3
4 $array[$wert] = 0;
```





```
1 $wert = $dram[1000]; // Nur im Betriebssystem erlaubt!
2 // < Programm wird beendet >
3
4 $array[$wert] = 0;
```

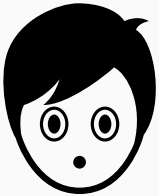
- Flush+Reload über alle Indices



```
1 $wert = $dram[1000]; // Nur im Betriebssystem erlaubt!
2 // < Programm wird beendet >
3
4 $array[$wert] = 0;
```

- Flush+Reload über alle Indices

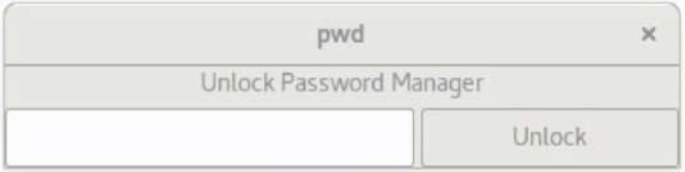
- Zugegriffener Index lädt schneller...



```
1 $wert = $dram[1000]; // Nur im Betriebssystem erlaubt!
2 // < Programm wird beendet >
3
4 $array[$wert] = 0;
```

- Flush+Reload über alle Indices

- Zugegriffener Index lädt schneller...
- ...und verrät damit den Wert



Terminal

File Edit View Search Terminal Help

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

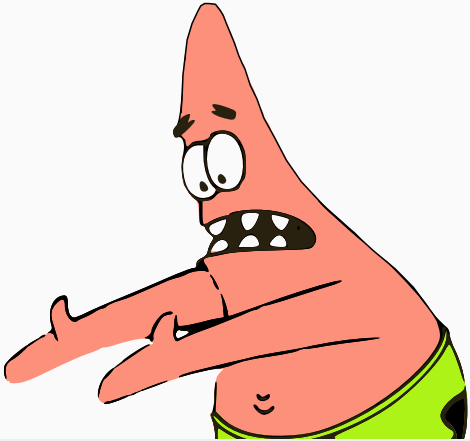
Und jetzt?...

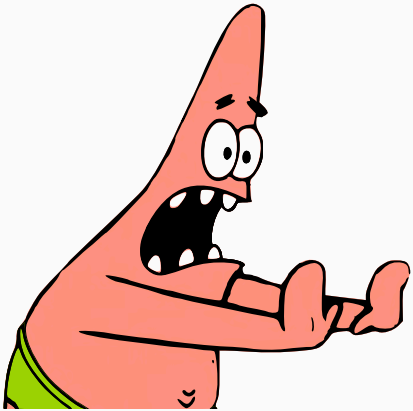
Take the kernel addresses...

- Das Problem: Betriebssystem ist sichtbar

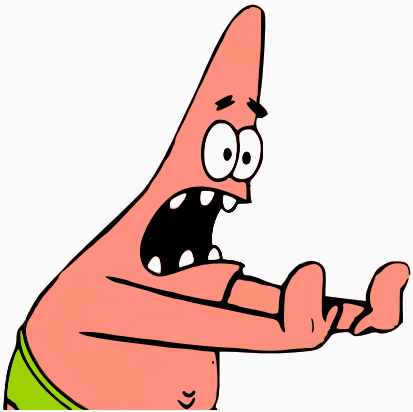
Take the kernel addresses...

- Das Problem: Betriebssystem ist sichtbar
- Warum nehmen wir nicht das Betriebssystem...





- ...und verstecken es wenn wir es nicht brauchen?



- ...und verstecken es wenn wir es nicht brauchen?
- Die Überprüfung der Berechtigung in Hardware ist nicht zuverlässig

Idea

- Wir lassen das Betriebssystem verschwinden

Idea

- Wir lassen das Betriebssystem verschwinden
- Geschützter Speicher ist unsichtbar

Idea

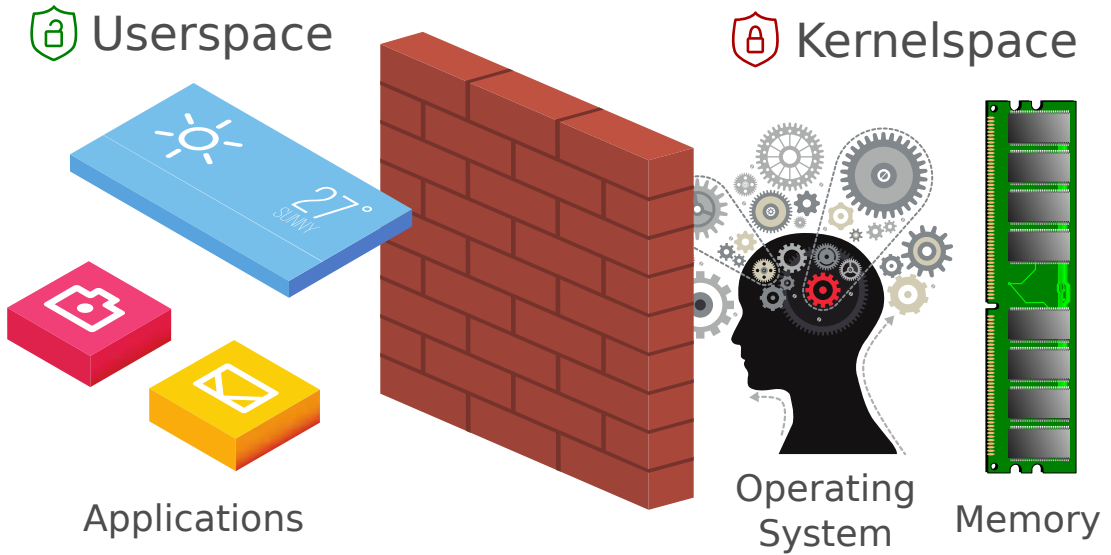
- Wir lassen das Betriebssystem verschwinden
- Geschützter Speicher ist unsichtbar
- Niemand kann auf geschützten Speicher zugreifen


Kernel Address Isolation to have Side channels Efficiently Removed

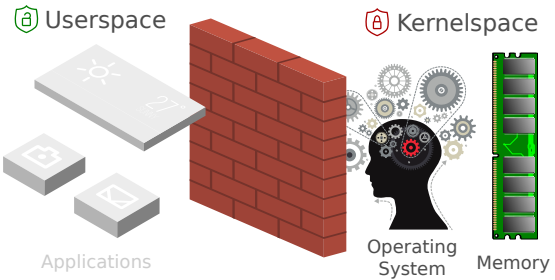
KAISER /ˈkʌɪzə/

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

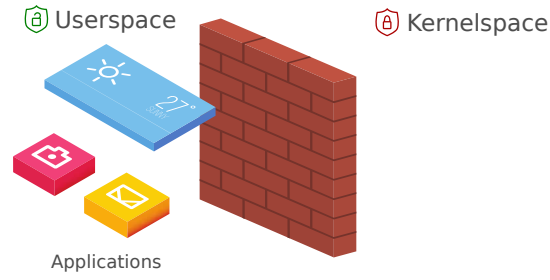
Kernel **A**ddress **I**solation to have **S**ide channels **E**fficiently **R**emoved

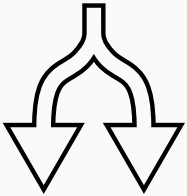

Userspace
Kernelspace
27°
Applications
Operating System
Memory

Kernel View

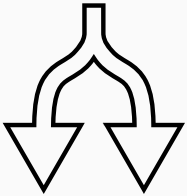


User View

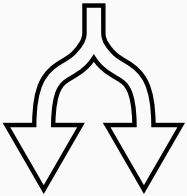




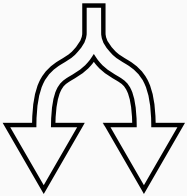
- Wir haben KAISER im Juli 2017 veröffentlicht



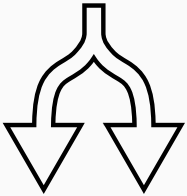
- Wir haben KAISER im Juli 2017 veröffentlicht
- Intel und andere haben es verbessert und in Linux als KPTI (Kernel Page Table Isolation) aufgenommen



- Wir haben KAISER im Juli 2017 veröffentlicht
- Intel und andere haben es verbessert und in Linux als KPTI (Kernel Page Table Isolation) aufgenommen
- Microsoft hat in Windows 10 etwas Ähnliches implementiert

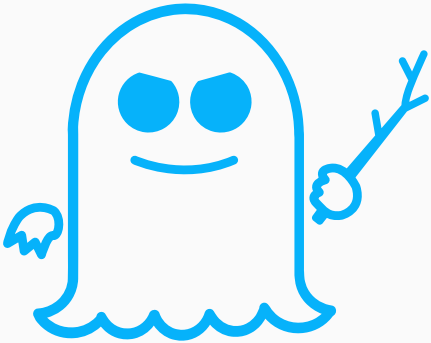


- Wir haben KAISER im Juli 2017 veröffentlicht
- Intel und andere haben es verbessert und in Linux als KPTI (Kernel Page Table Isolation) aufgenommen
- Microsoft hat in Windows 10 etwas Ähnliches implementiert
- Apple in macOS 10.13.2 ebenfalls ("Double Map")



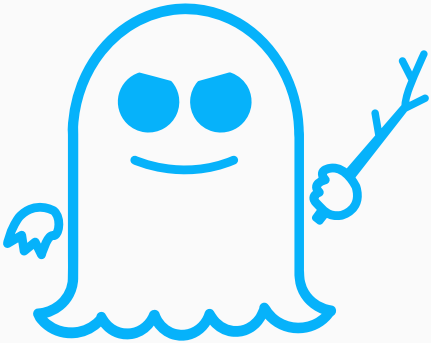
- Wir haben KAISER im Juli 2017 veröffentlicht
- Intel und andere haben es verbessert und in Linux als KPTI (Kernel Page Table Isolation) aufgenommen
- Microsoft hat in Windows 10 etwas Ähnliches implementiert
- Apple in macOS 10.13.2 ebenfalls ("Double Map")
- Grundidee ist immer gleich: das Betriebssystem verstecken

MELTDOWN

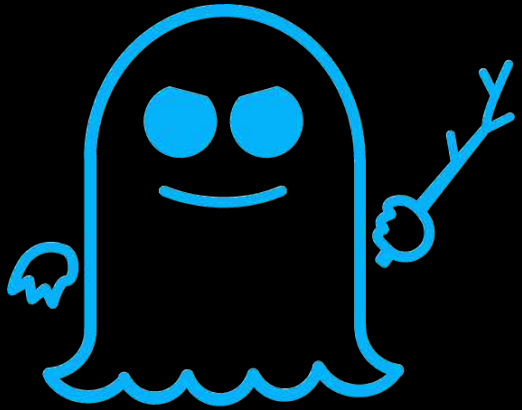


SPECTRE

MELTDOWN



SPECTRE



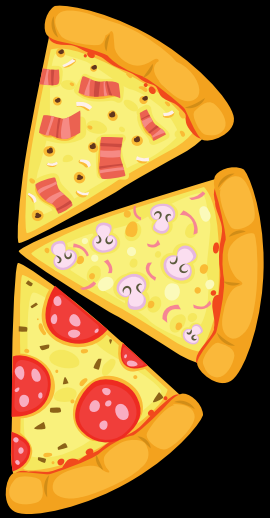
SPECTRE

PIZZA

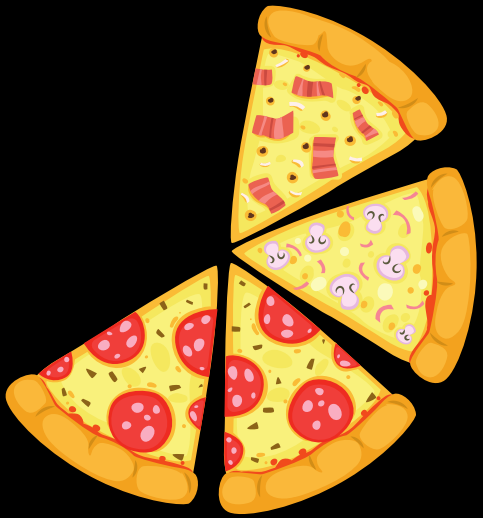
SPECIAL RECIPES

Prosciutto

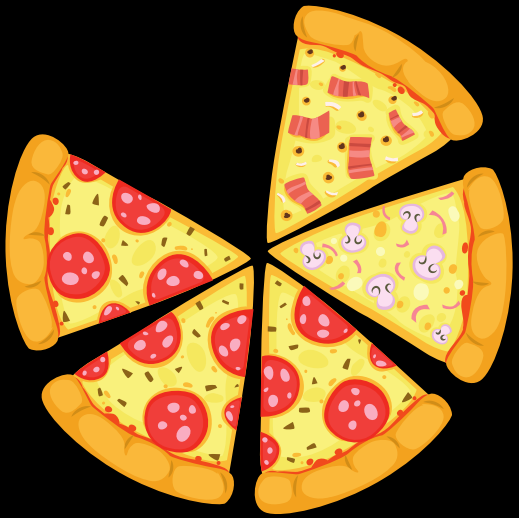
Funghi



Diavolo



Diavolo



Diavolo

Diavolo


»A table for 6 please«

Speculative Cooking


»A table for 6 please«


PIZZA
SPECIAL RECIPES


PIZZA
SPECIAL RECIPES

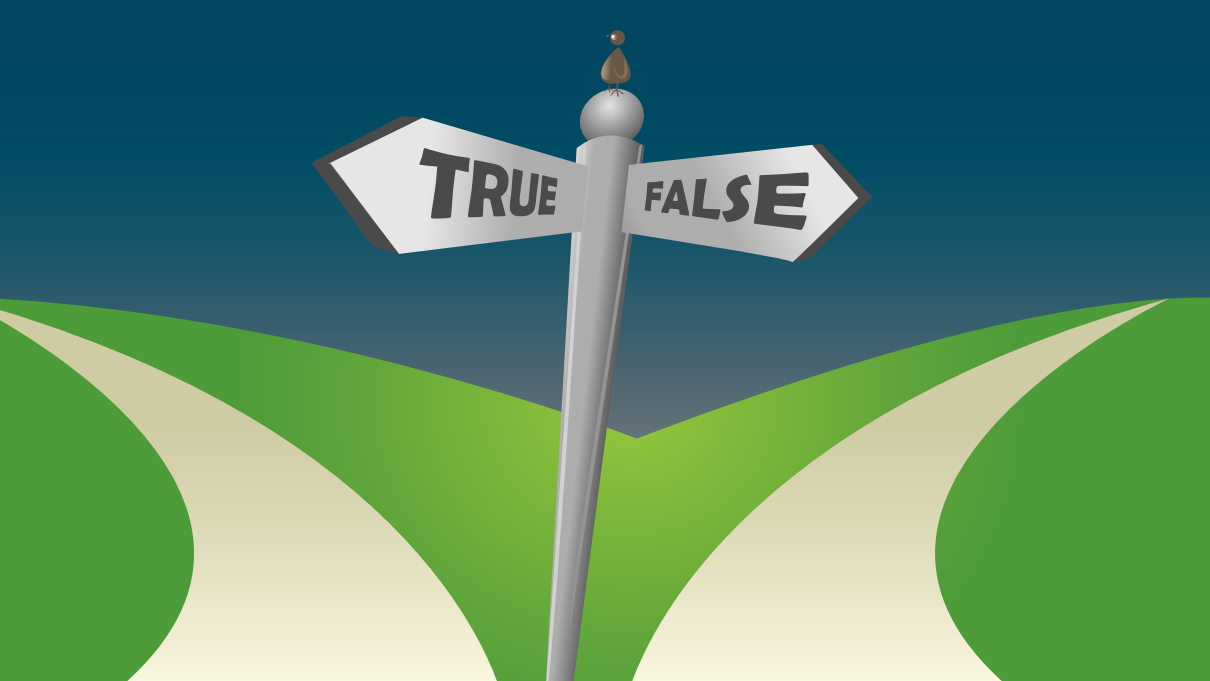

Pizza


PIZZA
SPECIAL RECIPES


Pizza

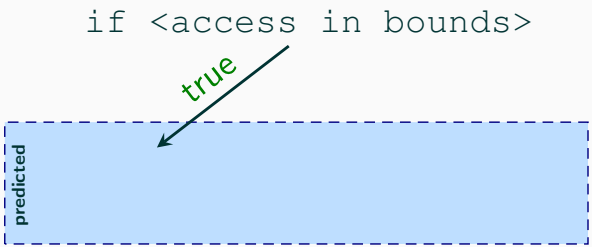

PIZZA
SPECIAL RECIPES

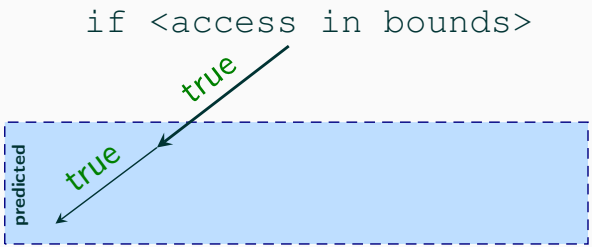

Pizza

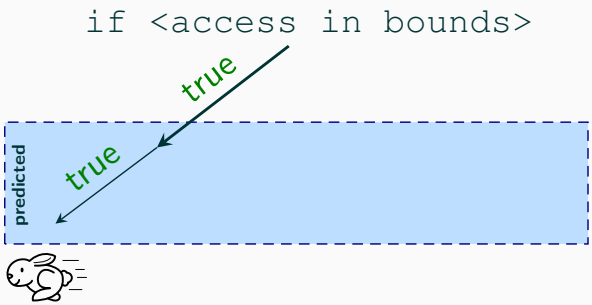

TRUE
FALSE

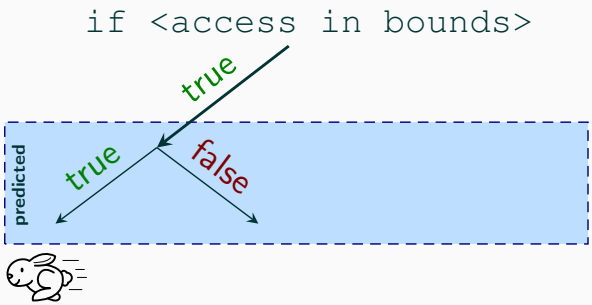
```
1 $werte = array(1, 2, 4, 8);
2 if($index < count($werte)) {
3   echo $werte[$index];
4 }
```

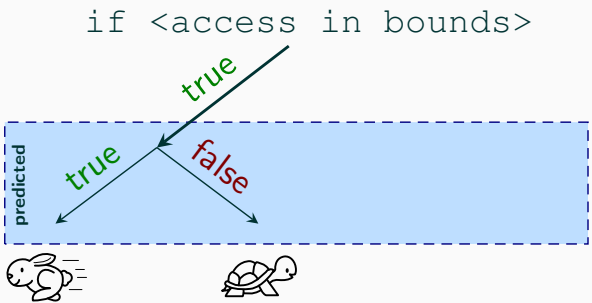
```
if <access in bounds>
```

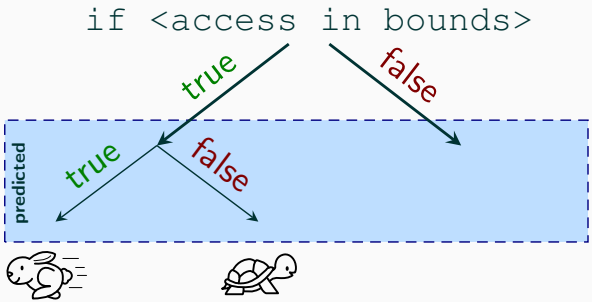

if <access in bounds>
true
predicted

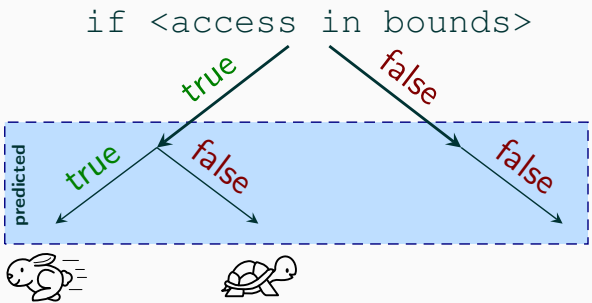

if <access in bounds>
true
predicted
true

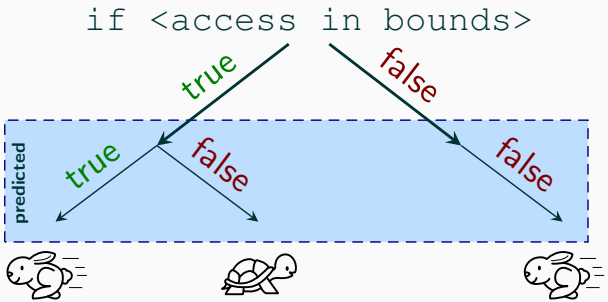

if <access in bounds>
true
predicted
true

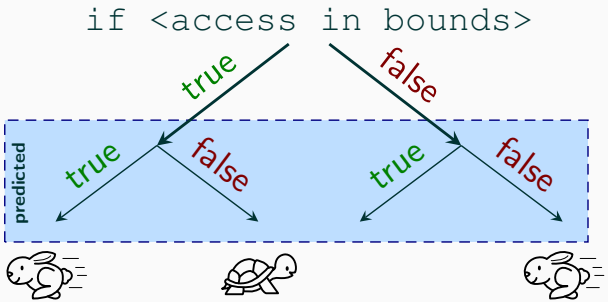

if <access in bounds>
true
predicted
true
false

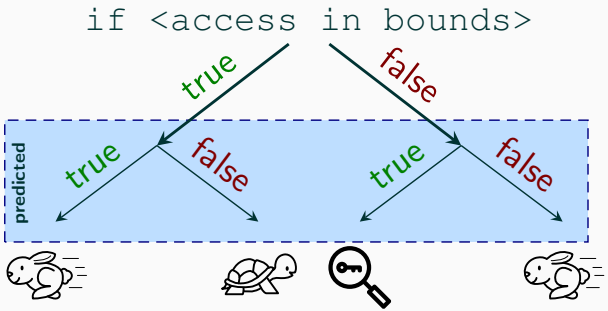

if <access in bounds>
true
predicted
true
false


if <access in bounds>
true
false
predicted
true
false

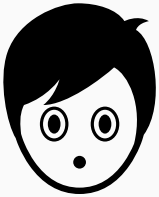

if <access in bounds>
true
false
predicted
true
false
false


if <access in bounds>
true
false
predicted
true
false
false

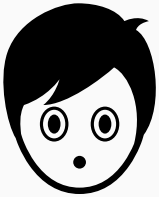

if <access in bounds>
true
false
predicted
true
false
true
false


if <access in bounds>
true
false
predicted
true
false
true
false

```
1 $werte = array(1, 2, 4, 8, 84);
2 //                          ^-- nicht zugreifbar
3 //          v-- nur die ersten 4 Werte sind zugreifbar
4 if($index < 4) {
5   echo $array[$werte[$index]];
6 //         ^-- gleicher Trick wie bei Meltdown
7 }
```

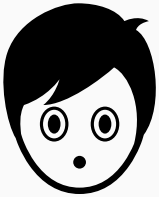


- Flush+Reload über alle Indices



- Flush+Reload über alle Indices

- Zugegriffener Index lädt schneller...



- Flush+Reload über alle Indices

- Zugegriffener Index lädt schneller...
- ...und verrät damit wieder den Wert

- Spectre kann den eigenen Speicher auslesen

- Spectre kann den eigenen Speicher auslesen
- "Überzeugt" andere Programme ihre Geheimnisse zu offenbaren

- Spectre kann den eigenen Speicher auslesen
- "Überzeugt" andere Programme ihre Geheimnisse zu offenbaren
- Basiert wieder auf Flush+Reload

- Spectre kann den eigenen Speicher auslesen
- "Überzeugt" andere Programme ihre Geheimnisse zu offenbaren
- Basiert wieder auf Flush+Reload
- Schwer zu beheben, KAISER hilft nicht dagegen

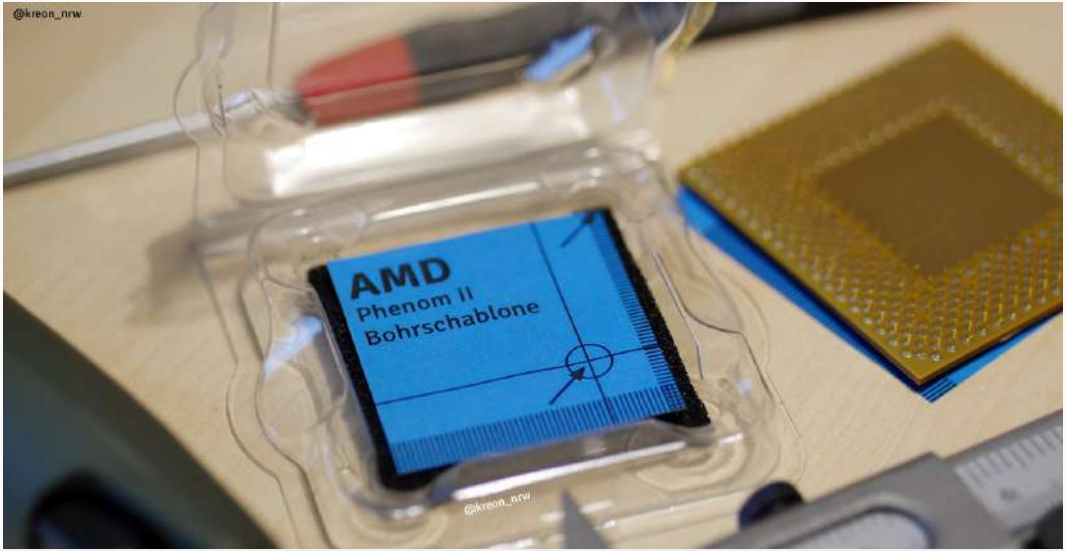
- Triviale Lösung: Spekulation deaktivieren

- Triviale Lösung: Spekulation deaktivieren
- Dadurch können keine Fehler passieren

- Triviale Lösung: Spekulation deaktivieren
- Dadurch können keine Fehler passieren
- Problem: Performance!

- Triviale Lösung: Spekulation deaktivieren
- Dadurch können keine Fehler passieren
- Problem: Performance!
- Und: Wie kann man es deaktivieren?

- Triviale Lösung: Spekulation deaktivieren
- Dadurch können keine Fehler passieren
- Problem: Performance!
- Und: Wie kann man es deaktivieren?
- Spekulative Ausführung ist ein wichtiger Baustein von Prozessoren


@kreon_nrw
AMD
Phenom II
Bohrschablone
@kreon_nrw


GHOSTBUSTERS

- Befehl um Spekluation zu stoppen

- Befehl um Spekluation zu stoppen

→ nach jedem `if` einfügen

- Befehl um Spekluation zu stoppen

→ nach jedem `if` einfügen

- Versuche zu automatisieren noch nicht ausgereift

- Befehl um Spekluation zu stoppen

→ nach jedem `if` einfügen

- Versuche zu automatisieren noch nicht ausgereift
- Außerdem: Performance! Bis zu 440 % langsamer


GHOSTBUSTERS

- Es gibt noch weitere Varianten von Spectre

- Es gibt noch weitere Varianten von Spectre
- Können teilweise durch CPU Updates und Betriebssystem Updates verhindert werden

- Es gibt noch weitere Varianten von Spectre
- Können teilweise durch CPU Updates und Betriebssystem Updates verhindert werden
- Werden erst in einigen Jahren wirklich behoben sein...

- Es gibt noch weitere Varianten von Spectre
- Können teilweise durch CPU Updates und Betriebssystem Updates verhindert werden
- Werden erst in einigen Jahren wirklich behoben sein...
- ...oder gar nie?

Was nicht funktioniert

- Zeitmessung aus der CPU entfernen

Was nicht funktioniert

- Zeitmessung aus der CPU entfernen

→ Kann man sich selbst bauen

Was nicht funktioniert

- Zeitmessung aus der CPU entfernen

→ Kann man sich selbst bauen

- Flush Befehl entfernen

Was nicht funktioniert

- Zeitmessung aus der CPU entfernen

→ Kann man sich selbst bauen

- Flush Befehl entfernen

→ Kann man auch nachbauen

Was nicht funktioniert

- Zeitmessung aus der CPU entfernen

→ Kann man sich selbst bauen

- Flush Befehl entfernen

→ Kann man auch nachbauen

- Caches entfernen

Was nicht funktioniert

- Zeitmessung aus der CPU entfernen

→ Kann man sich selbst bauen

- Flush Befehl entfernen

→ Kann man auch nachbauen

- Caches entfernen

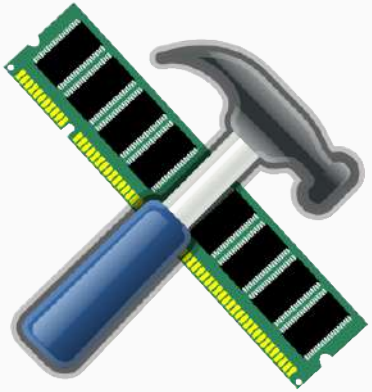
→ Performanceverlust von Faktor 100

Kann man Daten auch verändern?

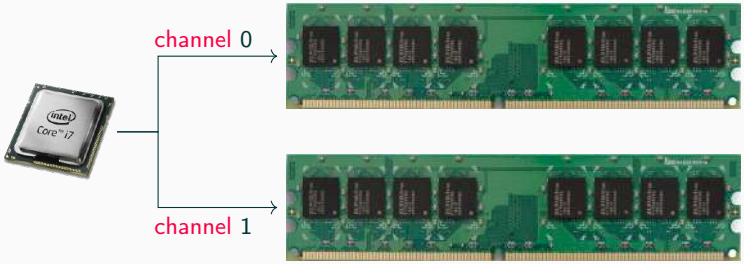
- Nicht mit Meltdown oder Spectre...

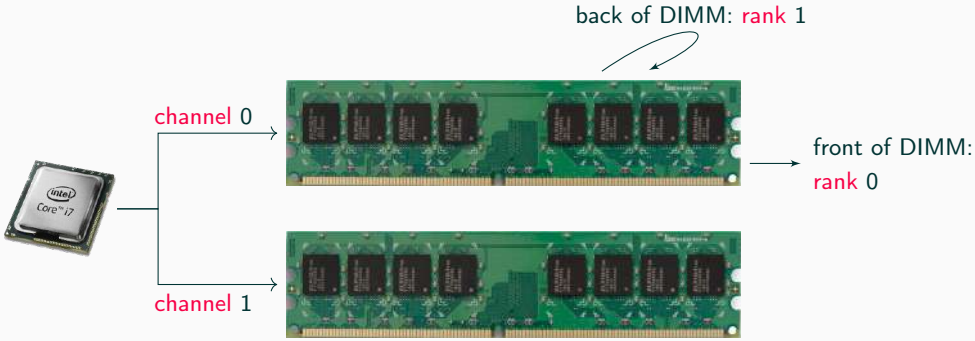
- Nicht mit Meltdown oder Spectre...
- ...auch nicht mit Flush+Reload

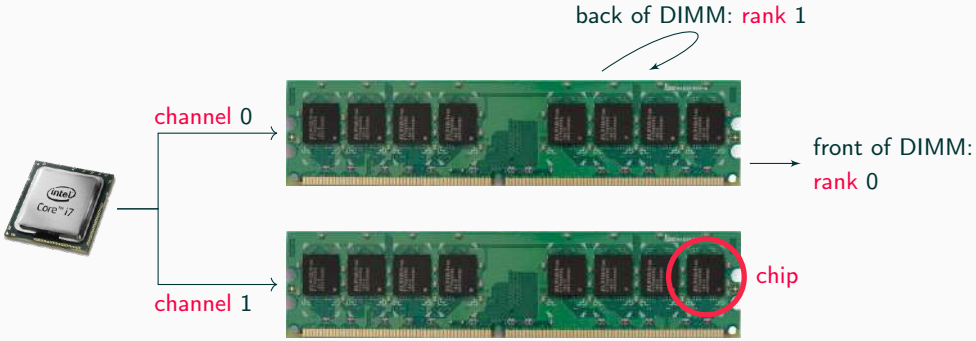
- Nicht mit Meltdown oder Spectre...
- ...auch nicht mit Flush+Reload
- Jedoch mit einem anderen mikroarchitekturellen Angriff



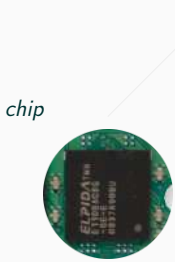
Rowhammer

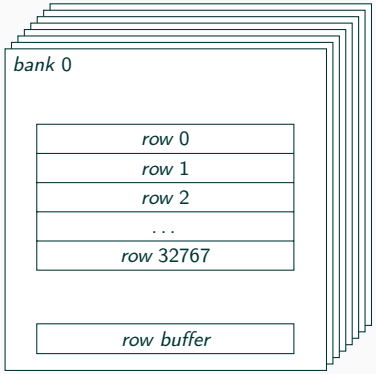

channel 0
channel 1
intel
Core™ i7


back of DIMM: rank 1
channel 0
front of DIMM:
rank 0
channel 1
intel
Core™ i7


back of DIMM: rank 1
channel 0
front of DIMM:
rank 0
chip
channel 1

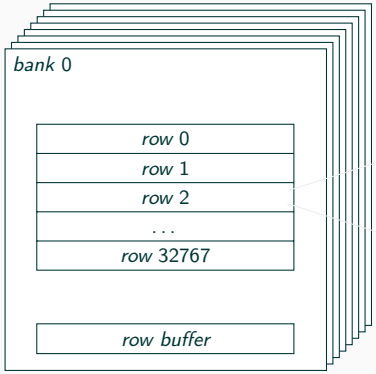
DRAM organization





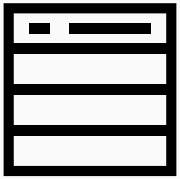
DRAM organization

chip



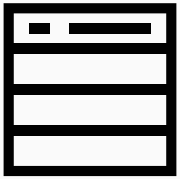
64k Zellen
1 Kondensator +
1 Transistor pro Zelle

Row buffer



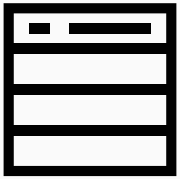
- DRAM kann immer nur eine gesamte Zeile (*row*) lesen

Row buffer



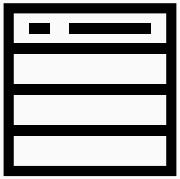
- DRAM kann immer nur eine gesamte Zeile (*row*) lesen
- Lesen leert die Kondensatoren

Row buffer



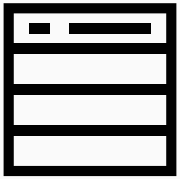
- DRAM kann immer nur eine gesamte Zeile (*row*) lesen
- Lesen leert die Kondensatoren
- Gelesene Werte werden zwischengespeichert...

Row buffer



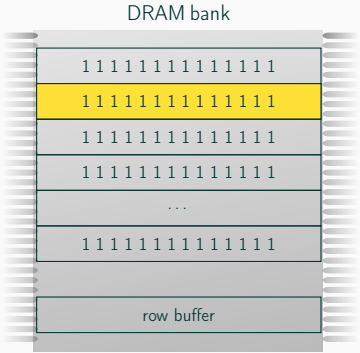
- DRAM kann immer nur eine gesamte Zeile (*row*) lesen
- Lesen leert die Kondensatoren
- Gelesene Werte werden zwischengespeichert...
- ..und danach wieder zurück geschrieben

Row buffer



- DRAM kann immer nur eine gesamte Zeile (*row*) lesen
- Lesen leert die Kondensatoren
- Gelesene Werte werden zwischengespeichert...
- ..und danach wieder zurück geschrieben
- $\rightarrow$ Zwischenspeicher heißt *Row buffer*

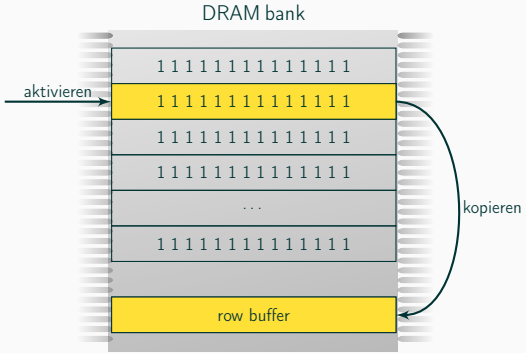
Lesen von DRAM



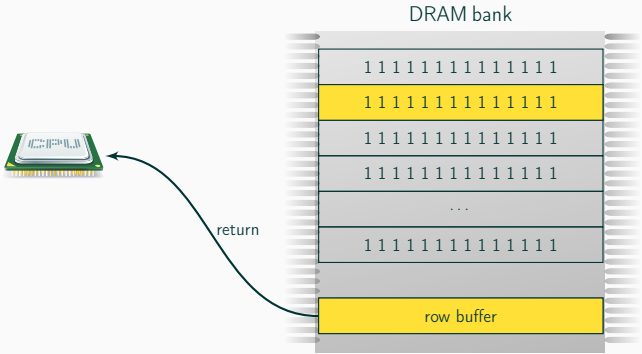
CPU liest Zeile 1:

→ Aktivieren, in Row buffer kopieren

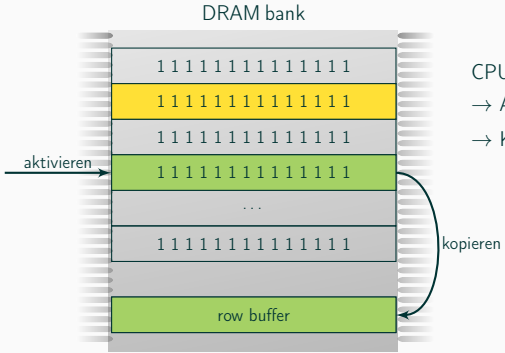
Lesen von DRAM



Lesen von DRAM



Lesen von DRAM

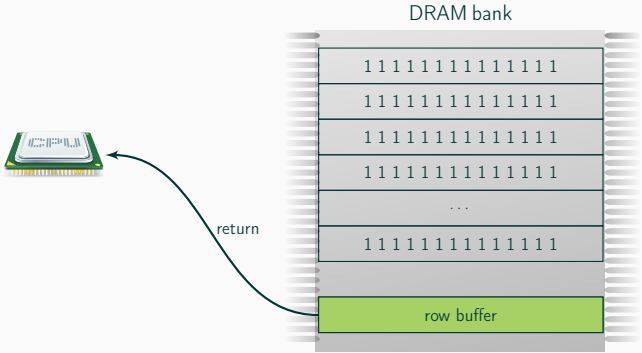


CPU liest Zeile 2:

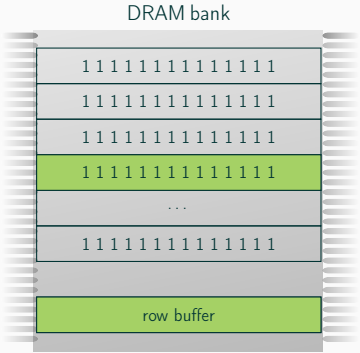
→ Aktivieren, in Row buffer kopieren

→ Konflikt → langsam

Lesen von DRAM



Lesen von DRAM

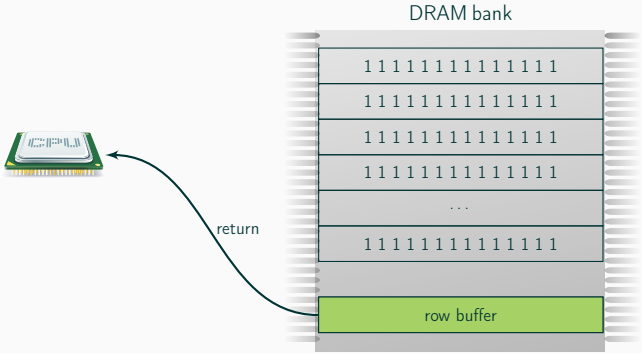


CPU liest Zeile 2:

→ Bereits im Row buffer

→ Treffer → schnell

Lesen von DRAM

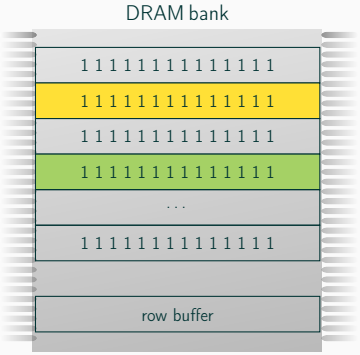


CPU liest Zeile 2:

→ Bereits im Row buffer

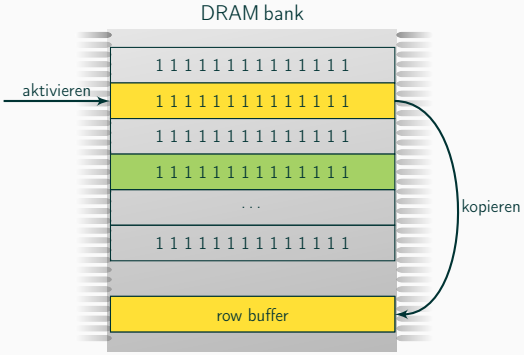
→ Treffer → schnell

Rowhammer

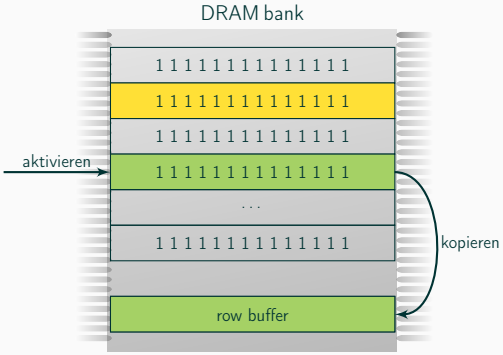


- Zellen verlieren Speicher (Leckströme) → Auffrischen notwendig
- Schneller Zugriff entlädt Nachbarzellen schneller → Rowhammer

Rowhammer

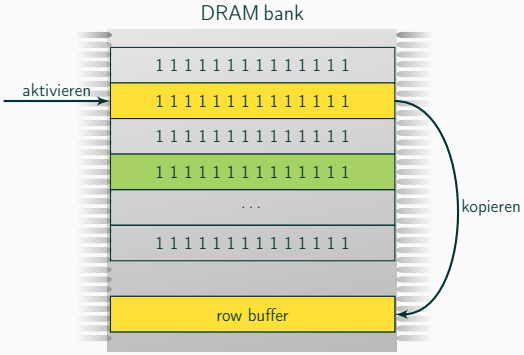


- Zellen verlieren Speicher (Leckströme) → Auffrischen notwendig
- Schneller Zugriff entlädt Nachbarzellen schneller → Rowhammer

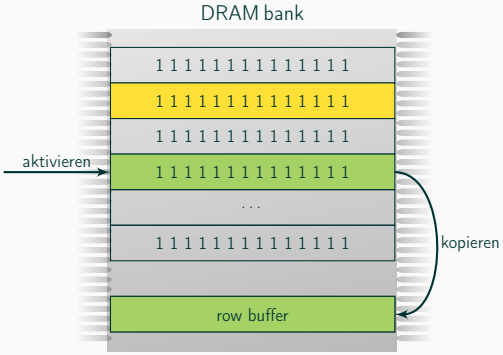


- Zellen verlieren Speicher (Leckströme) → Auffrischen notwendig
- Schneller Zugriff entlädt Nachbarzellen schneller → Rowhammer

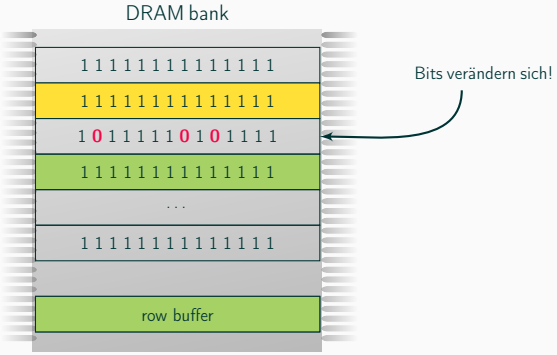
Rowhammer



- Zellen verlieren Speicher (Leckströme) → Auffrischen notwendig
- Schneller Zugriff entlädt Nachbarzellen schneller → Rowhammer



- Zellen verlieren Speicher (Leckströme) → Auffrischen notwendig
- Schneller Zugriff entlädt Nachbarzellen schneller → Rowhammer



- Zellen verlieren Speicher (Leckströme) → Auffrischen notwendig
- Schneller Zugriff entlädt Nachbarzellen schneller → Rowhammer

Was kann ein Bit anrichten?

- Ein (oder mehrere) Bits verändern sich

Was kann ein Bit anrichten?

- Ein (oder mehrere) Bits verändern sich
- Passiert irgendwo im Speicher

Was kann ein Bit anrichten?

- Ein (oder mehrere) Bits verändern sich
- Passiert irgendwo im Speicher
- Vermutlich in einem anderen Programm

Was kann ein Bit anrichten?

- Ein (oder mehrere) Bits verändern sich
- Passiert irgendwo im Speicher
- Vermutlich in einem anderen Programm
- Ist das ein Problem?

Was kann ein Bit anrichten?

```
1 if($privilege >= ADMIN) {
2   // Befehl mit Adminrechten ausführen
3 } else {
4   // Befehlt mit Benutzerrechten ausführen
5 }
```

Was kann ein Bit anrichten?

```
1 if($privilege >= ADMIN) {
2   // Befehl mit Adminrechten ausführen
3 } else {
4   // Befehlt mit Benutzerrechten ausführen
5 }
```

“>=” ist im Speicher `00111110 00111101`

Was kann ein Bit anrichten?

```
1 if($privilege >= ADMIN) {
2   // Befehl mit Adminrechten ausführen
3 } else {
4   // Befehlt mit Benutzerrechten ausführen
5 }
```

“>=” ist im Speicher `00111110 00111101`

“<=“ ist im Speicher `00111100 00111101`

Auswirkung

```
1 if($privilege >= ADMIN) {
2   // Befehl mit Adminrechten ausführen
3 } else {
4   // Befehlt mit Benutzerrechten ausführen
5 }
```

```
1 if($privilege >= ADMIN) {
2   // Befehl mit Adminrechten ausführen
3 } else {
4   // Befehlt mit Benutzerrechten ausführen
5 }
```

```
1 if($privilege <= ADMIN) {
2   // Befehl mit Adminrechten ausführen
3 } else {
4   // Befehlt mit Benutzerrechten ausführen
5 }
```

- Funktioniert auch mit echten Anwendungen

- Funktioniert auch mit echten Anwendungen
- `sudo`: (Linux Tool um Befehle als Admin auszuführen)

- Funktioniert auch mit echten Anwendungen
- `sudo`: (Linux Tool um Befehle als Admin auszuführen)
  - Passwort Überprüfung invertieren

- Funktioniert auch mit echten Anwendungen
- `sudo`: (Linux Tool um Befehle als Admin auszuführen)
  - Passwort Überprüfung invertieren
  - Nur mit dem falschen Passwort wird man Administrator

- Funktioniert auch mit echten Anwendungen
- `sudo`: (Linux Tool um Befehle als Admin auszuführen)
  - Passwort Überprüfung invertieren
  - Nur mit dem falschen Passwort wird man Administrator
- Wurde sogar im Browser demonstriert

Real World

- Funktioniert auch mit echten Anwendungen
- `sudo`: (Linux Tool um Befehle als Admin auszuführen)
  - Passwort Überprüfung invertieren
  - Nur mit dem falschen Passwort wird man Administrator
- Wurde sogar im Browser demonstriert
- Und vor 2 Wochen auch über das Netzwerk

Was lernen wir daraus?

- Software Seitenkanäle gibt es seit vielen Jahren

Was lernen wir daraus?

- Software Seitenkanäle gibt es seit vielen Jahren
- Waren nie “interessant”

Was lernen wir daraus?

- Software Seitenkanäle gibt es seit vielen Jahren
- Waren nie “interessant”
- Erst als wir gezeigt haben, dass man damit Daten lesen kann ($\rightarrow$ Meltdown, Spectre)

Was lernen wir daraus?

- Software Seitenkanäle gibt es seit vielen Jahren
- Waren nie “interessant”
- Erst als wir gezeigt haben, dass man damit Daten lesen kann ($\rightarrow$ Meltdown, Spectre)
- Optimierungen führen neue Seitenkanäle ein

Wenn man das Handbuch liest...

- Niemand liest das Handbuch zur CPU

Wenn man das Handbuch liest...

- Niemand liest das Handbuch zur CPU
- Dort ist Spectre sogar dokumentiert

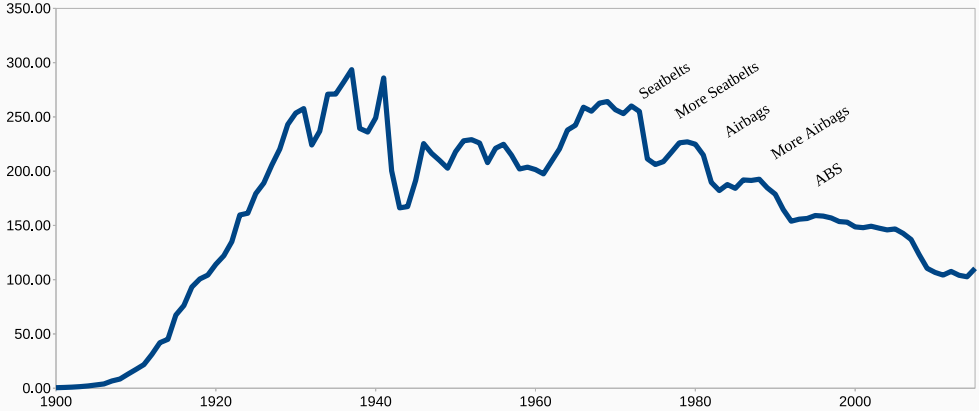
Wenn man das Handbuch liest...

- Niemand liest das Handbuch zur CPU
- Dort ist Spectre sogar dokumentiert
- Allerdings wurde es nicht als Sicherheitsproblem erkannt

Wenn man das Handbuch liest...

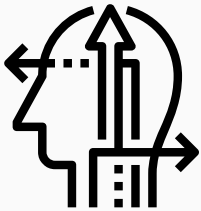
- Niemand liest das Handbuch zur CPU
- Dort ist Spectre sogar dokumentiert
- Allerdings wurde es nicht als Sicherheitsproblem erkannt
- Nur ein Nebeneffekt ohne sichtbare Auswirkung

Was lernen wir daraus?



Todesfälle durch Autos in den USA pro Jahr

Eine Chance



Die Entdeckung gibt uns die Chance

- neue Prozessorarchitekturen zu entwickeln
- mehr an Sicherheit zu denken (wie in der Autoindustrie)
- Kompromisse zwischen Sicherheit und Performance zu finden

Fazit

- Seitenkanal-Angriffe wurden zu lange unterschätzt
  - Grundlegende Konzepte gibt es schon lange (Flush+Reload)
- Es wird zu wenig Wert auf Sicherheit gelegt
  - Wir brauchen mehr Fokus auf Sicherheit
  - Performance darf nicht mehr das einzige Kriterium bei Prozessoren sein
- Wir brauchen Nachwuchs (euch!) um Probleme zu entdecken und Lösungen zu finden

Fragen

Flush+Reload, Meltdown, Spectre, Rowhammer

Zu Risiken und Nebenwirkungen fragen Sie Ihr Prozessorhandbuch oder Ihren Seitenkanal-Forscher

---

Michael Schwarz (@misc0110)

25.05.2018

www.iaik.tugraz.at