Software-basierte Mikroarchitekturangriffe

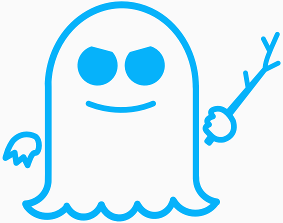
Daniel Gruss

7.5.2018

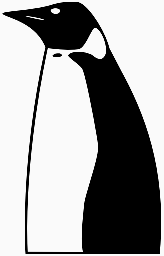
Technische Universität Graz

Folgen / Auswirkungen meiner Dissertation

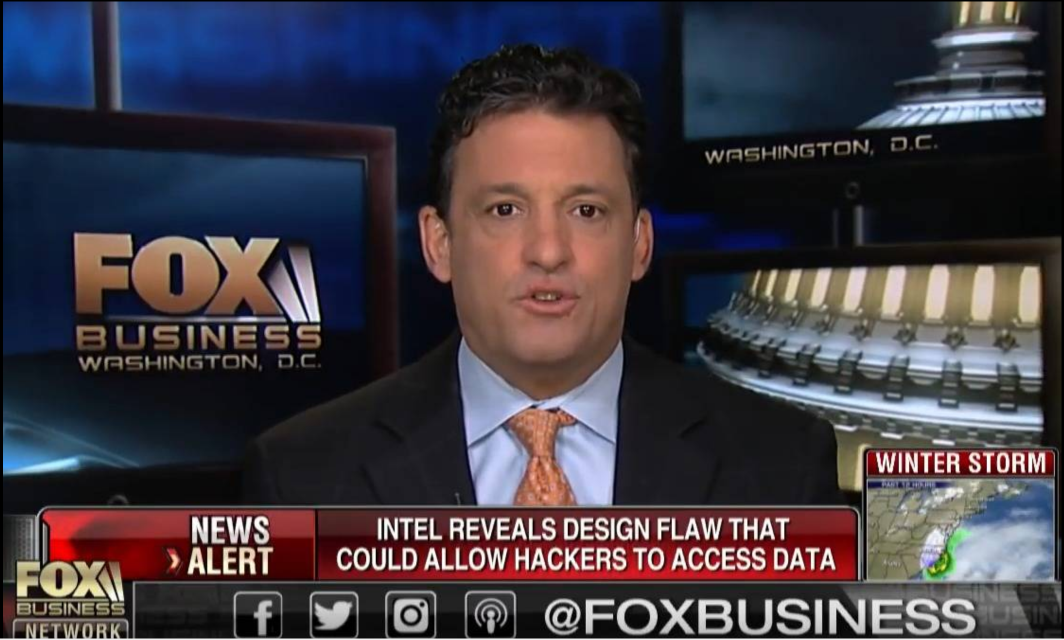
MELTDOWN



SPECTRE



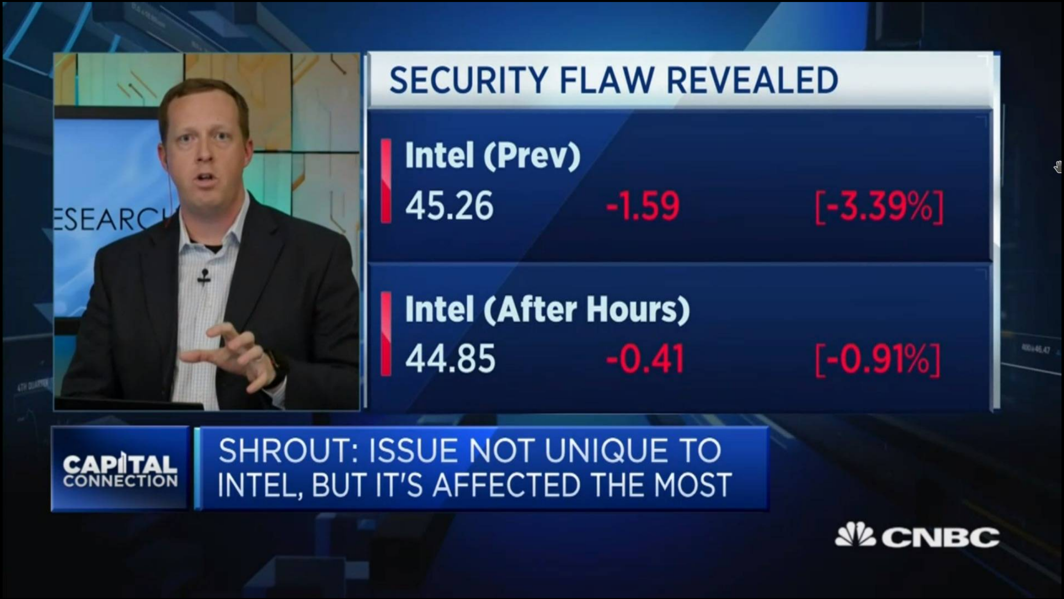
KAISER


FOX BUSINESS
WASHINGTON, D.C.
WASHINGTON, D.C.
WINTER STORM
NEWS ALERT
FOX BUSINESS NETWORK
@FOXBUSINESS

INTEL REVEALS DESIGN FLAW THAT COULD ALLOW HACKERS TO ACCESS DATA

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
£:HKS 10.58
EURO:£ 0.891



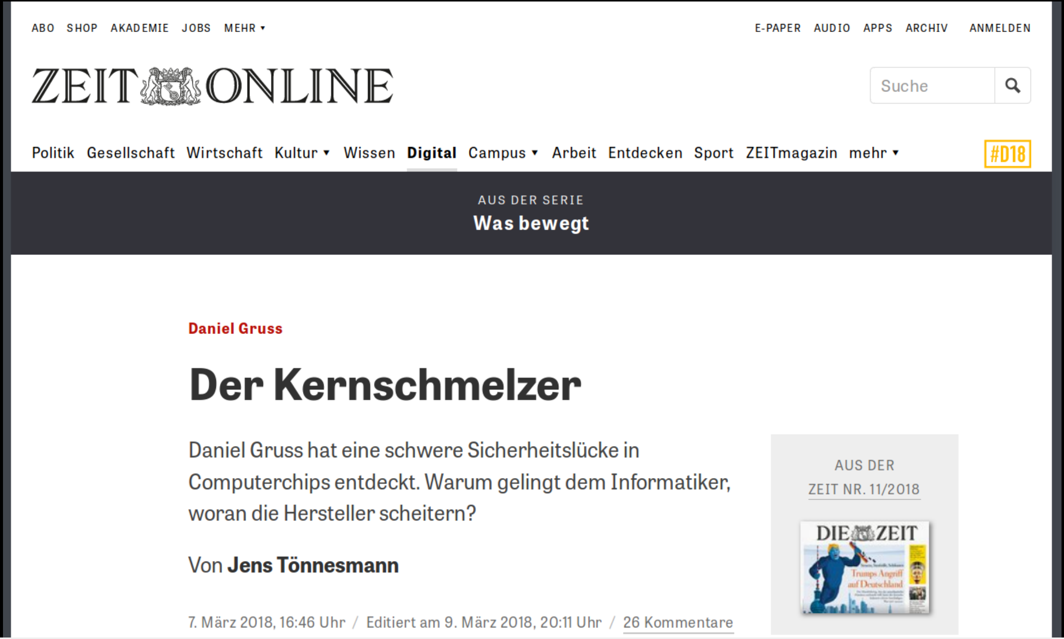
SECURITY FLAW REVEALED

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


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

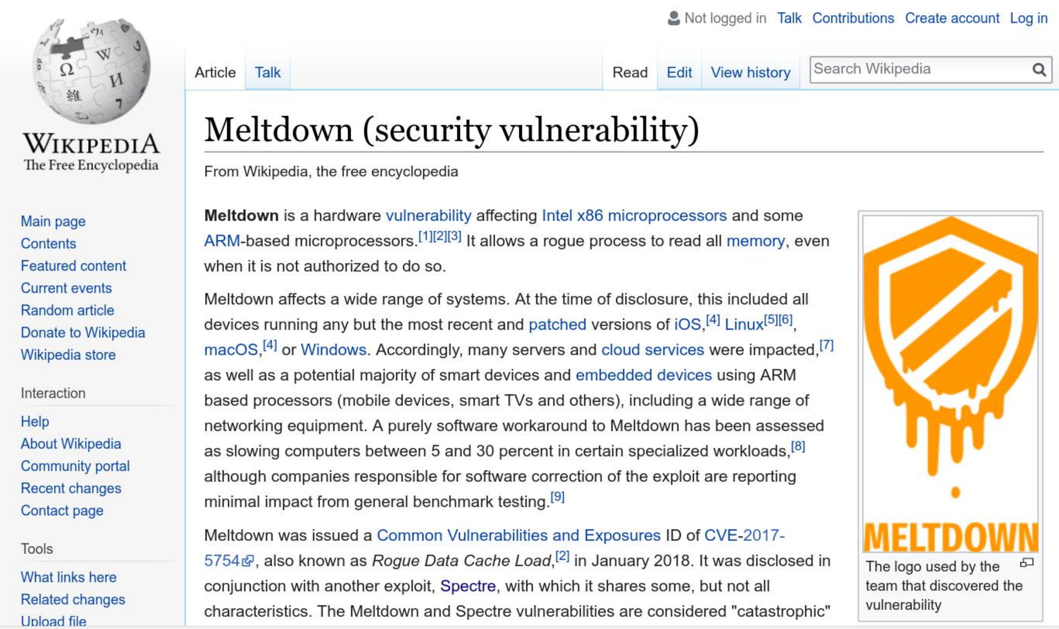
Daniel Gruss hat eine schwere Sicherheitslücke in Computerchips entdeckt. Warum gelingt dem Informatiker, woran die Hersteller scheitern?

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
Trumps Angriff auf Deutschland


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

Meltdown (security vulnerability)

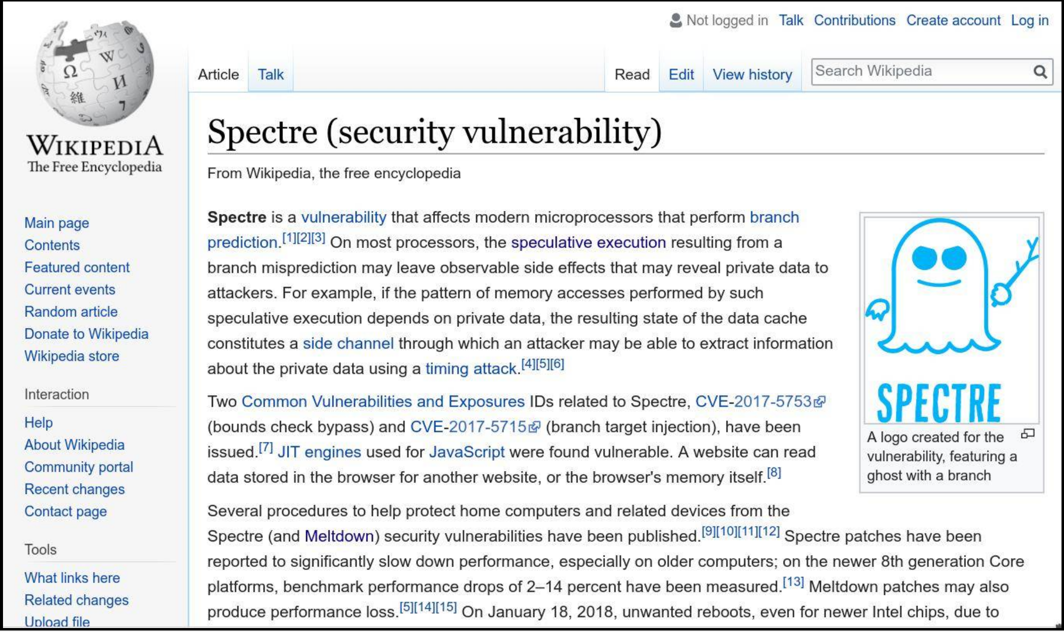
From Wikipedia, the free encyclopedia

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"

The logo used by the team that discovered the vulnerability


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
SPECTRE

Spectre (security vulnerability)

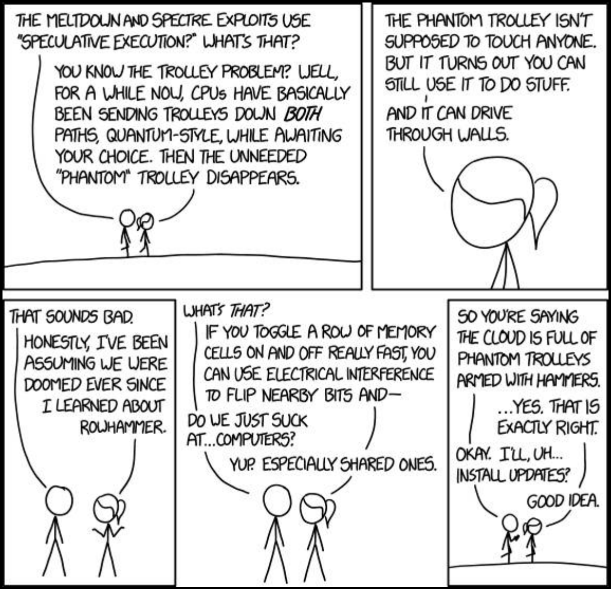
From Wikipedia, the free encyclopedia

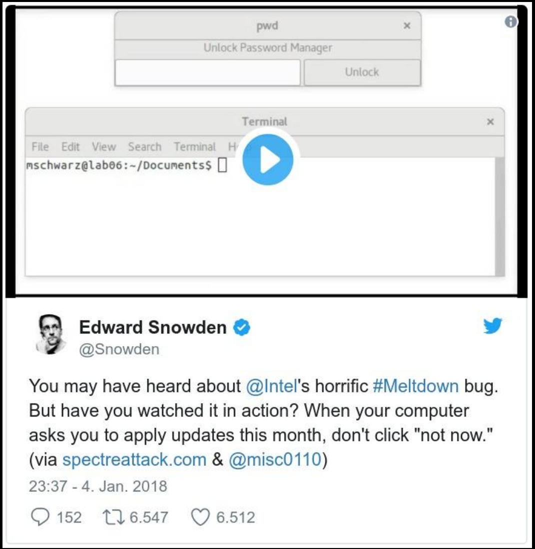
Spectre is a vulnerability that affects modern microprocessors that perform branch prediction.[1][2][3] On most processors, the speculative execution resulting from a branch misprediction may leave observable side effects that may reveal private data to attackers. For example, if the pattern of memory 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]

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]

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

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 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.


pwd
×
Unlock Password Manager
Unlock
Terminal
×
File Edit View Search Terminal H

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

Edward Snowden
@Snowden

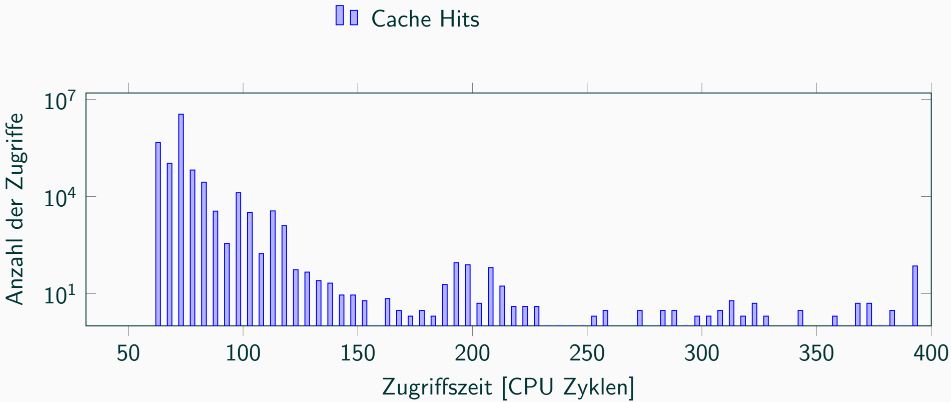
You may have heard about @Intel's horrific #Meltdown bug. But have you watched it in action? When your computer asks you to apply updates this month, don't click "not now." (via spectreattack.com & @misc0110)

23:37 - 4. Jan. 2018

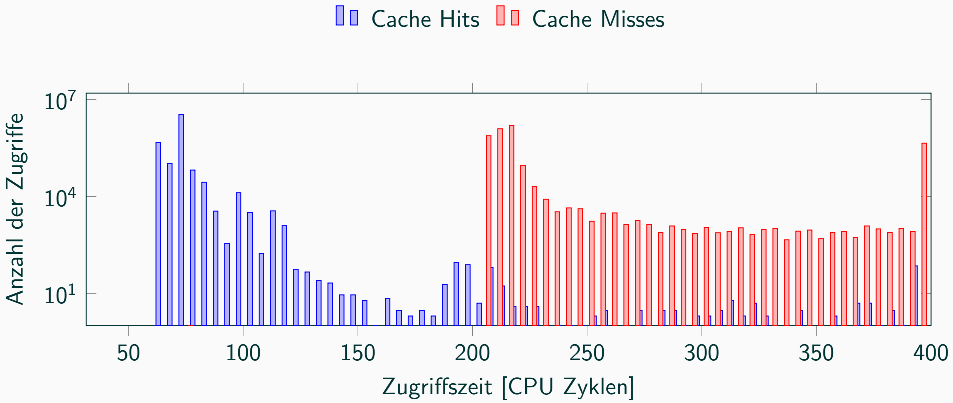
152 6.547 6.512

National Geographic

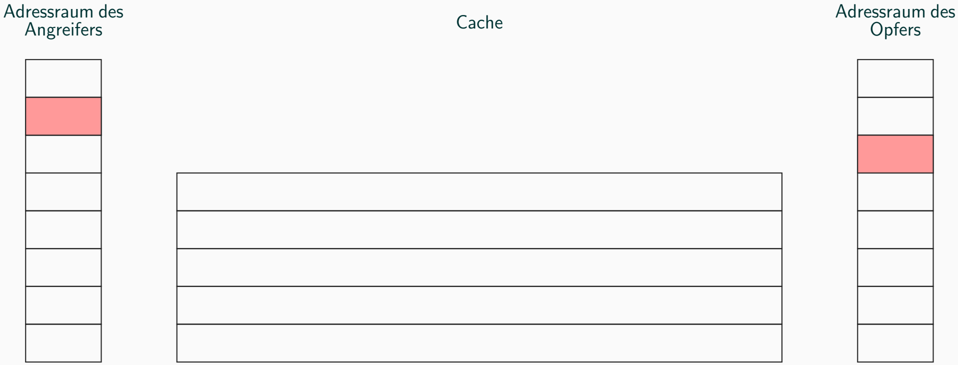
Einfluss des Caches auf die Zugriffslatenz



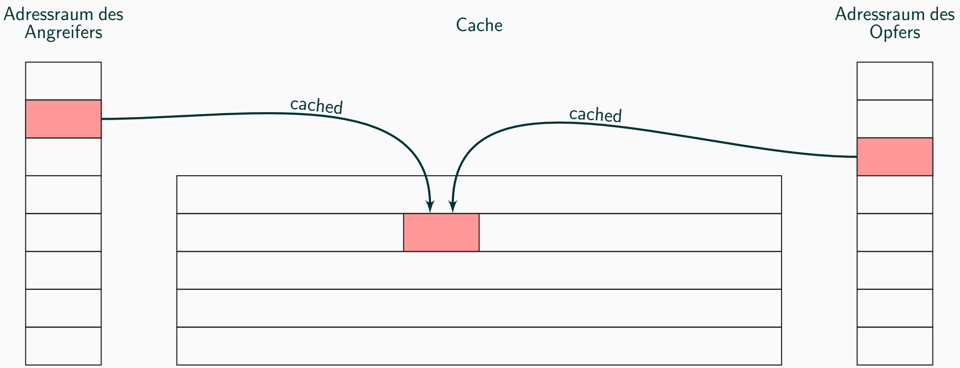
Einfluss des Caches auf die Zugriffslatenz



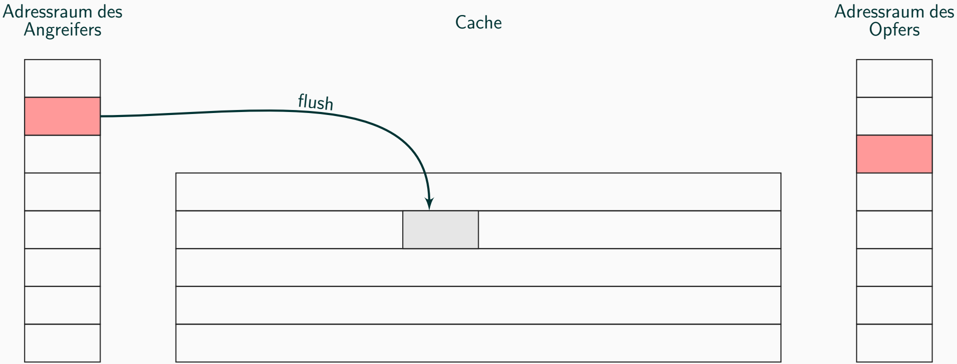
Flush+Reload


Adressraum des Angreifers
Cache
Adressraum des Opfers

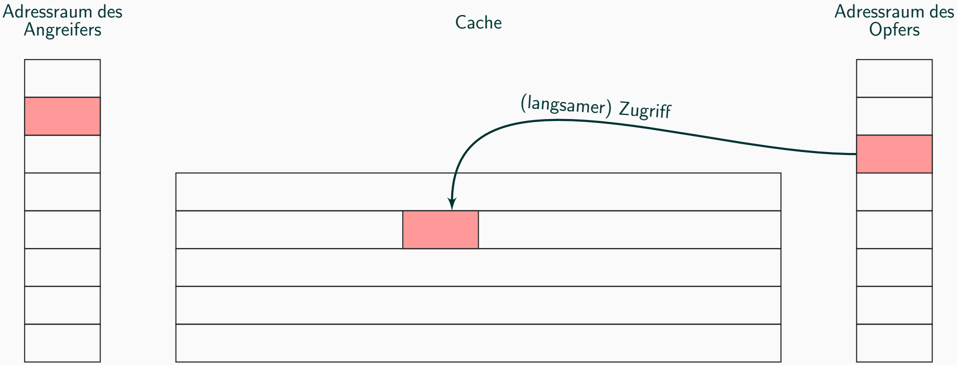
Flush+Reload


Adressraum des Angreifers
Cache
Adressraum des Opfers
cached
cached

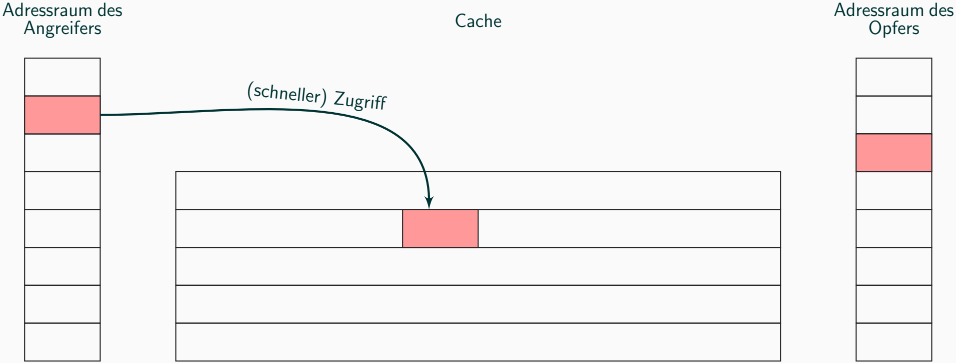
Flush+Reload


Adressraum des Angreifers
Cache
Adressraum des Opfers
flush

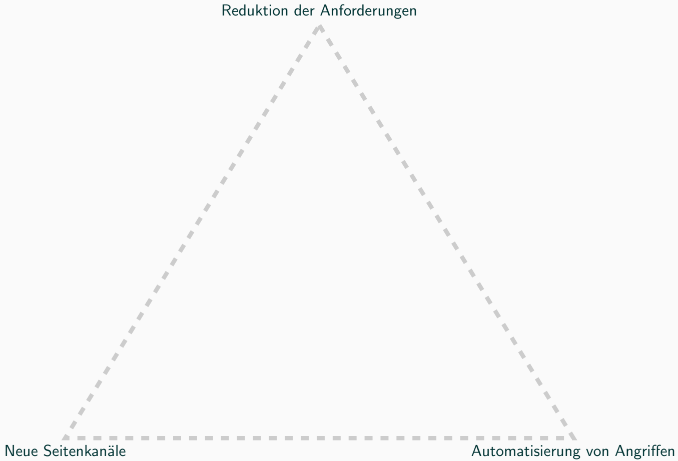
Flush+Reload


Adressraum des Angreifers
Cache
Adressraum des Opfers
(langsamer) Zugriff

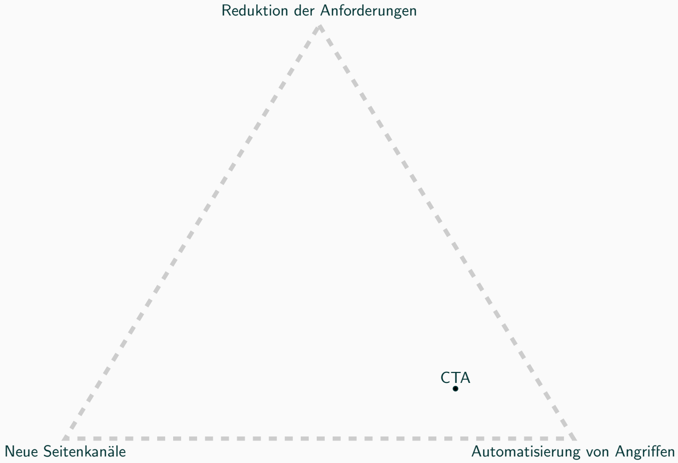
Flush+Reload


Adressraum des Angreifers
Cache
Adressraum des Opfers
(schneller) Zugriff

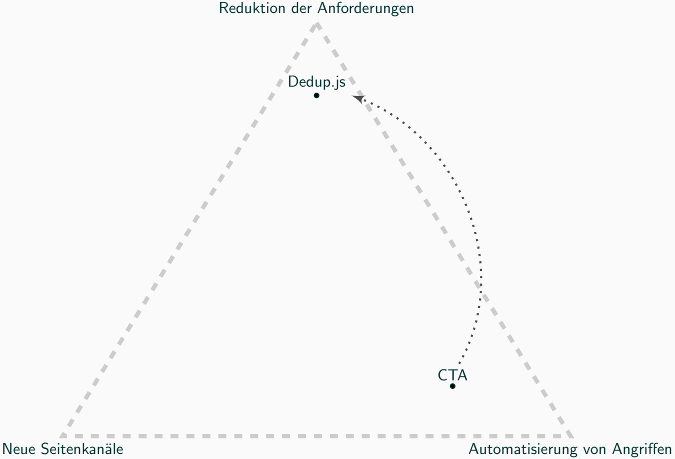
Das Verhältnis zwischen den Publikationen


Reduktion der Anforderungen
Neue Seitenkanäle
Automatisierung von Angriffen

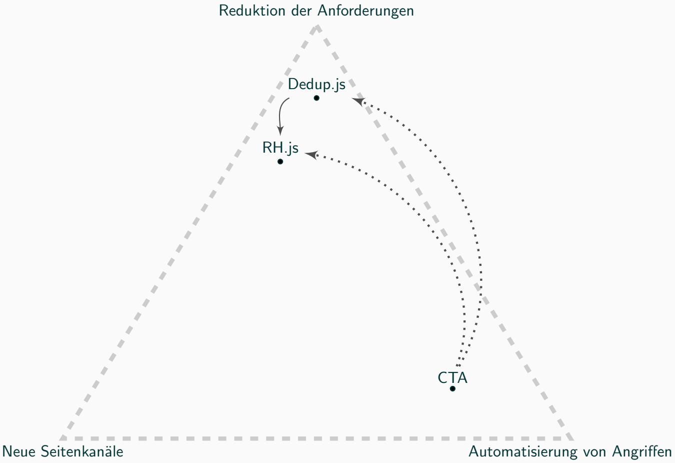
Das Verhältnis zwischen den Publikationen


Reduktion der Anforderungen
CTA
Neue Seitenkanäle
Automatisierung von Angriffen

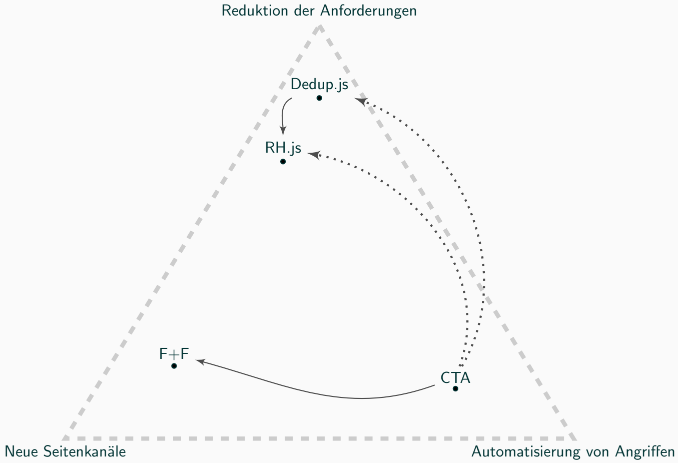
Das Verhältnis zwischen den Publikationen


Reduktion der Anforderungen
Dedup.js
CTA
Neue Seitenkanäle
Automatisierung von Angriffen

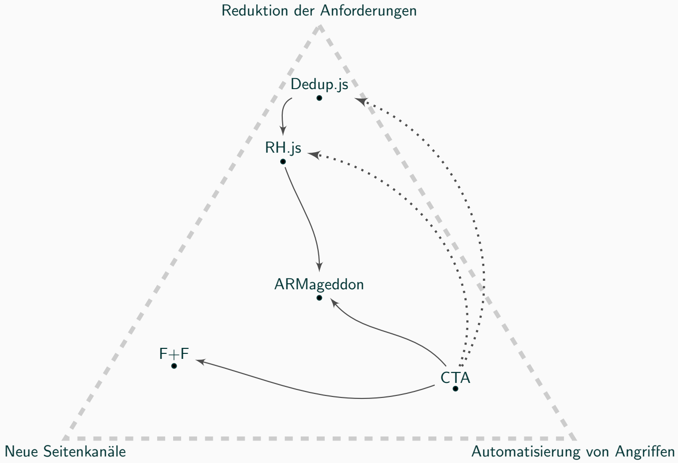
Das Verhältnis zwischen den Publikationen


Reduktion der Anforderungen
Dedup.js
RH.js
CTA
Neue Seitenkanäle
Automatisierung von Angriffen

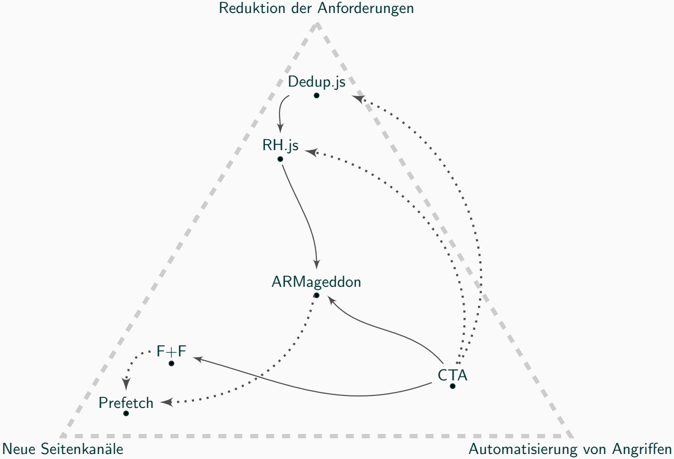
Das Verhältnis zwischen den Publikationen


Reduktion der Anforderungen
Dedup.js
RH.js
F+F
CTA
Neue Seitenkanäle
Automatisierung von Angriffen

Das Verhältnis zwischen den Publikationen


Reduktion der Anforderungen
Dedup.js
RH.js
ARMageddon
F+F
CTA
Neue Seitenkanäle
Automatisierung von Angriffen

Das Verhältnis zwischen den Publikationen


Reduktion der Anforderungen
Dedup.js
RH.js
ARMageddon
F+F
CTA
Prefetch
Neue Seitenkanäle
Automatisierung von Angriffen

Cache Template Angriffe

Cache Template Angriffe: Idee

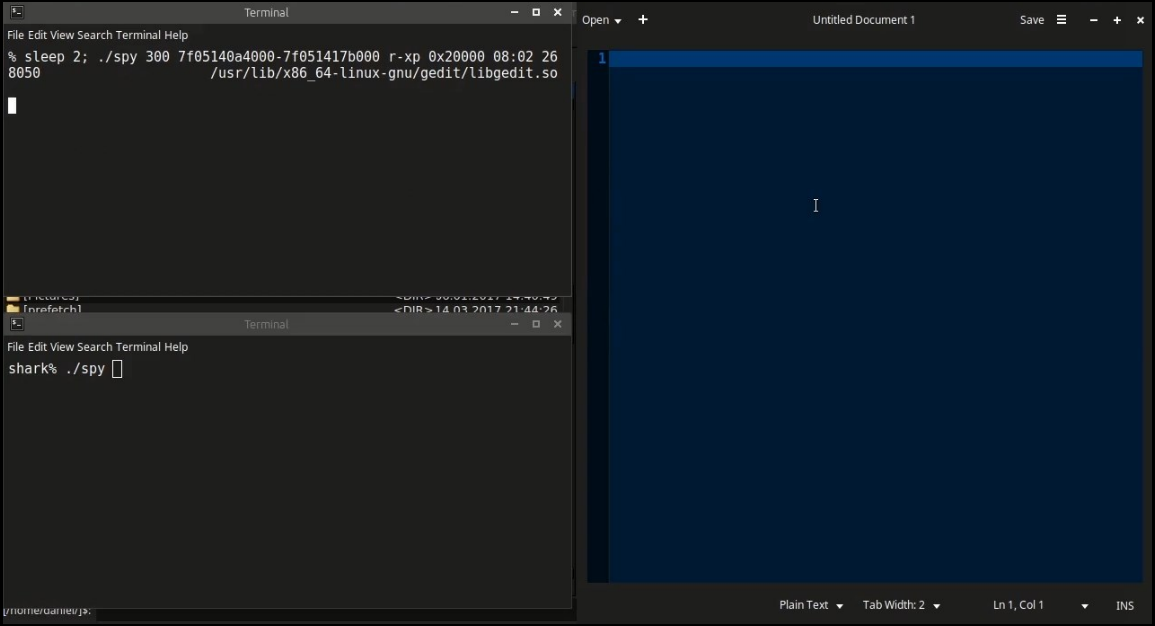
Cache Template Angriffe: Idee

1. Wie findet man Speicherstellen die abhängig von Geheimnissen zugegriffen werden?

Cache Template Angriffe: Idee

1. Wie findet man Speicherstellen die abhängig von Geheimnissen zugegriffen werden?

→ Cache Angriffe automatisieren

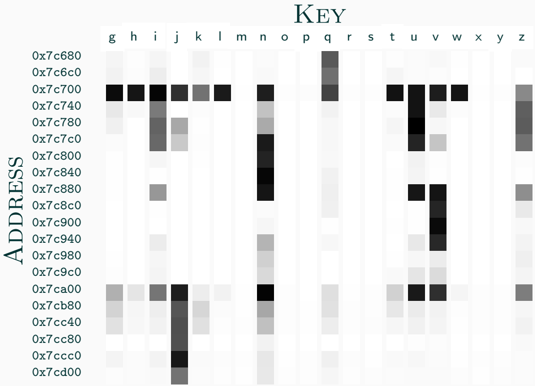

Terminal
File Edit View Search Terminal Help
[prefetch] <DIR> 14.03.2017 21:44:26
Terminal
File Edit View Search Terminal Help
Open
Untitled Document 1
Save
1
Plain Text
Tab Width: 2
Ln 1, Col 1
INS

```
% sleep 2; ./spy 300 7f05140a4000-7f051417b000 r-xp 0x20000 08:02 26
8050                     /usr/lib/x86_64-linux-gnu/gedit/libgedit.so
```

```
shark% ./spy
```

Cache Template Angriff Demo

Cache Template

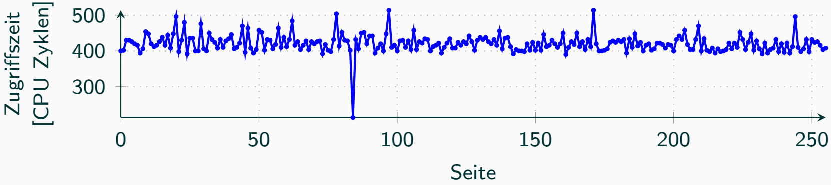


2018: Meltdown

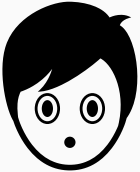
2018: Meltdown

- Einfaches Template über ein Array

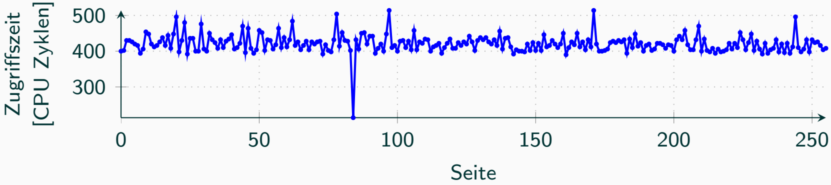
- Einfaches Template über ein Array



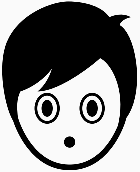
2018: Meltdown



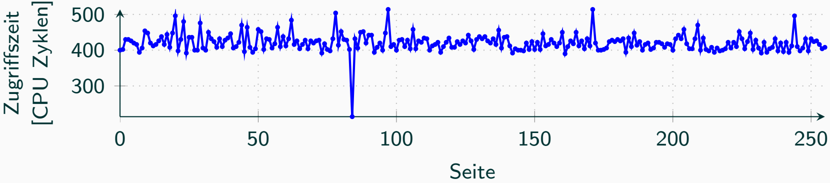
- Einfaches Template über ein Array



2018: Meltdown

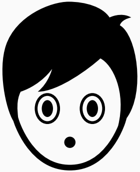


- Einfaches Template über ein Array

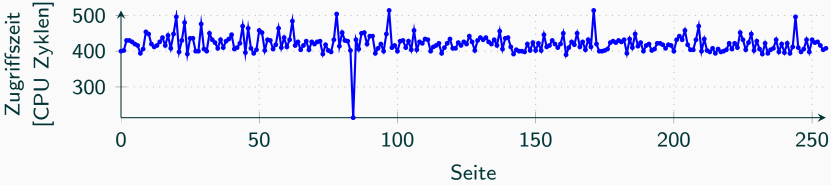


- Geheimer Wert aus dem Kernel direkt ablesbar

2018: Meltdown



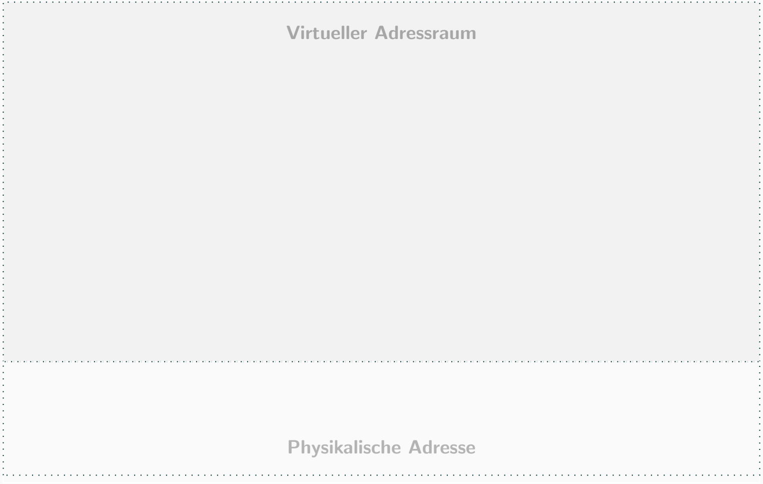
- Einfaches Template über ein Array



- Geheimer Wert aus dem Kernel direkt ablesbar
- Sämtliche Daten im System betroffen

Page Deduplication Angriffe in JavaScript

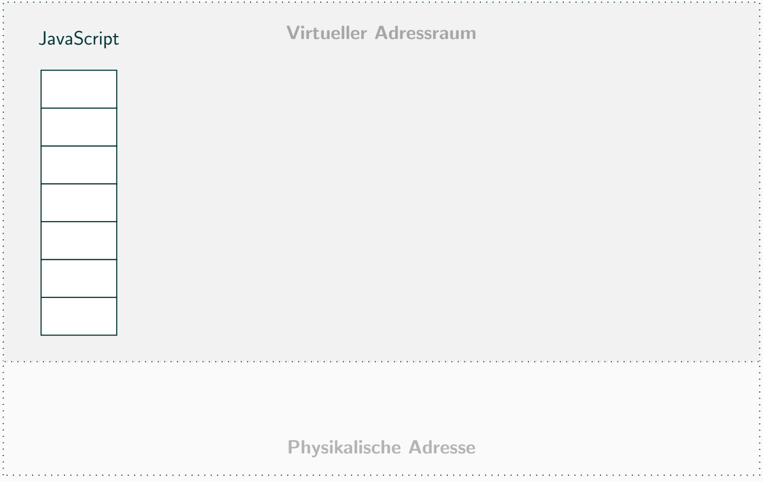
Page Deduplication Angriffe



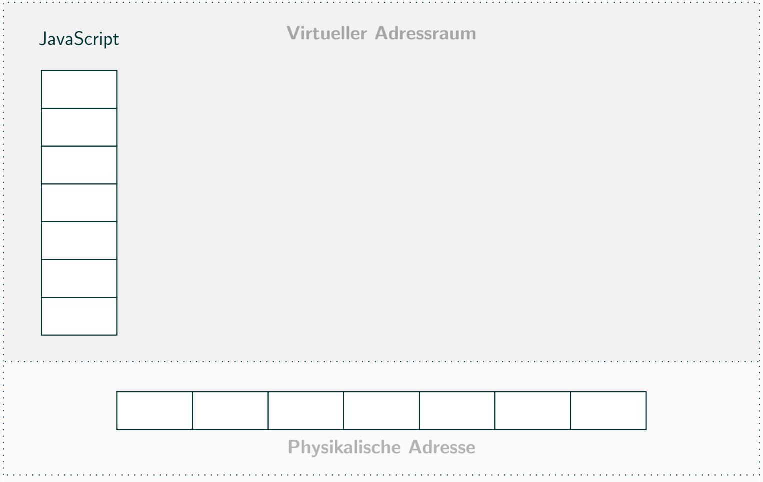
Virtueller Adressraum

Physikalische Adresse

Page Deduplication Angriffe


JavaScript
Virtueller Adressraum
Physikalische Adresse

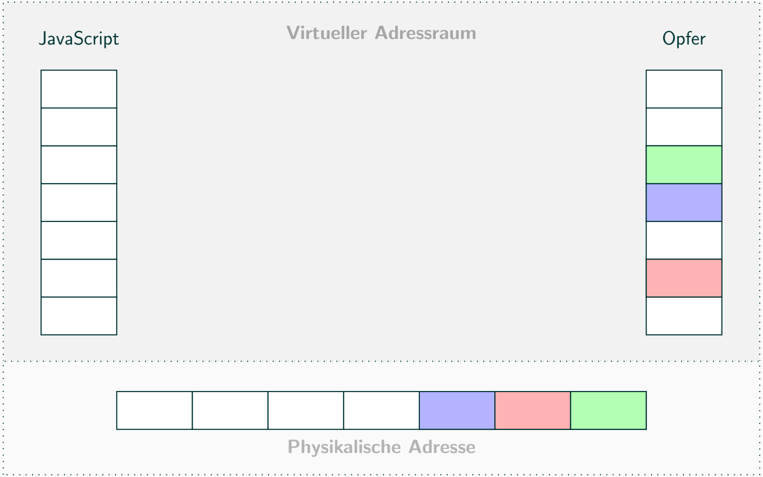
Page Deduplication Angriffe


JavaScript
Virtueller Adressraum
Physikalische Adresse

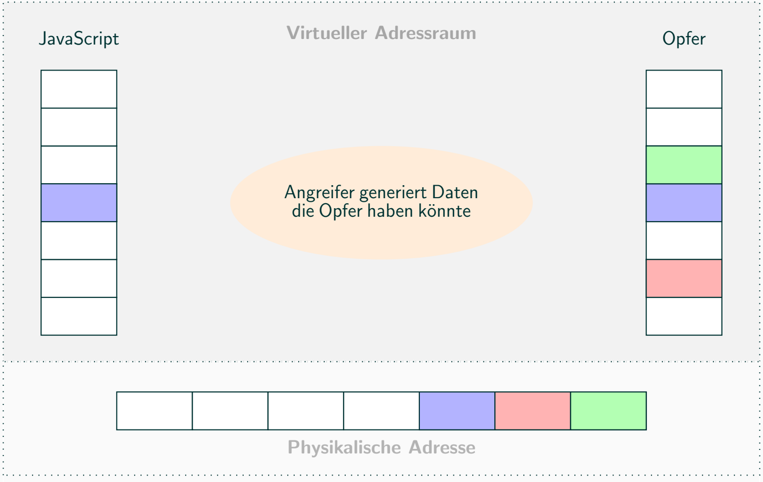
Page Deduplication Angriffe

JavaScript
Virtueller Adressraum
Opfer
Physikalische Adresse

Page Deduplication Angriffe

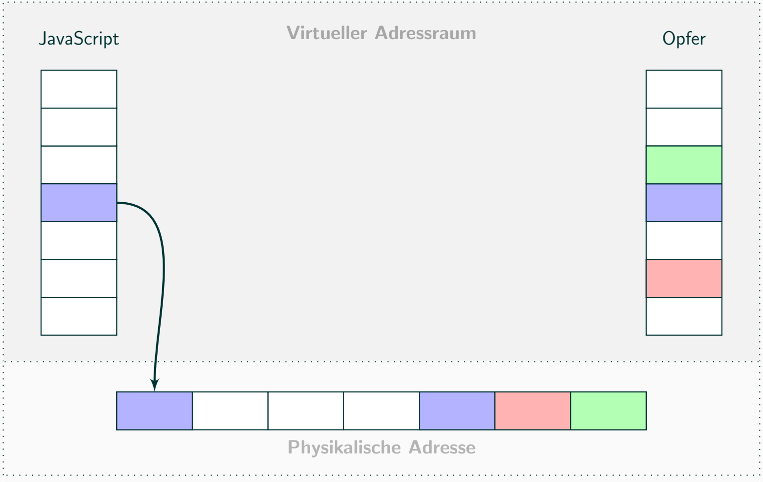

Virtueller Adressraum
JavaScript
Opfer
Physikalische Adresse

Page Deduplication Angriffe

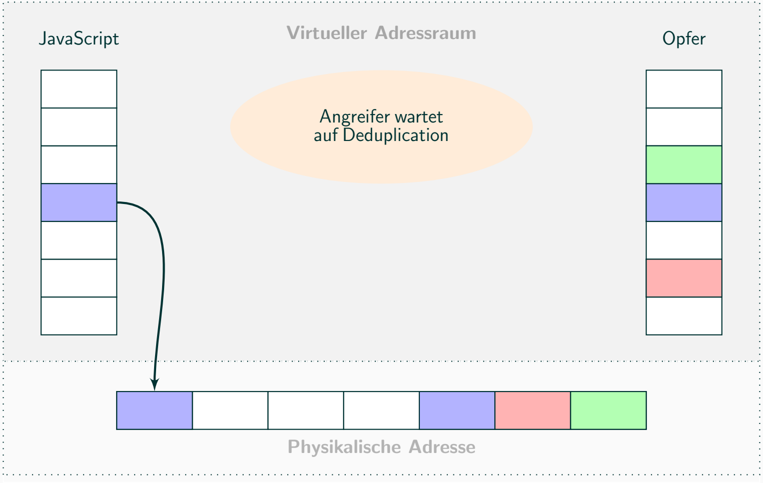

Virtueller Adressraum
JavaScript
Opfer
Physikalische Adresse

Angreifer generiert Daten
die Opfer haben könnte

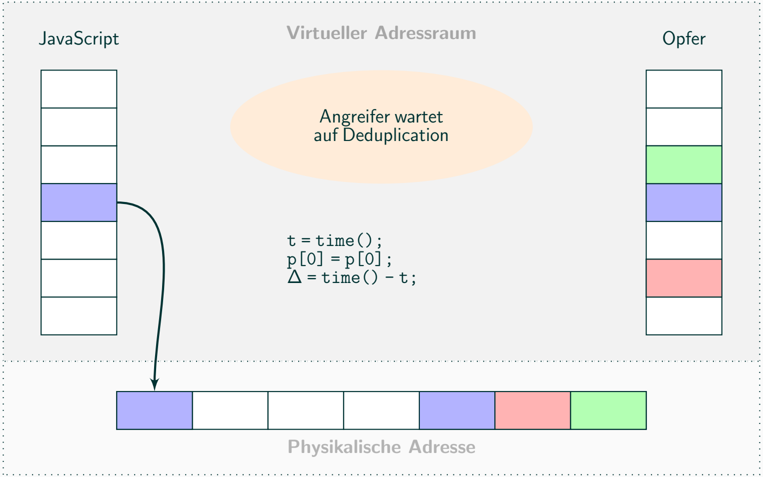
Page Deduplication Angriffe


JavaScript
Virtueller Adressraum
Opfer
Physikalische Adresse

Page Deduplication Angriffe

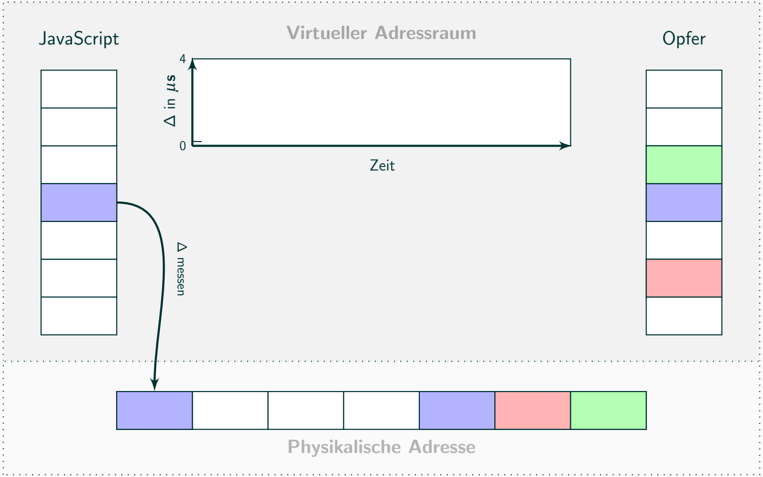

JavaScript
Virtueller Adressraum
Opfer
Angreifer wartet auf Deduplication
Physikalische Adresse

Page Deduplication Angriffe

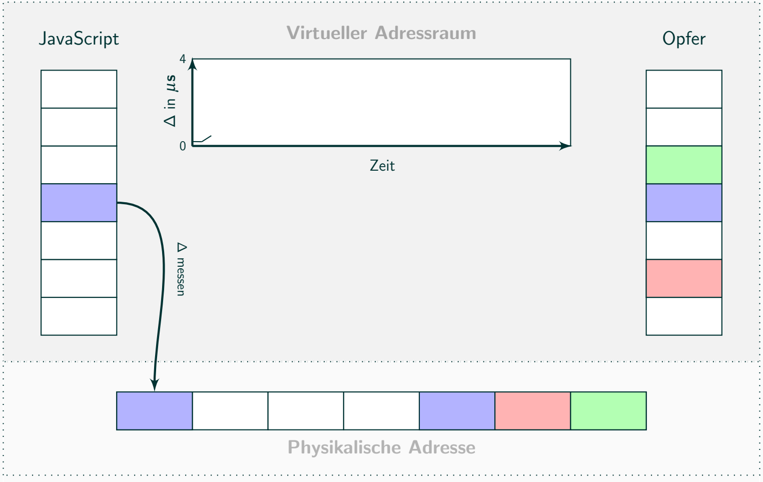

JavaScript
Virtueller Adressraum
Opfer
Angreifer wartet
auf Deduplication
Physikalische Adresse

```
t = time();
p[0] = p[0];
Δ = time() - t;
```

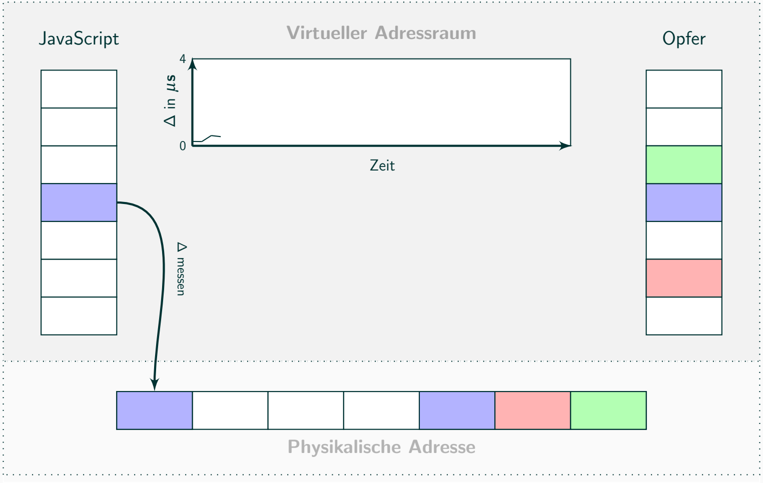
Page Deduplication Angriffe


Virtueller Adressraum
JavaScript
Opfer
4
Δ in μs
0
Zeit
Δ messen
Physikalische Adresse

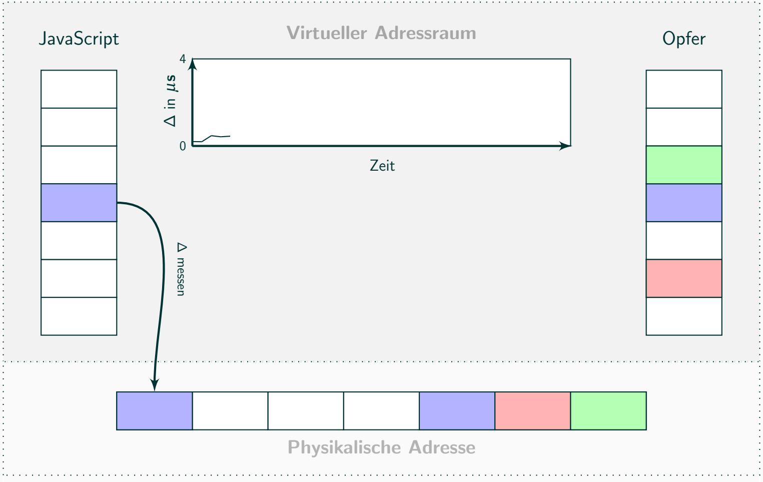
Page Deduplication Angriffe


Virtueller Adressraum
JavaScript
Opfer
4
0
Δ in μs
Zeit
Δ messen
Physikalische Adresse

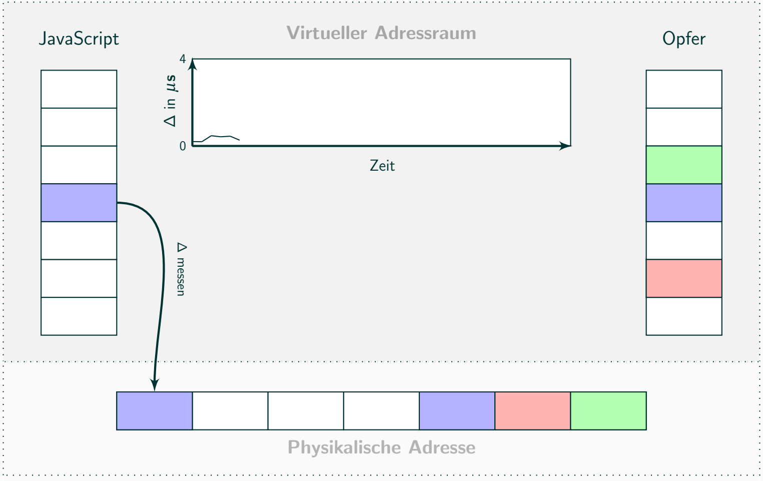
Page Deduplication Angriffe


JavaScript
Virtueller Adressraum
Opfer
4
0
Δ in µs
Zeit
Δ messen
Physikalische Adresse

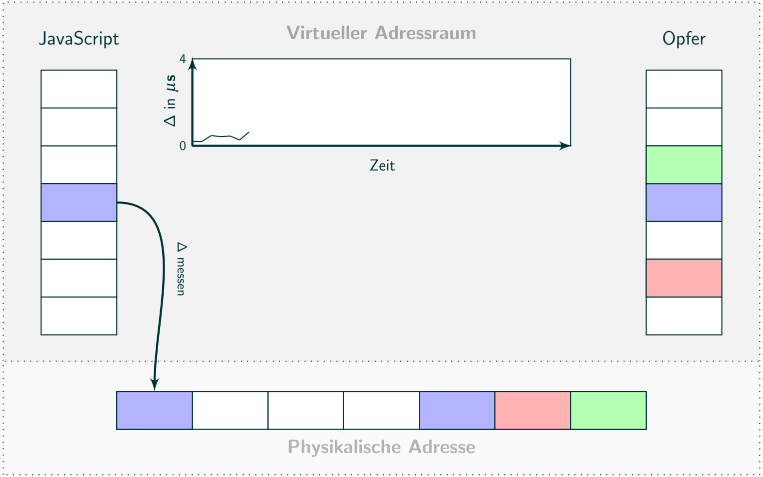
Page Deduplication Angriffe


JavaScript
Virtueller Adressraum
Opfer
4
0
Δ in µs
Zeit
Δ messen
Physikalische Adresse

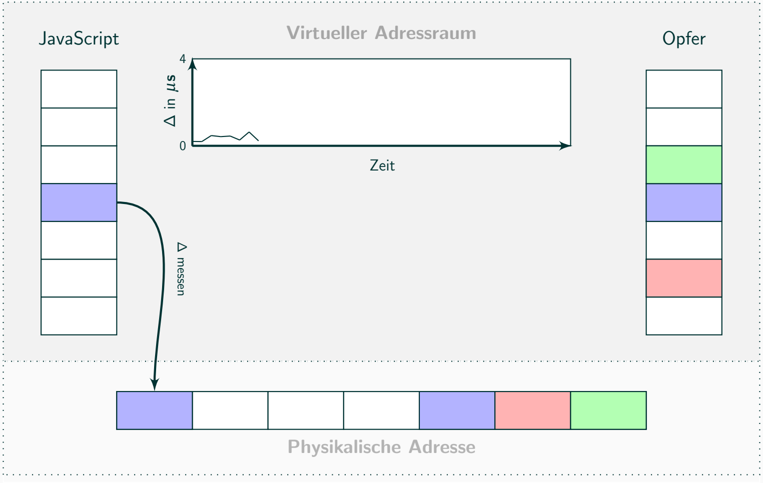
Page Deduplication Angriffe


JavaScript
Virtueller Adressraum
Opfer
4
Δ in μs
0
Zeit
Δ messen
Physikalische Adresse

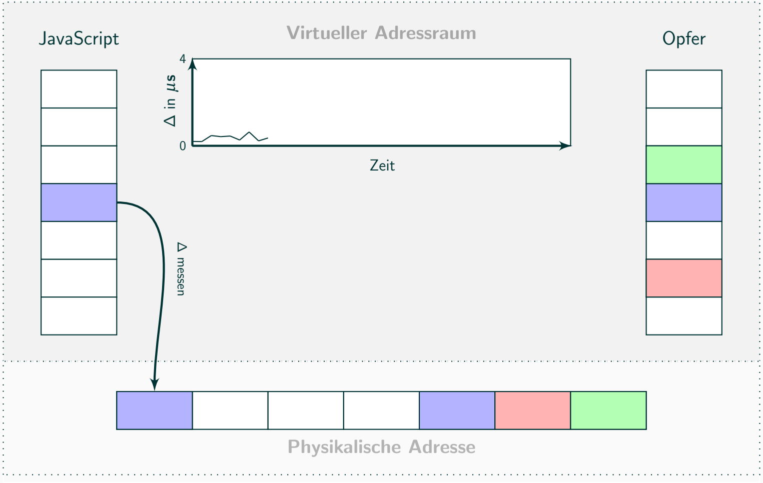
Page Deduplication Angriffe


JavaScript
Virtueller Adressraum
Opfer
4
0
Δ in µs
Zeit
Δ messen
Physikalische Adresse

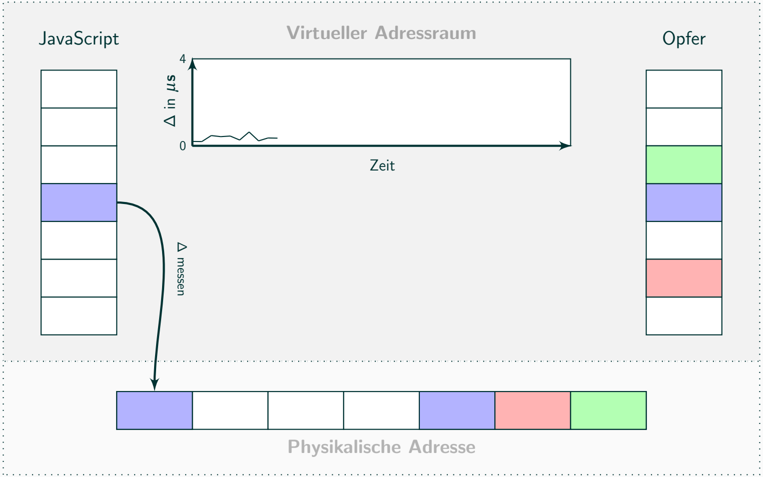
Page Deduplication Angriffe


JavaScript
Virtueller Adressraum
Opfer
4
Δ in μs
0
Zeit
Δ messen
Physikalische Adresse

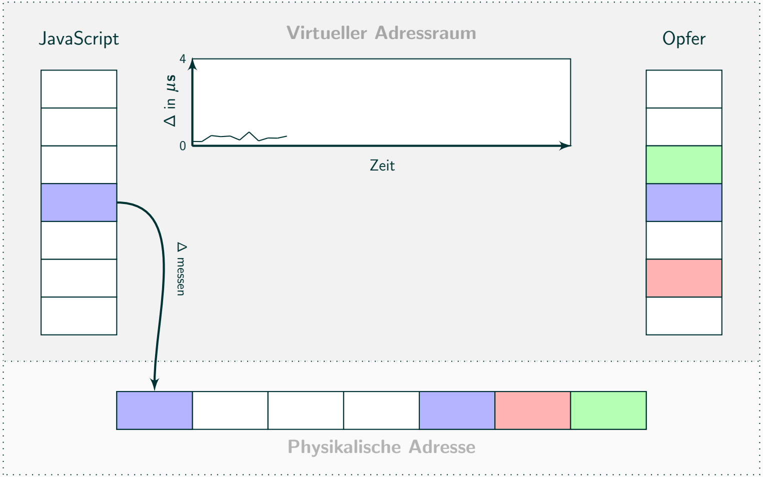
Page Deduplication Angriffe


JavaScript
Virtueller Adressraum
Opfer
4
0
Δ in µs
Zeit
Δ messen
Physikalische Adresse

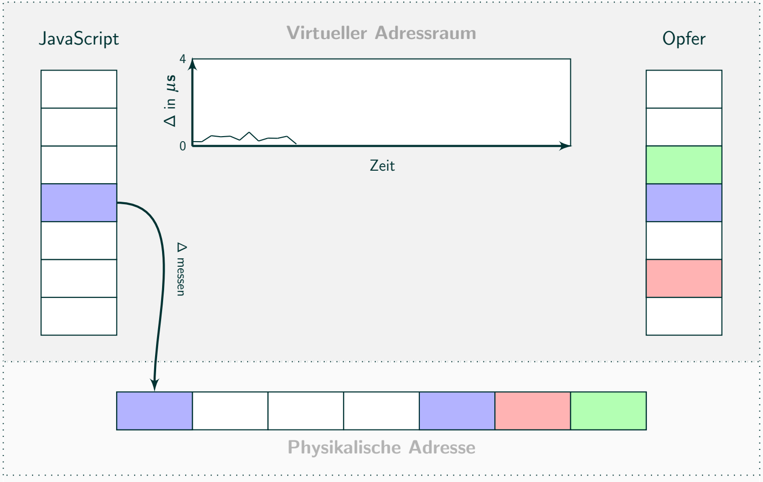
Page Deduplication Angriffe


Virtueller Adressraum
JavaScript
Opfer
4
0
Δ in μs
Zeit
Δ messen
Physikalische Adresse

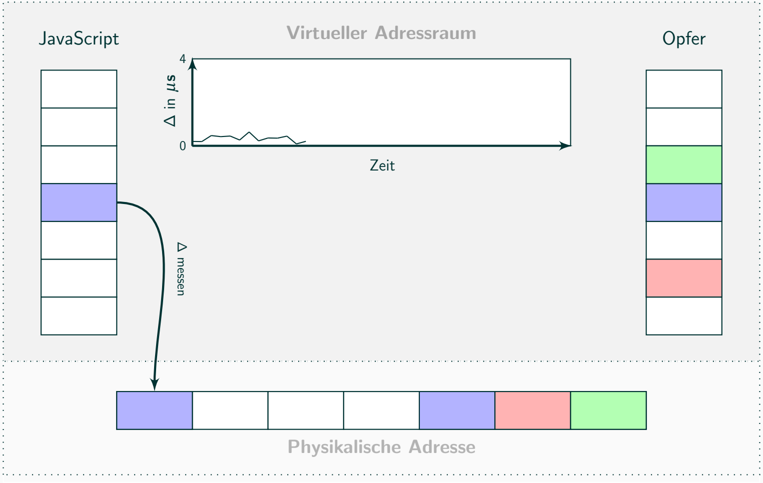
Page Deduplication Angriffe


JavaScript
Virtueller Adressraum
Opfer
4
Δ in μs
0
Zeit
Δ messen
Physikalische Adresse

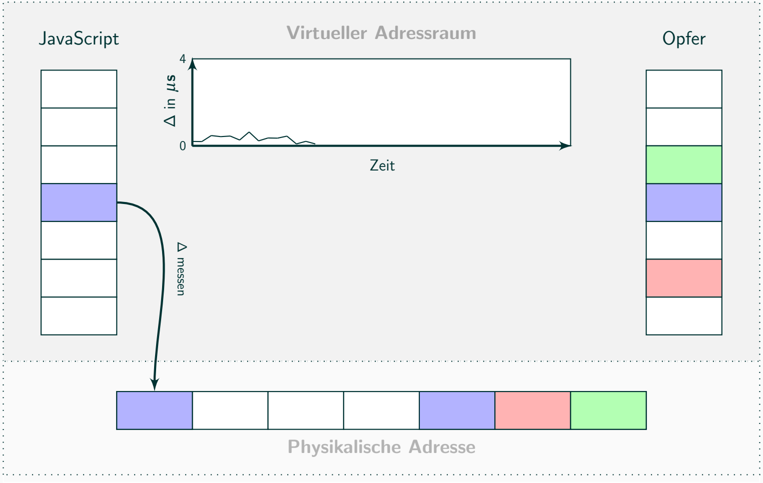
Page Deduplication Angriffe


JavaScript
Virtueller Adressraum
Opfer
4
0
Δ in μs
Zeit
Δ messen
Physikalische Adresse

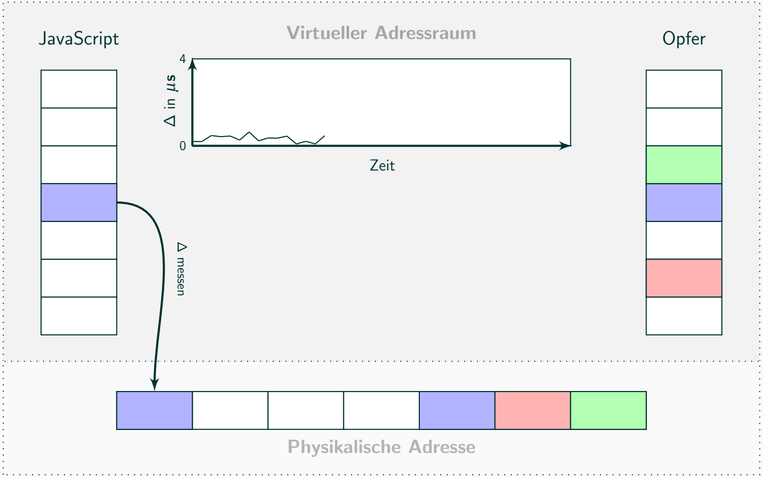
Page Deduplication Angriffe


Virtueller Adressraum
JavaScript
Opfer
4
0
Δ in μs
Zeit
Δ messen
Physikalische Adresse

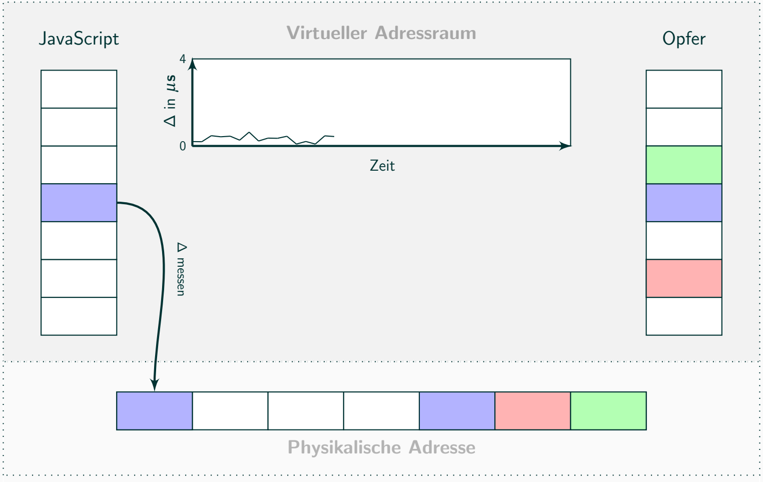
Page Deduplication Angriffe


JavaScript
Virtueller Adressraum
Opfer
4
Δ in μs
0
Zeit
Δ messen
Physikalische Adresse

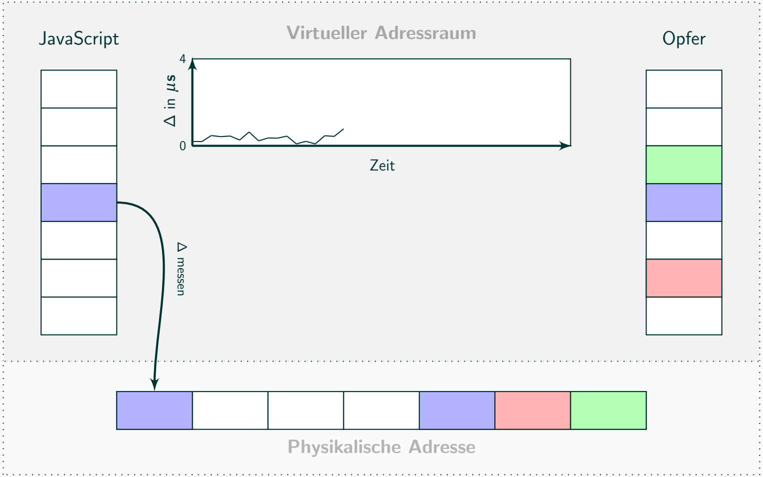
Page Deduplication Angriffe


JavaScript
Virtueller Adressraum
Opfer
4
0
Δ in μs
Zeit
Δ messen
Physikalische Adresse

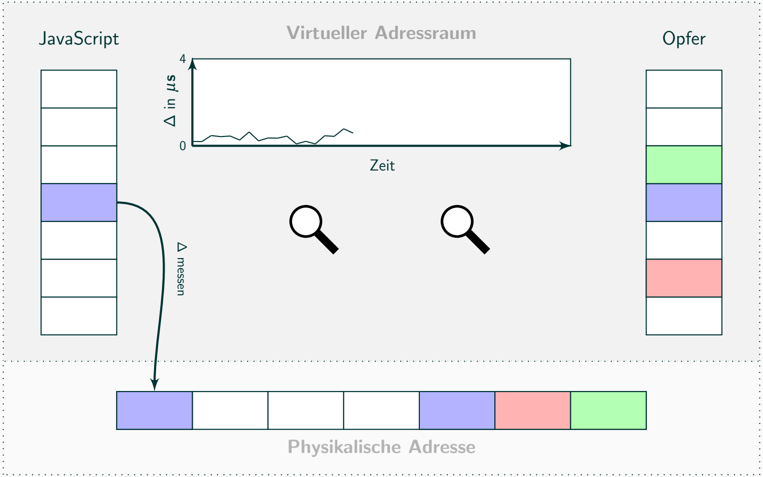
Page Deduplication Angriffe


JavaScript
Virtueller Adressraum
Opfer
4
0
Δ in μs
Zeit
Δ messen
Physikalische Adresse

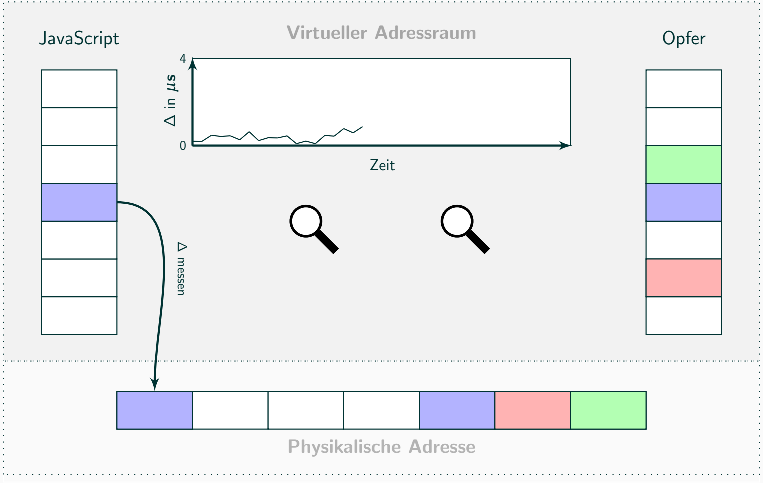
Page Deduplication Angriffe


JavaScript
Virtueller Adressraum
Opfer
4
0
Δ in μs
Zeit
Δ messen
Physikalische Adresse

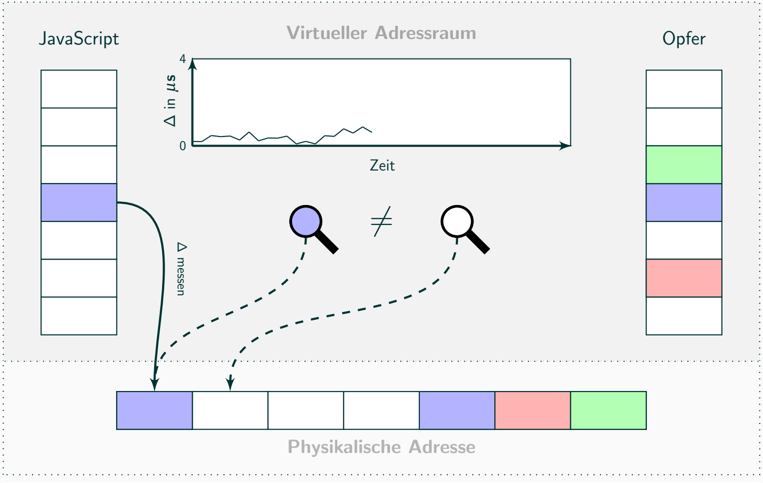
Page Deduplication Angriffe


JavaScript
Virtueller Adressraum
Opfer
4
0
Δ in μs
Zeit
Δ messen
Physikalische Adresse

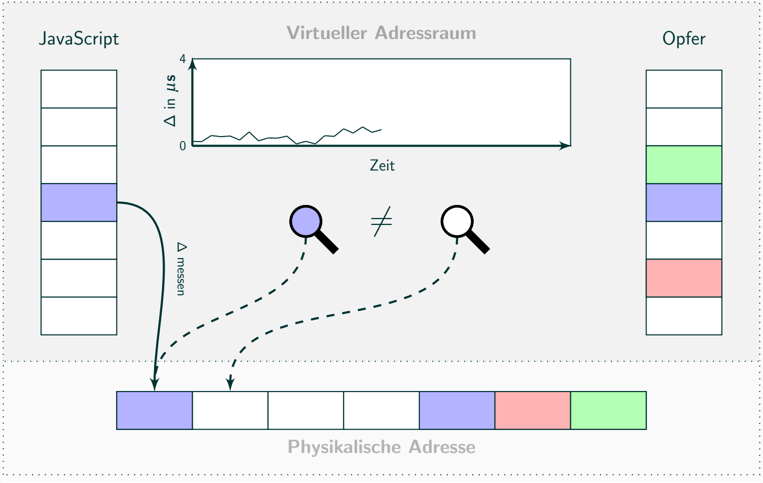
Page Deduplication Angriffe


Virtueller Adressraum
JavaScript
Opfer
4
0
Δ in µs
Zeit
Δ messen
Physikalische Adresse

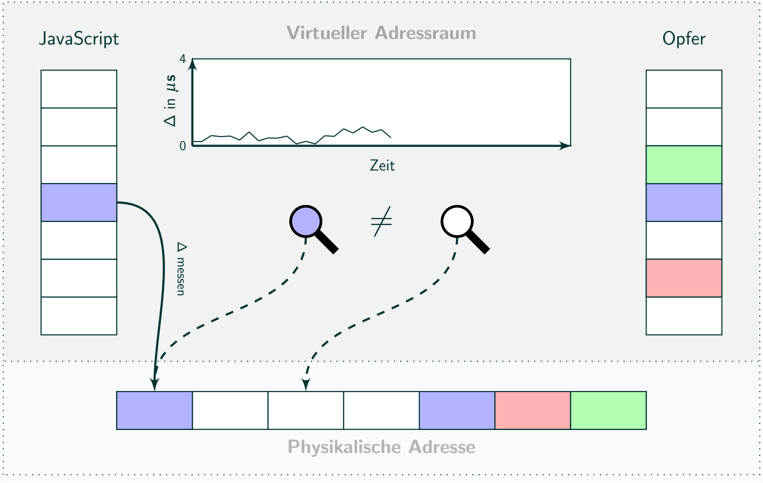
Page Deduplication Angriffe


Virtueller Adressraum
JavaScript
Opfer
4
Δ in μs
0
Zeit
Δ messen
≠
Physikalische Adresse

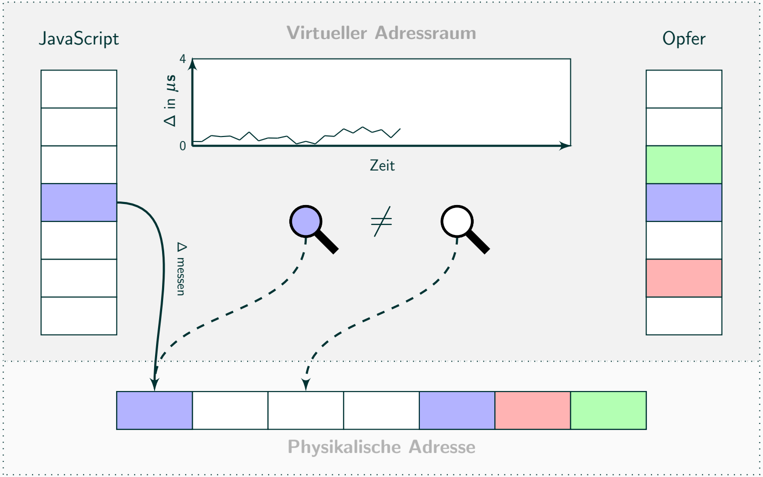
Page Deduplication Angriffe


JavaScript
Virtueller Adressraum
Opfer
4
Δ in μs
0
Zeit
Δ messen
≠
Physikalische Adresse

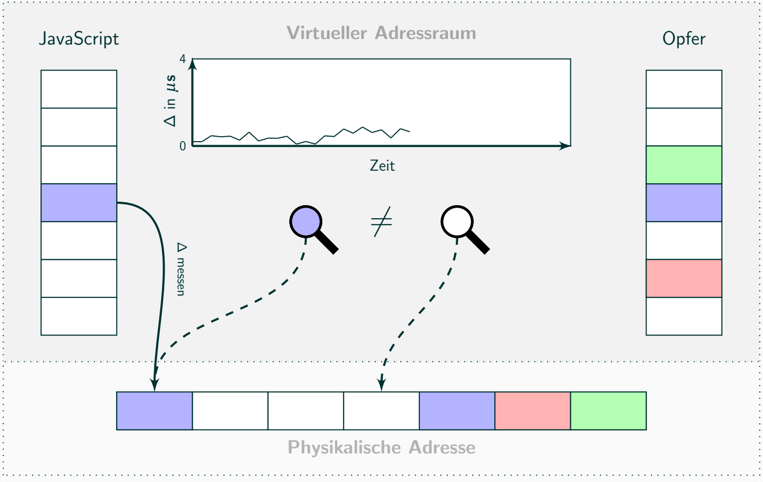
Page Deduplication Angriffe


JavaScript
Virtueller Adressraum
Opfer
4
Δ in μs
0
Zeit
Δ messen
≠
Physikalische Adresse

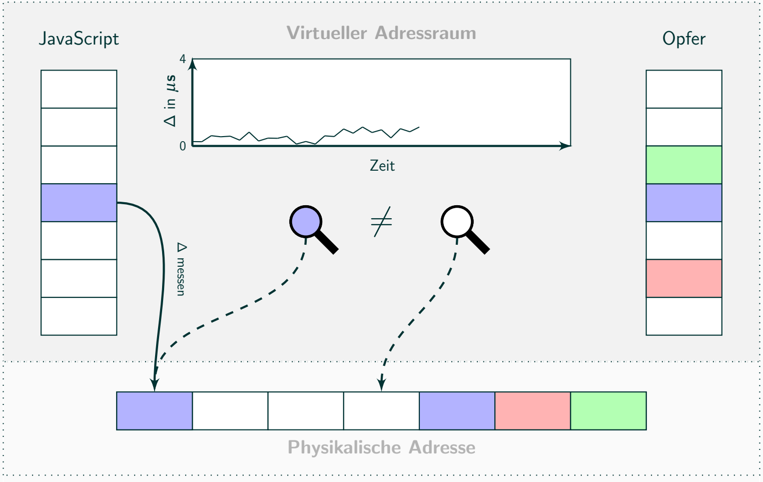
Page Deduplication Angriffe


JavaScript
Virtueller Adressraum
Opfer
4
0
Δ in µs
Zeit
≠
Δ messen
Physikalische Adresse

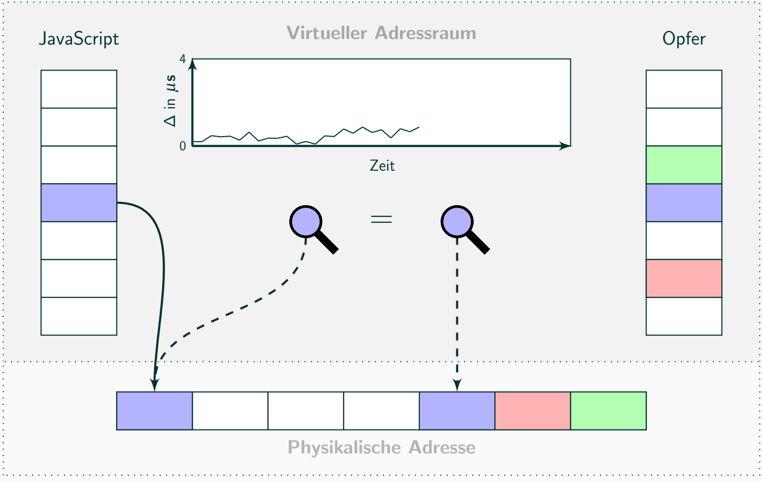
Page Deduplication Angriffe


JavaScript
Virtueller Adressraum
Opfer
4
0
Δ in µs
Zeit
Δ messen
≠
Physikalische Adresse

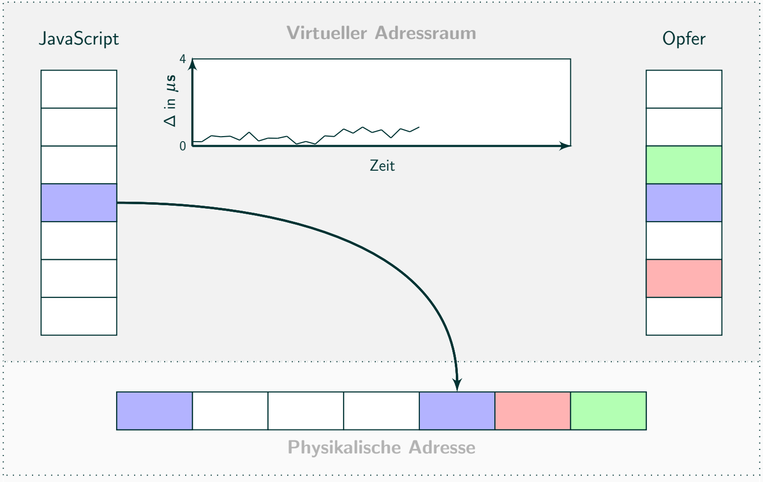
Page Deduplication Angriffe


JavaScript
Virtueller Adressraum
Opfer
4
0
Δ in µs
Zeit
Δ messen
≠
Physikalische Adresse

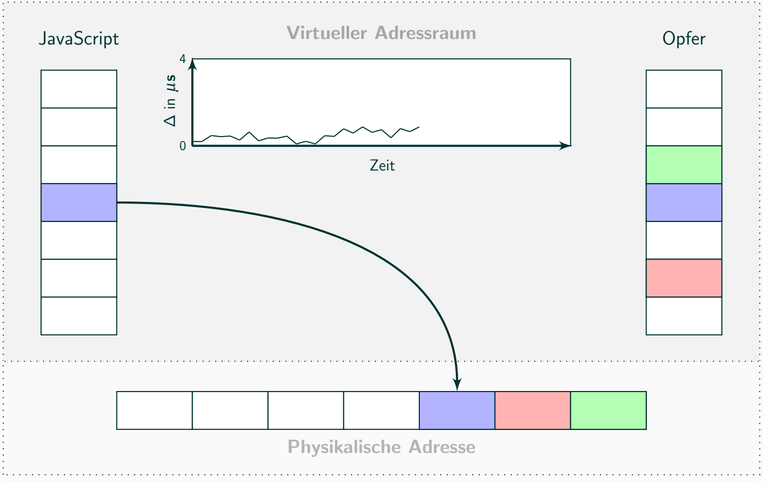
Page Deduplication Angriffe


JavaScript
Virtueller Adressraum
Opfer
4
0
Δ in μs
Zeit
=
Physikalische Adresse

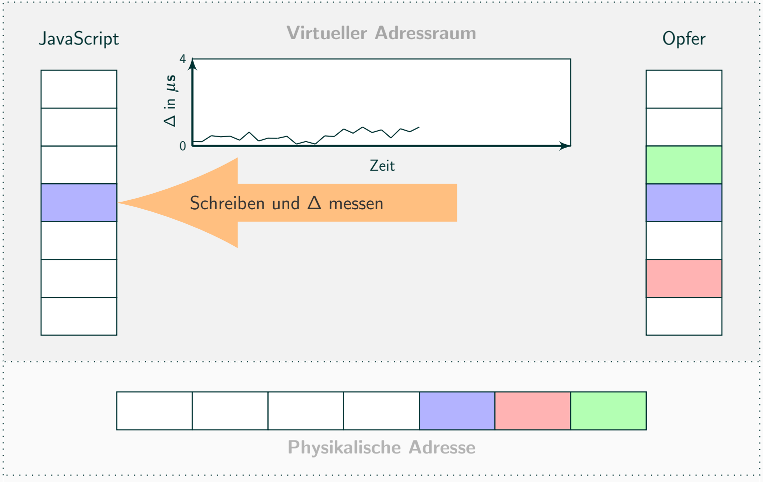
Page Deduplication Angriffe


JavaScript
Virtueller Adressraum
Opfer
4
0
Δ in µs
Zeit
Physikalische Adresse

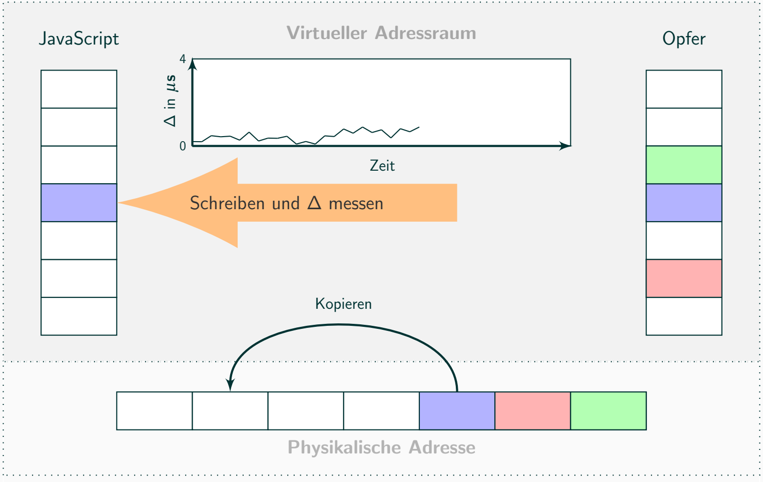
Page Deduplication Angriffe


JavaScript
Virtueller Adressraum
Opfer
4
Δ in µs
0
Zeit
Physikalische Adresse

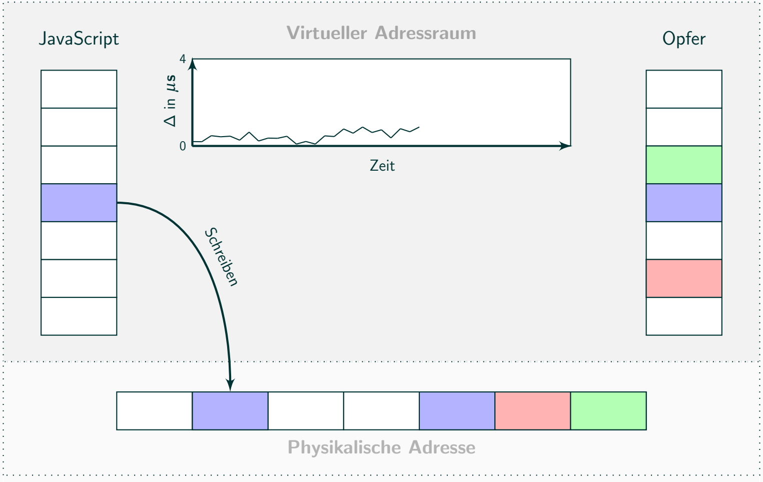
Page Deduplication Angriffe


JavaScript
Virtueller Adressraum
Opfer
4
Δ in μs
0
Zeit
Schreiben und Δ messen
Physikalische Adresse

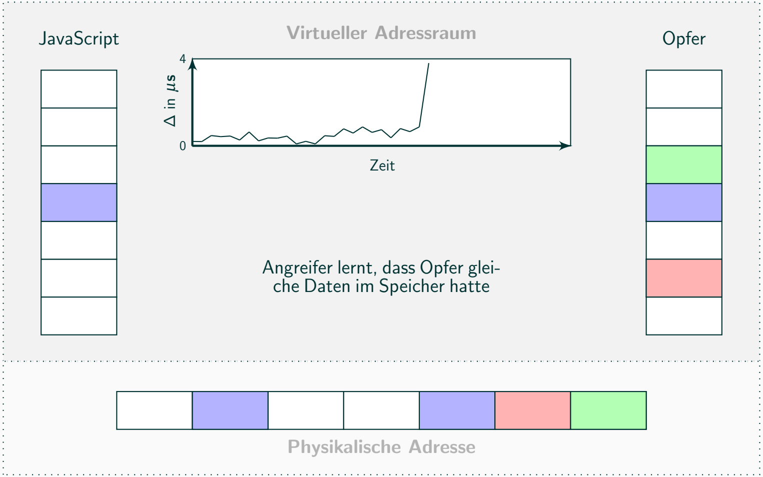
Page Deduplication Angriffe


Virtueller Adressraum
JavaScript
Opfer
4
0
Δ in µs
Zeit
Schreiben und Δ messen
Kopieren
Physikalische Adresse

Page Deduplication Angriffe

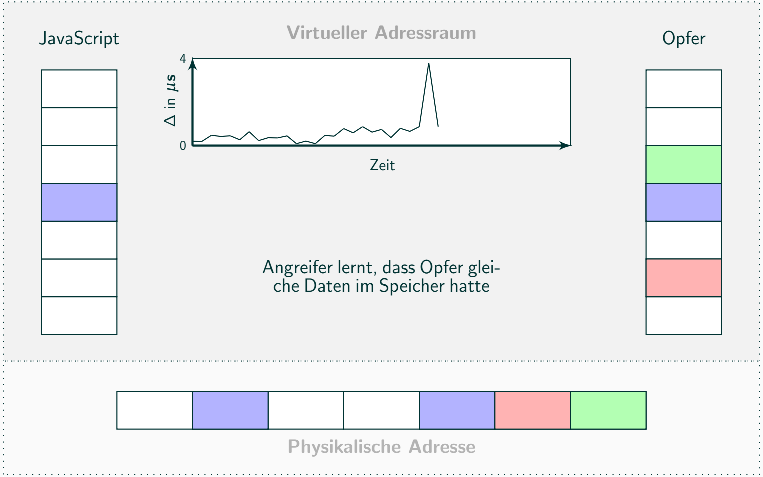

JavaScript
Virtueller Adressraum
Opfer
4
Δ in μs
0
Zeit
Schreiben
Physikalische Adresse

Page Deduplication Angriffe


JavaScript
Virtueller Adressraum
Opfer
4
0
Δ in μs
Zeit
Physikalische Adresse

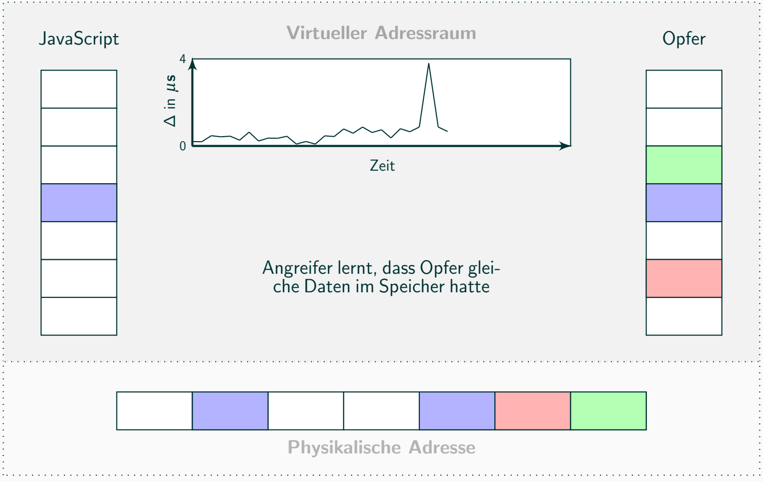
Angreifer lernt, dass Opfer gleiche Daten im Speicher hatte

Page Deduplication Angriffe


JavaScript
Virtueller Adressraum
Opfer
4
0
Δ in µs
Zeit
Physikalische Adresse

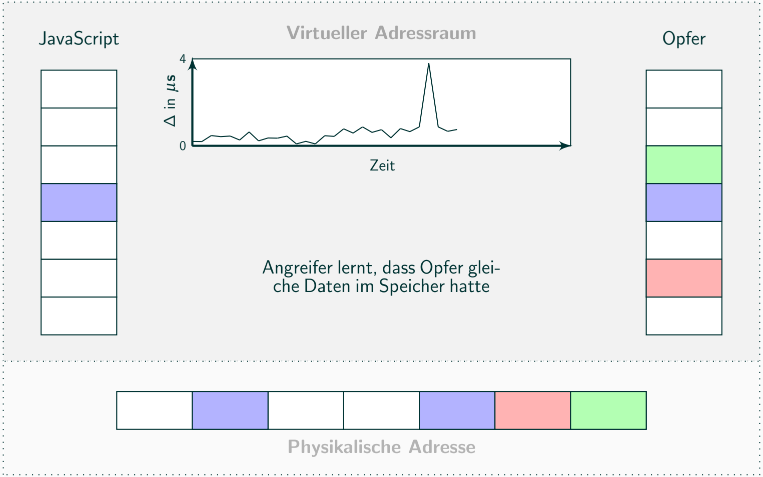
Angreifer lernt, dass Opfer gleiche Daten im Speicher hatte

Page Deduplication Angriffe


JavaScript
Virtueller Adressraum
Opfer
4
0
Δ in µs
Zeit
Physikalische Adresse

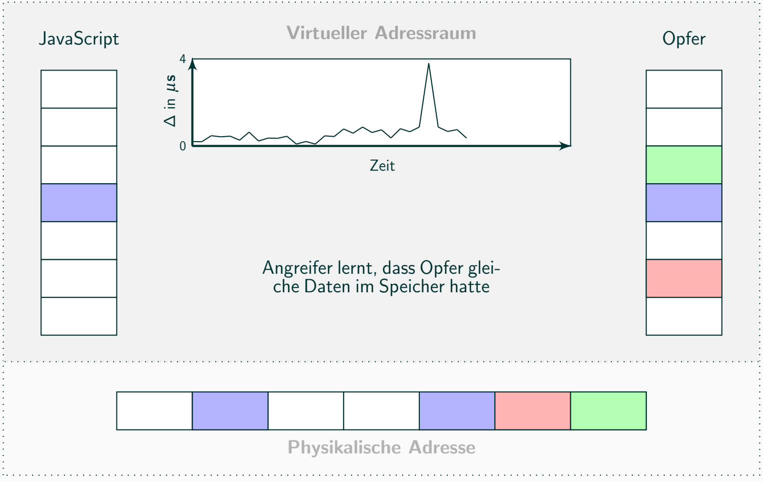
Angreifer lernt, dass Opfer gleiche Daten im Speicher hatte

Page Deduplication Angriffe


JavaScript
Virtueller Adressraum
Opfer
4
0
Δ in μs
Zeit
Physikalische Adresse

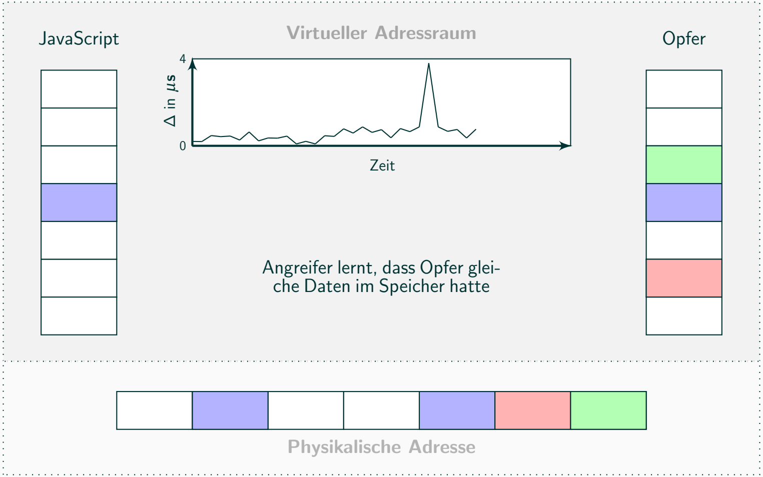
Angreifer lernt, dass Opfer gleiche Daten im Speicher hatte

Page Deduplication Angriffe


JavaScript
Virtueller Adressraum
Opfer
4
0
Δ in μs
Zeit
Physikalische Adresse

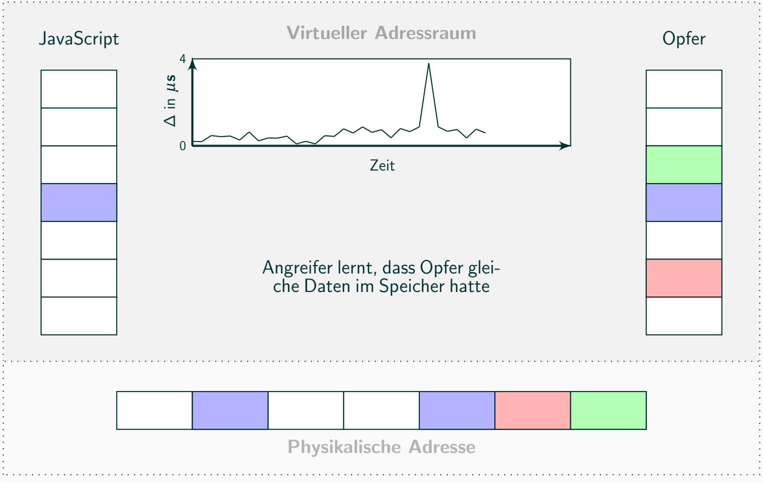
Angreifer lernt, dass Opfer gleiche Daten im Speicher hatte

Page Deduplication Angriffe


JavaScript
Virtueller Adressraum
Opfer
4
0
Δ in μs
Zeit
Physikalische Adresse

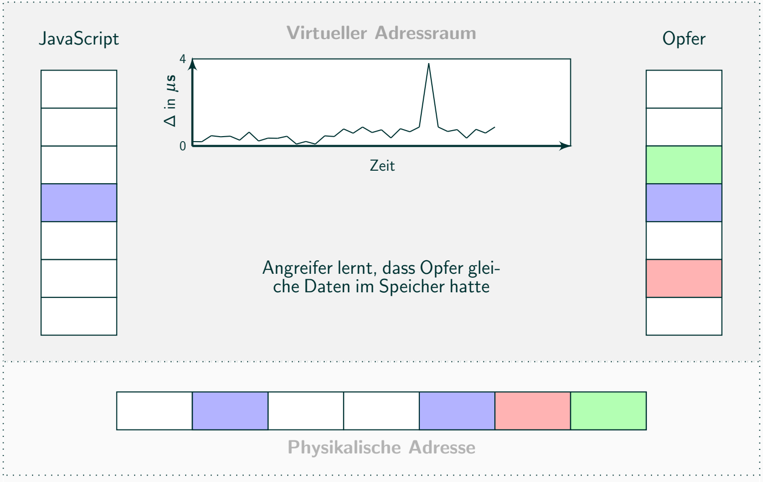
Angreifer lernt, dass Opfer gleiche Daten im Speicher hatte

Page Deduplication Angriffe


JavaScript
Virtueller Adressraum
Opfer
4
0
Δ in μs
Zeit
Physikalische Adresse

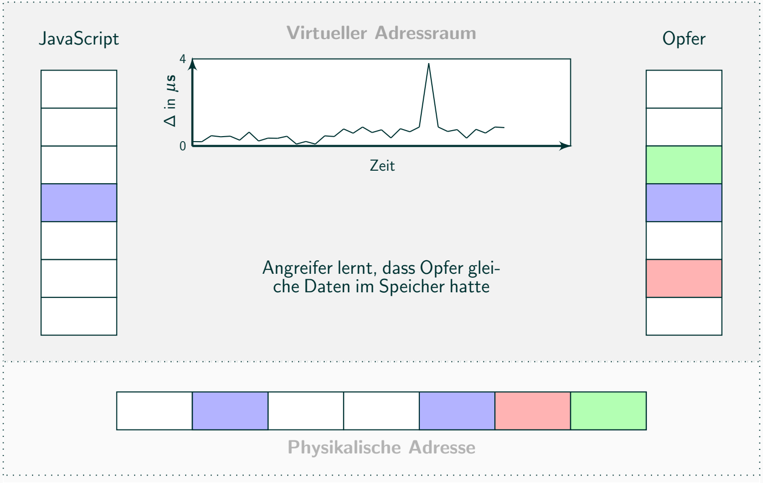
Angreifer lernt, dass Opfer gleiche Daten im Speicher hatte

Page Deduplication Angriffe


JavaScript
Virtueller Adressraum
Opfer
4
Δ in µs
0
Zeit
Physikalische Adresse

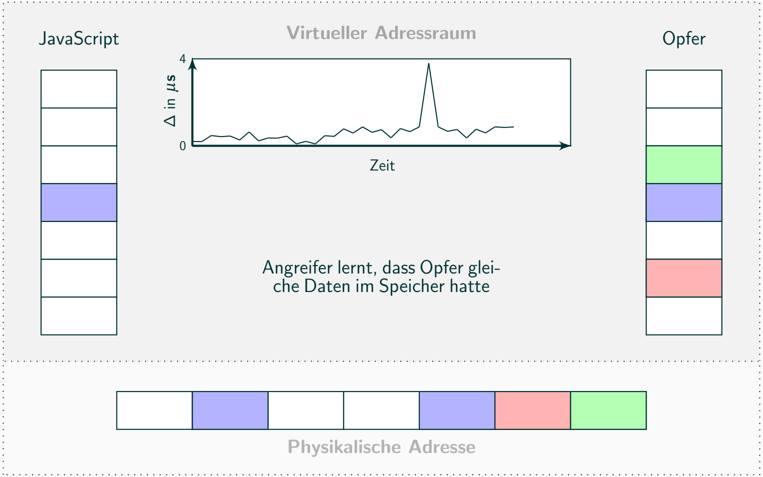
Angreifer lernt, dass Opfer gleiche Daten im Speicher hatte

Page Deduplication Angriffe


JavaScript
Virtueller Adressraum
Opfer
4
0
Δ in μs
Zeit
Physikalische Adresse

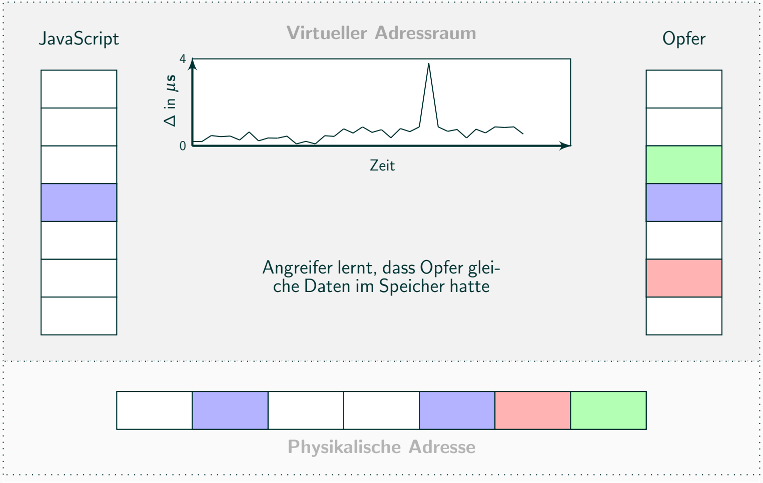
Angreifer lernt, dass Opfer gleiche Daten im Speicher hatte

Page Deduplication Angriffe


JavaScript
Virtueller Adressraum
Opfer
4
0
Δ in μs
Zeit
Physikalische Adresse

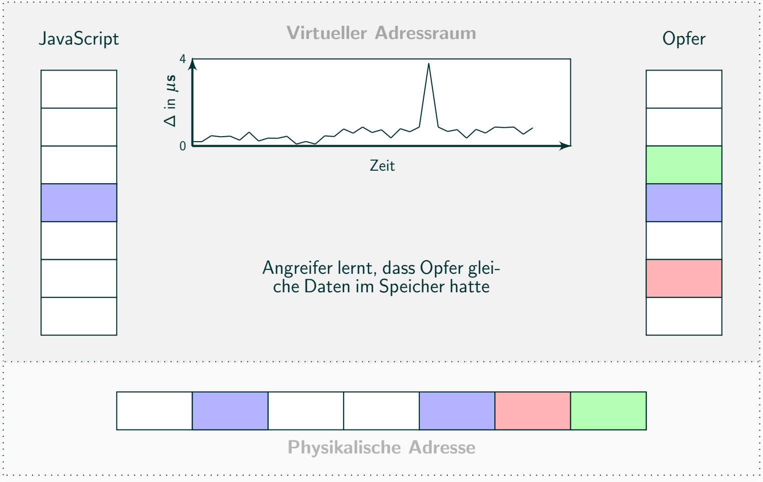
Angreifer lernt, dass Opfer gleiche Daten im Speicher hatte

Page Deduplication Angriffe


JavaScript
Virtueller Adressraum
Opfer
4
0
Δ in µs
Zeit
Physikalische Adresse

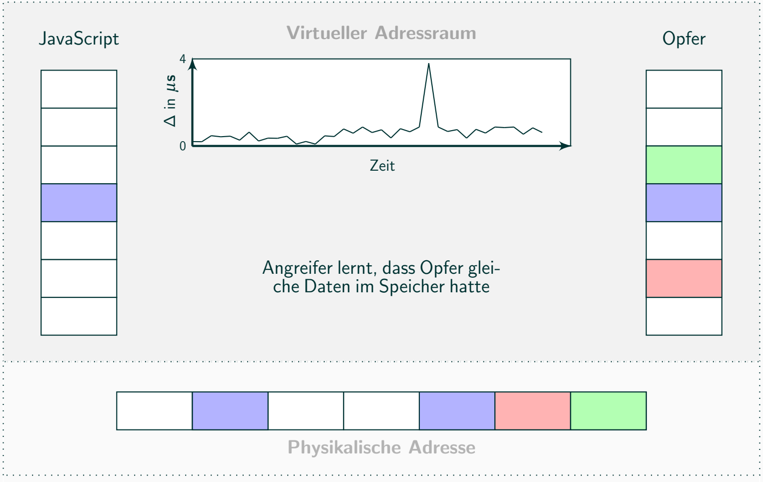
Angreifer lernt, dass Opfer gleiche Daten im Speicher hatte

Page Deduplication Angriffe


JavaScript
Virtueller Adressraum
Opfer
4
0
Δ in μs
Zeit
Physikalische Adresse

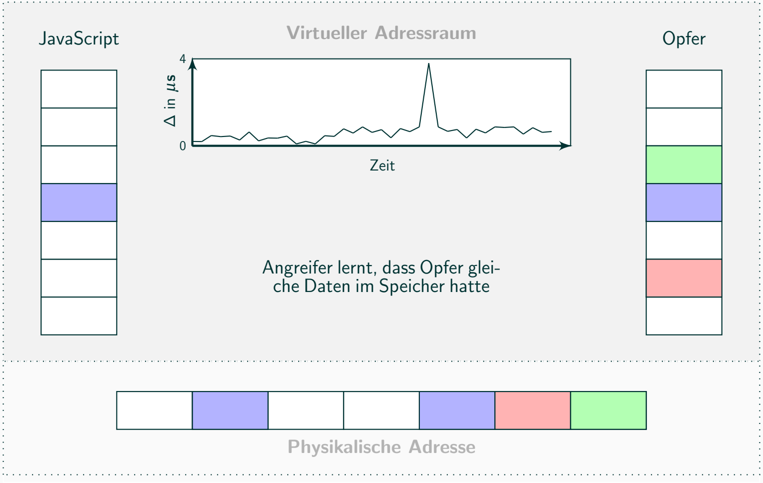
Angreifer lernt, dass Opfer gleiche Daten im Speicher hatte

Page Deduplication Angriffe


JavaScript
Virtueller Adressraum
Opfer
4
0
Δ in μs
Zeit
Physikalische Adresse

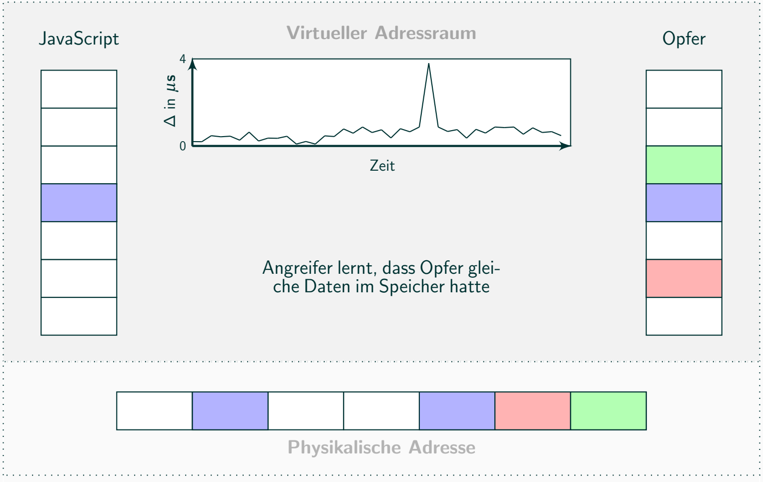
Angreifer lernt, dass Opfer gleiche Daten im Speicher hatte

Page Deduplication Angriffe


Virtueller Adressraum
JavaScript
Opfer
4
0
Δ in μs
Zeit
Physikalische Adresse

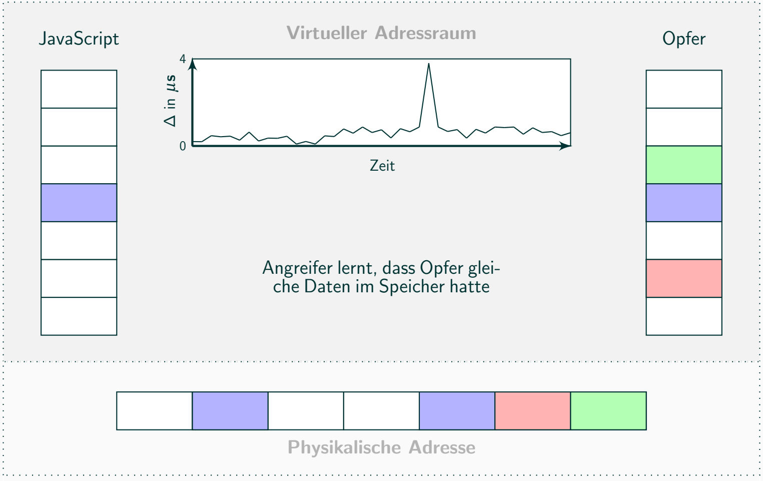
Angreifer lernt, dass Opfer gleiche Daten im Speicher hatte

Page Deduplication Angriffe


JavaScript
Virtueller Adressraum
Opfer
4
0
Δ in µs
Zeit
Physikalische Adresse

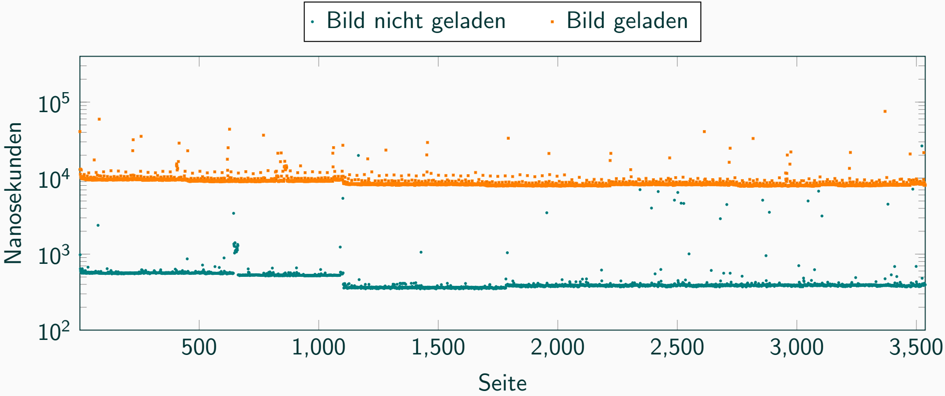
Angreifer lernt, dass Opfer gleiche Daten im Speicher hatte

Page Deduplication Angriffe


JavaScript
Virtueller Adressraum
Opfer
4
0
Δ in µs
Zeit
Physikalische Adresse

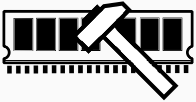
Angreifer lernt, dass Opfer gleiche Daten im Speicher hatte

Bilder von Webseiten detektieren (JavaScript, Cross-VM, KVM)

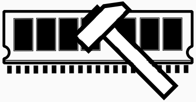


Rowhammer.js

Rowhammer

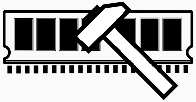


Rowhammer



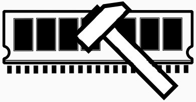
- DRAM Fehler der Bit-Flips verursacht

Rowhammer



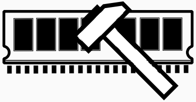
- DRAM Fehler der Bit-Flips verursacht
- Kann für Exploits verwendet werden

Rowhammer



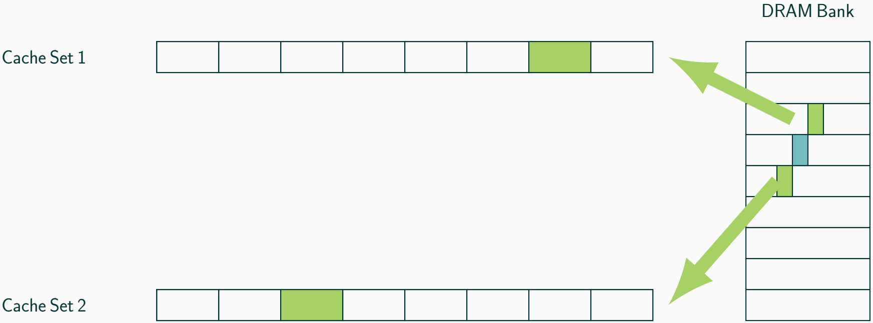
- DRAM Fehler der Bit-Flips verursacht
- Kann für Exploits verwendet werden
- Nur Adressen die nicht im Cache sind, erreichen den DRAM

Rowhammer

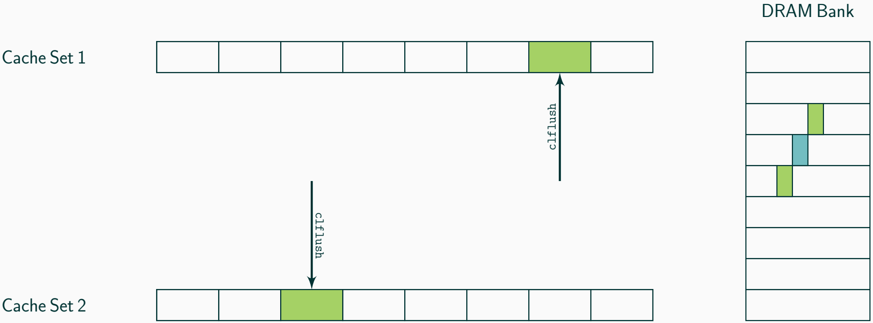


- DRAM Fehler der Bit-Flips verursacht
- Kann für Exploits verwendet werden
- Nur Adressen die nicht im Cache sind, erreichen den DRAM
    - ähnlich wie Flush+Reload

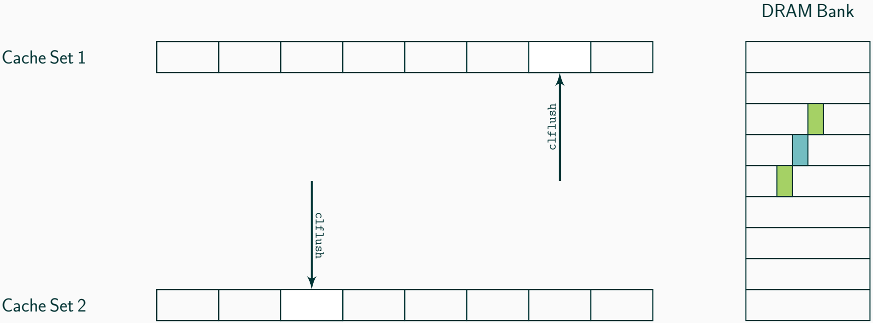
Rowhammer (mit `clflush`)


DRAM Bank
Cache Set 1
Cache Set 2

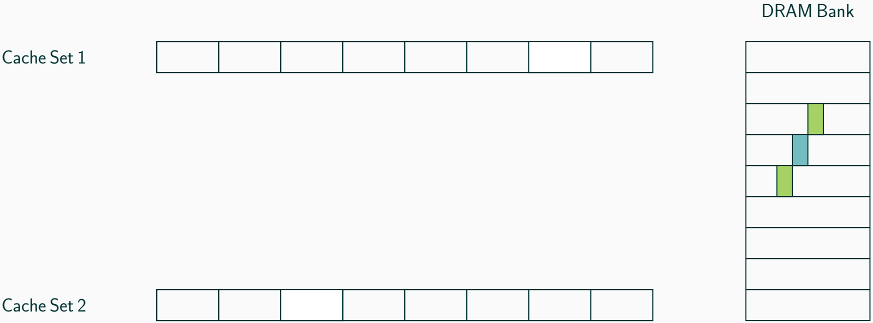
Rowhammer (mit `clflush`)


DRAM Bank
Cache Set 1
Cache Set 2
clflush
clflush

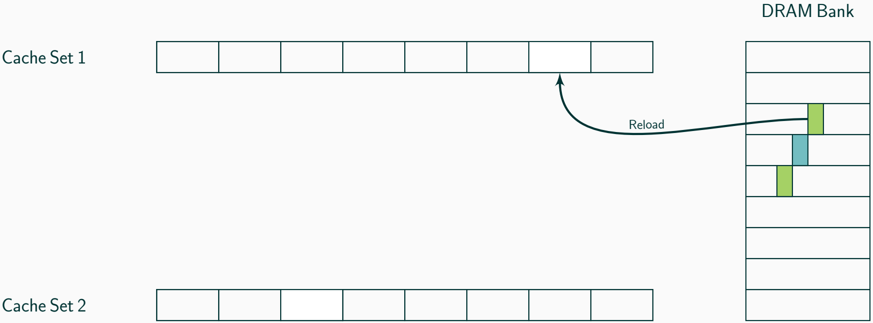
Rowhammer (mit `clflush`)


DRAM Bank
Cache Set 1
Cache Set 2
clflush
clflush

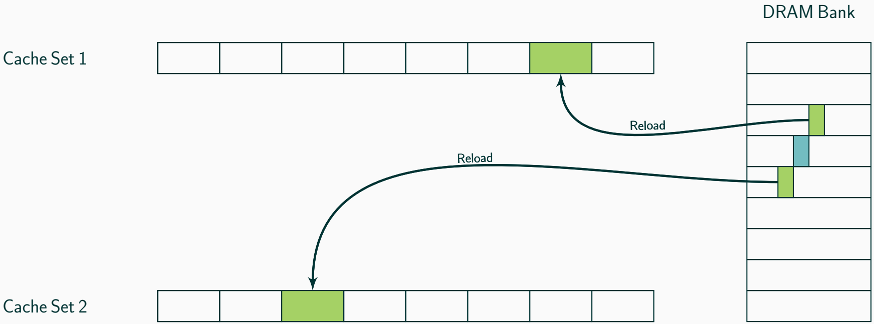
Rowhammer (mit `clflush`)


DRAM Bank
Cache Set 1
Cache Set 2

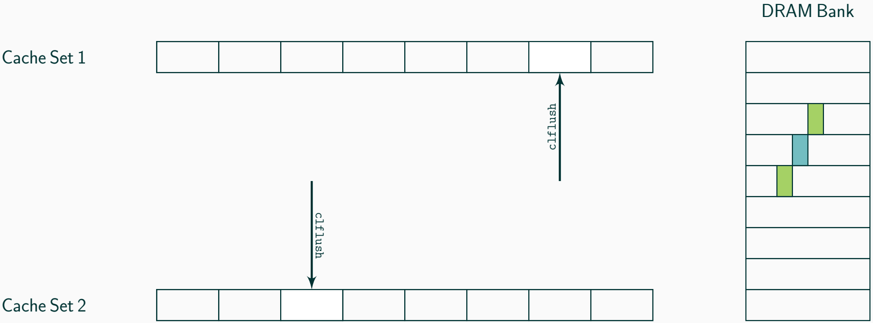
Rowhammer (mit `clflush`)


DRAM Bank
Cache Set 1
Reload
Cache Set 2

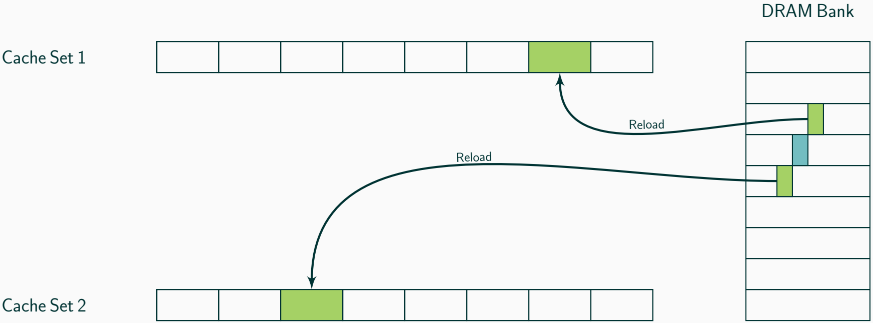
Rowhammer (mit `clflush`)


DRAM Bank
Cache Set 1
Reload
Reload
Cache Set 2

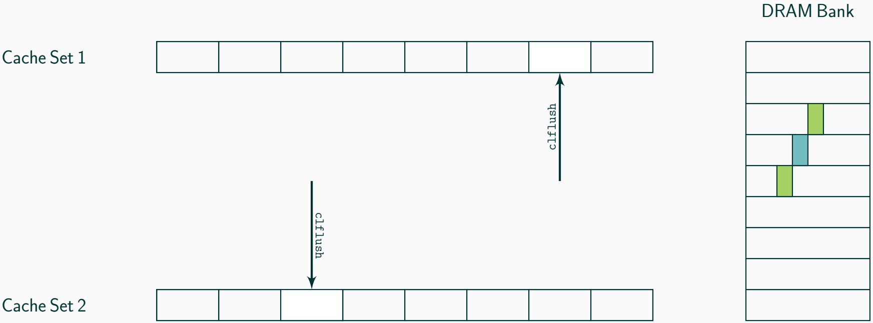
Rowhammer (mit `clflush`)


DRAM Bank
Cache Set 1
Cache Set 2
clflush
clflush

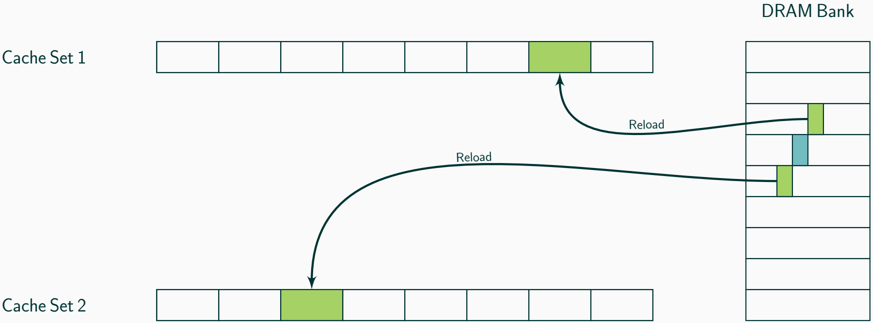
Rowhammer (mit `clflush`)


DRAM Bank
Cache Set 1
Cache Set 2
Reload
Reload

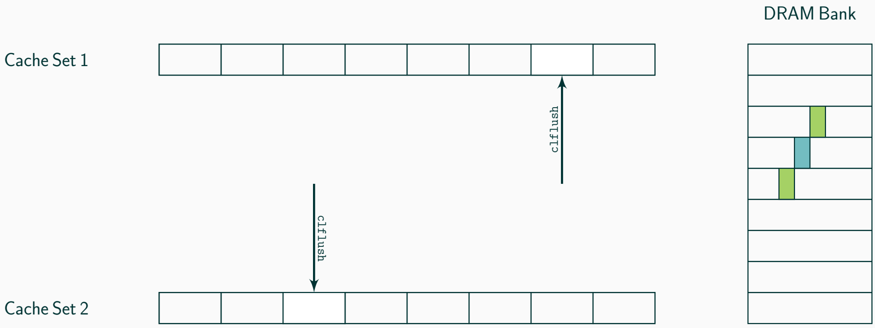
Rowhammer (mit `clflush`)


DRAM Bank
Cache Set 1
Cache Set 2
clflush
clflush

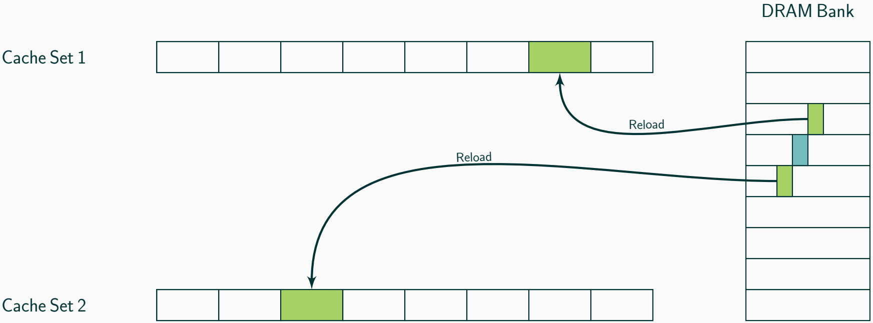
Rowhammer (mit `clflush`)


DRAM Bank
Cache Set 1
Cache Set 2
Reload
Reload

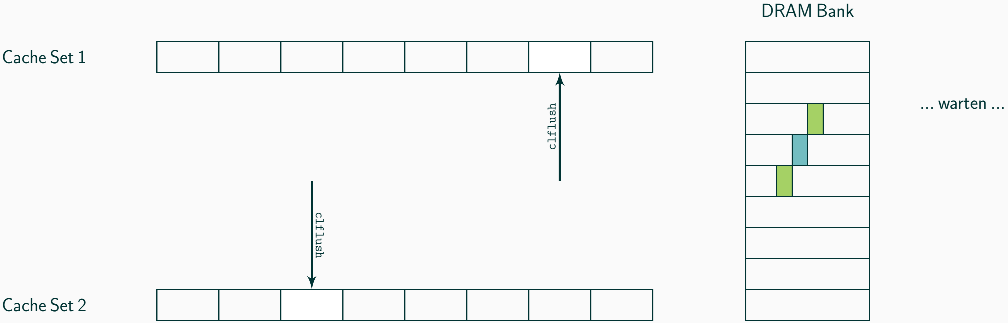
Rowhammer (mit `clflush`)


DRAM Bank
Cache Set 1
Cache Set 2
clflush
clflush

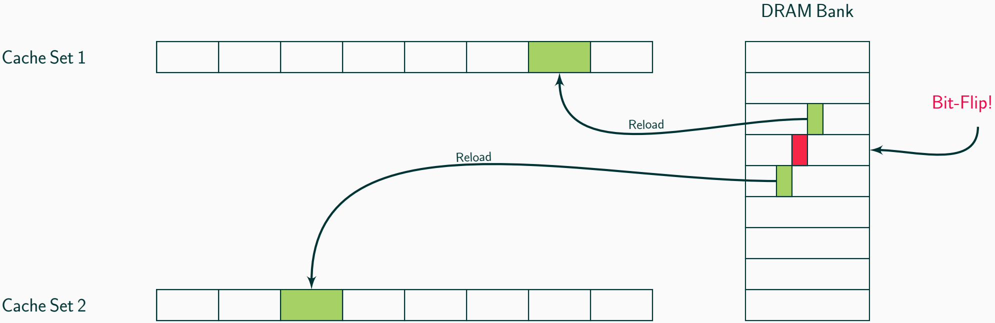
Rowhammer (mit `clflush`)


DRAM Bank
Cache Set 1
Reload
Reload
Cache Set 2

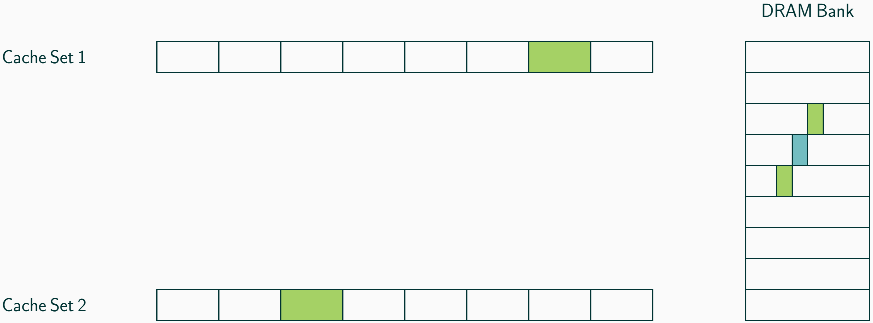
Rowhammer (mit `clflush`)


DRAM Bank
Cache Set 1
clflush
clflush
Cache Set 2
… warten …

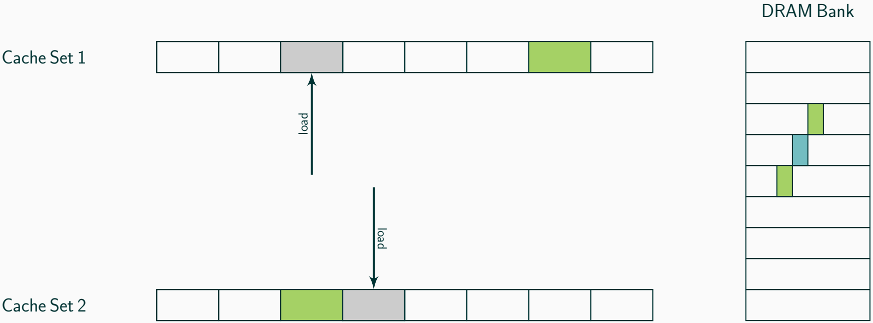
Rowhammer (mit `clflush`)


DRAM Bank
Cache Set 1
Cache Set 2
Reload
Reload
Bit-Flip!

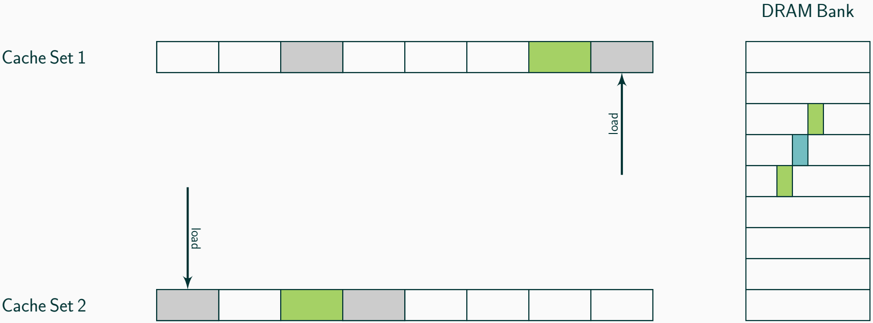
Rowhammer ohne `clflush`


DRAM Bank
Cache Set 1
Cache Set 2

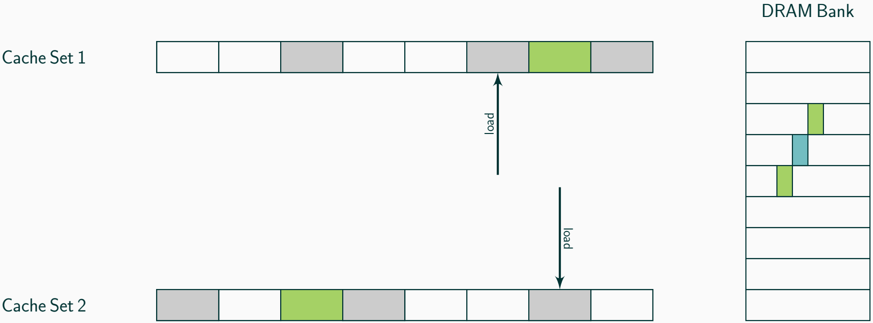
Rowhammer ohne `clflush`


DRAM Bank
Cache Set 1
Cache Set 2
load
load

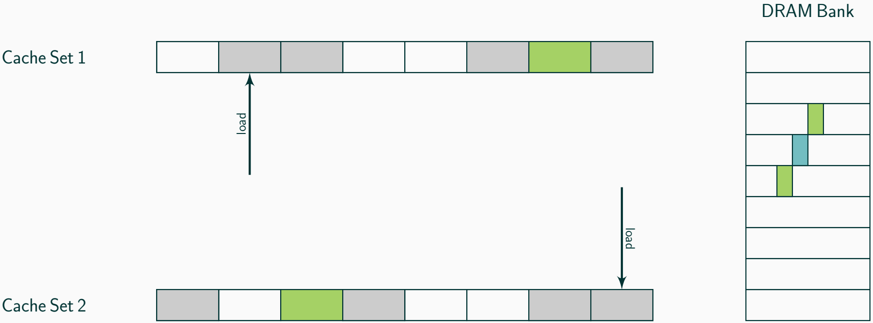
Rowhammer ohne `clflush`


DRAM Bank
Cache Set 1
load
load
Cache Set 2

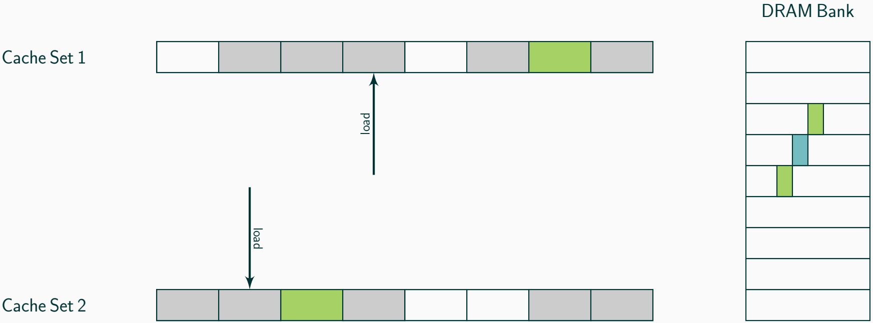
Rowhammer ohne `clflush`


DRAM Bank
Cache Set 1
load
load
Cache Set 2

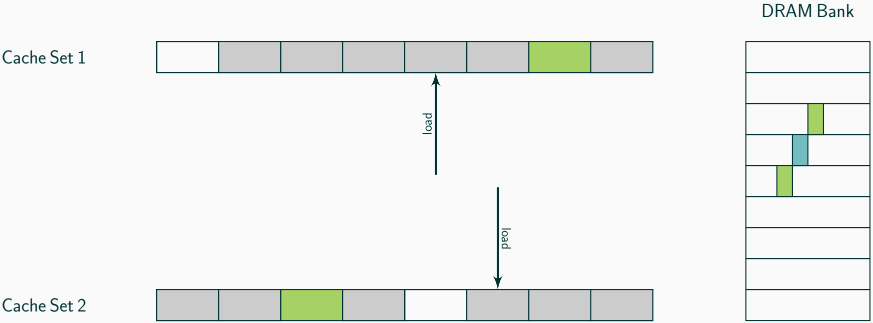
Rowhammer ohne `clflush`


DRAM Bank
Cache Set 1
load
load
Cache Set 2

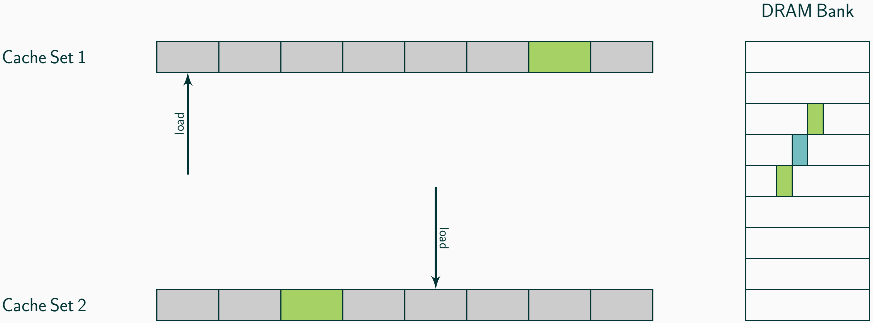
Rowhammer ohne `clflush`


DRAM Bank
Cache Set 1
load
load
Cache Set 2

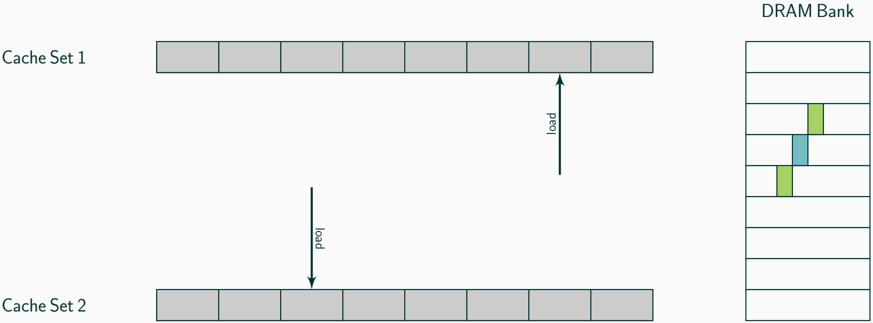
Rowhammer ohne `clflush`


DRAM Bank
Cache Set 1
load
load
Cache Set 2

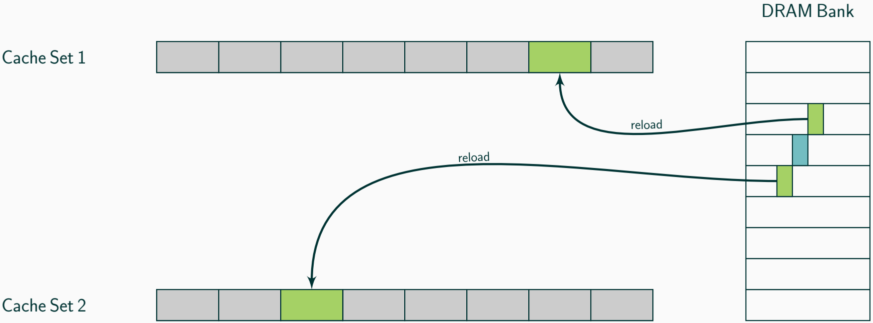
Rowhammer ohne `clflush`


DRAM Bank
Cache Set 1
load
load
Cache Set 2

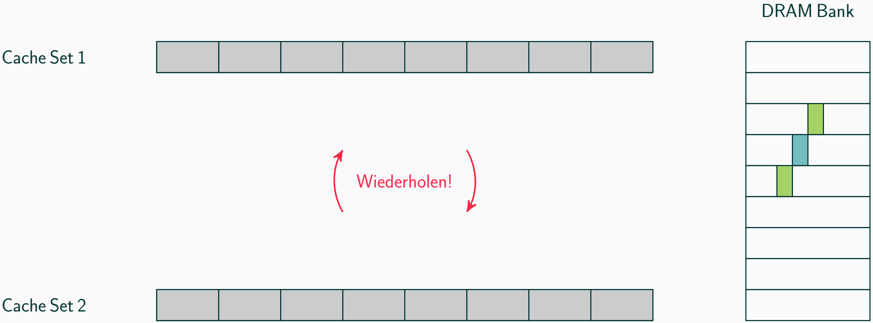
Rowhammer ohne `clflush`


DRAM Bank
Cache Set 1
load
load
Cache Set 2

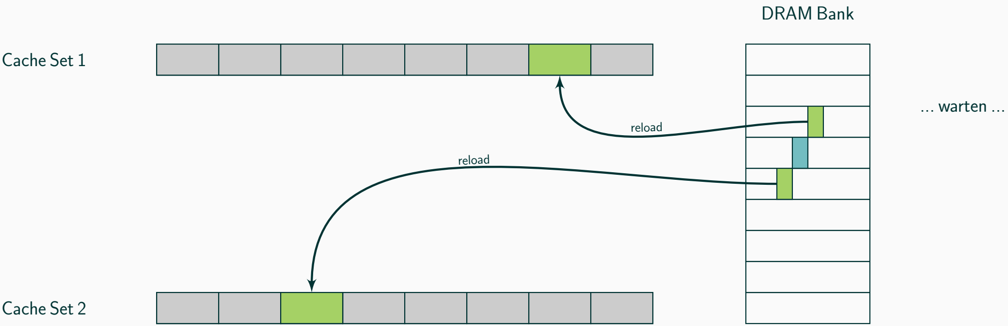
Rowhammer ohne `clflush`


DRAM Bank
Cache Set 1
Cache Set 2
reload
reload

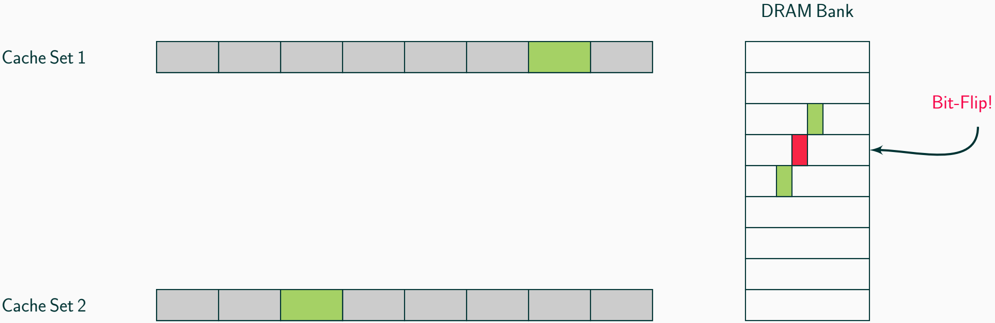
Rowhammer ohne `clflush`


DRAM Bank
Cache Set 1
Wiederholen!
Cache Set 2

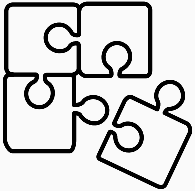
Rowhammer ohne `clflush`


DRAM Bank
Cache Set 1
Cache Set 2
reload
reload
… warten …

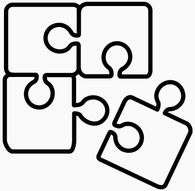
Rowhammer ohne `clflush`


DRAM Bank
Cache Set 1
Cache Set 2
Bit-Flip!

Rowhammer.js: Herausforderungen

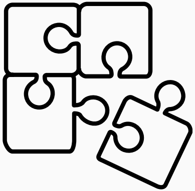


Rowhammer.js: Herausforderungen



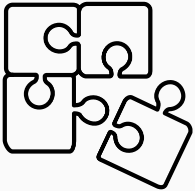
1. Wie bekommt man genau Zeitmessungen (in JS)?

Rowhammer.js: Herausforderungen



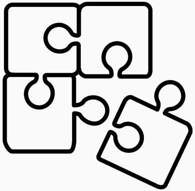
1. Wie bekommt man genau Zeitmessungen (in JS)? → einfach

Rowhammer.js: Herausforderungen



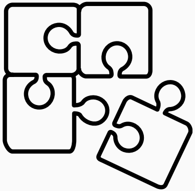
1. Wie bekommt man genau Zeitmessungen (in JS)? → einfach
2. Wie bekommt man physikalische Adressen (in JS)?

Rowhammer.js: Herausforderungen



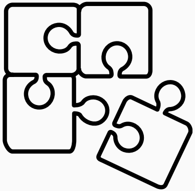
1. Wie bekommt man genau Zeitmessungen (in JS)? → einfach
2. Wie bekommt man physikalische Adressen (in JS)? → einfach

Rowhammer.js: Herausforderungen



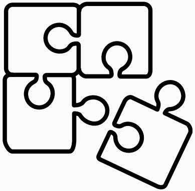
1. Wie bekommt man genau Zeitmessungen (in JS)? → einfach
2. Wie bekommt man physikalische Adressen (in JS)? → einfach
3. Welche physikalischen Adressen muss man zugreifen?

Rowhammer.js: Herausforderungen



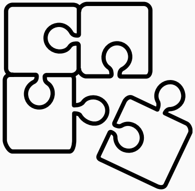
1. Wie bekommt man genau Zeitmessungen (in JS)? → einfach
2. Wie bekommt man physikalische Adressen (in JS)? → einfach
3. Welche physikalischen Adressen muss man zugreifen? → bereits gelöst

Rowhammer.js: Herausforderungen



1. Wie bekommt man genau Zeitmessungen (in JS)? → einfach
2. Wie bekommt man physikalische Adressen (in JS)? → einfach
3. Welche physikalischen Adressen muss man zugreifen? → bereits gelöst
4. In welcher Reihenfolge muss man zugreifen?

Rowhammer.js: Herausforderungen



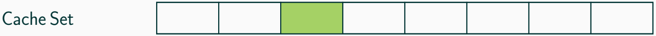
1. Wie bekommt man genau Zeitmessungen (in JS)? → einfach
2. Wie bekommt man physikalische Adressen (in JS)? → einfach
3. Welche physikalischen Adressen muss man zugreifen? → bereits gelöst
4. In welcher Reihenfolge muss man zugreifen? → lösen wir jetzt

Replacement Policy auf älteren Prozessoren

“LRU eviction” Speicherzugriffe

Replacement Policy auf älteren Prozessoren

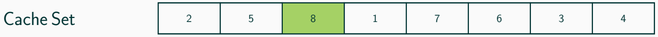
“LRU eviction” Speicherzugriffe



- LRU replacement policy: ältester Eintrag zuerst

Replacement Policy auf älteren Prozessoren

"LRU eviction" Speicherzugriffe

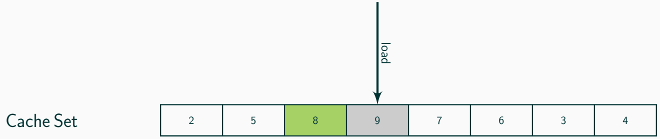


Cache Set	2	5	8	1	7	6	3	4

- LRU replacement policy: ältester Eintrag zuerst
- Zeitstempel für jede Cache Zeile

Replacement Policy auf älteren Prozessoren

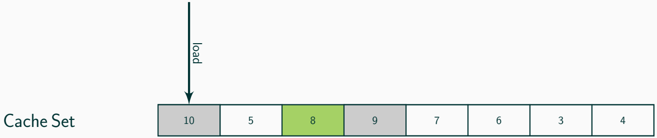
"LRU eviction" Speicherzugriffe



- LRU replacement policy: ältester Eintrag zuerst
- Zeitstempel für jede Cache Zeile
- Zugriff erneuert den Zeitstempel

Replacement Policy auf älteren Prozessoren

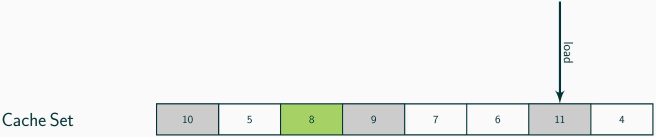
“LRU eviction” Speicherzugriffe



- LRU replacement policy: ältester Eintrag zuerst
- Zeitstempel für jede Cache Zeile
- Zugriff erneuert den Zeitstempel

Replacement Policy auf älteren Prozessoren

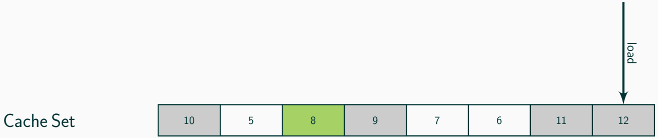
"LRU eviction" Speicherzugriffe



- LRU replacement policy: ältester Eintrag zuerst
- Zeitstempel für jede Cache Zeile
- Zugriff erneuert den Zeitstempel

Replacement Policy auf älteren Prozessoren

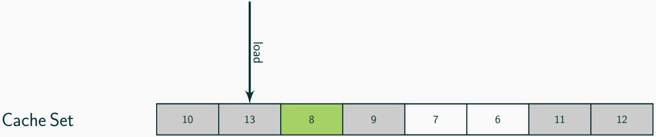
“LRU eviction” Speicherzugriffe



- LRU replacement policy: ältester Eintrag zuerst
- Zeitstempel für jede Cache Zeile
- Zugriff erneuert den Zeitstempel

Replacement Policy auf älteren Prozessoren

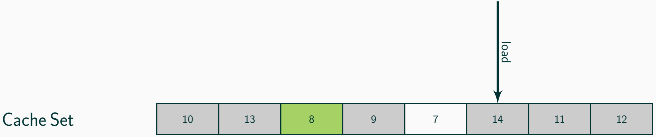
“LRU eviction” Speicherzugriffe


load
Cache Set
10
13
8
9
7
6
11
12

- LRU replacement policy: ältester Eintrag zuerst
- Zeitstempel für jede Cache Zeile
- Zugriff erneuert den Zeitstempel

Replacement Policy auf älteren Prozessoren

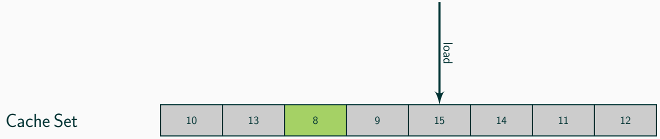
"LRU eviction" Speicherzugriffe


load
Cache Set
10
13
8
9
7
14
11
12

- LRU replacement policy: ältester Eintrag zuerst
- Zeitstempel für jede Cache Zeile
- Zugriff erneuert den Zeitstempel

Replacement Policy auf älteren Prozessoren

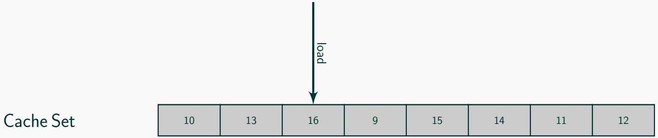
"LRU eviction" Speicherzugriffe


load
Cache Set
10
13
8
9
15
14
11
12

- LRU replacement policy: ältester Eintrag zuerst
- Zeitstempel für jede Cache Zeile
- Zugriff erneuert den Zeitstempel

Replacement Policy auf älteren Prozessoren

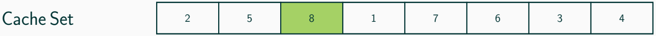
“LRU eviction” Speicherzugriffe


load
Cache Set
10
13
16
9
15
14
11
12

- LRU replacement policy: ältester Eintrag zuerst
- Zeitstempel für jede Cache Zeile
- Zugriff erneuert den Zeitstempel

Replacement Policy auf neueren CPUs

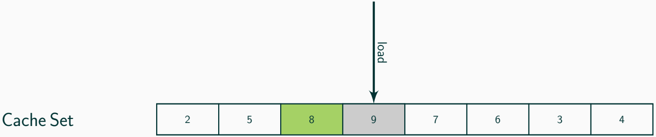
“LRU eviction” Speicherzugriffe


Cache Set
2
5
8
1
7
6
3
4

- Nicht dokumentierte Replacement Policy

Replacement Policy auf neueren CPUs

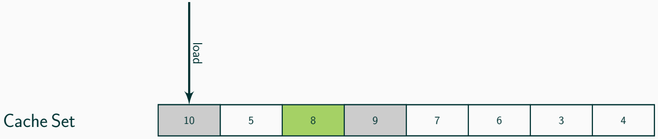
"LRU eviction" Speicherzugriffe


load
Cache Set
2
5
8
9
7
6
3
4

- Nicht dokumentierte Replacement Policy

Replacement Policy auf neueren CPUs

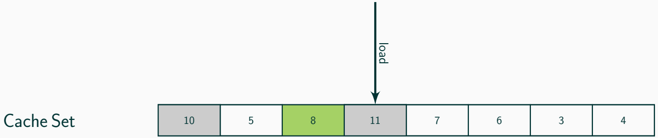
"LRU eviction" Speicherzugriffe


load
Cache Set
10
5
8
9
7
6
3
4

- Nicht dokumentierte Replacement Policy

Replacement Policy auf neueren CPUs

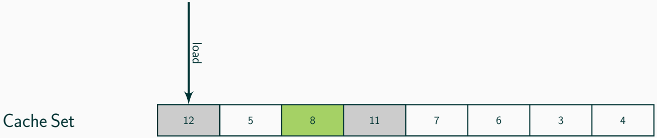
"LRU eviction" Speicherzugriffe


load
Cache Set
10
5
8
11
7
6
3
4

- Nicht dokumentierte Replacement Policy

Replacement Policy auf neueren CPUs

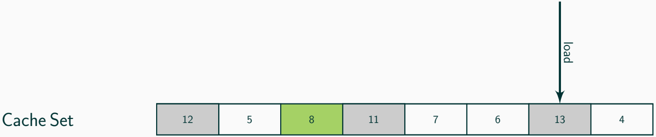
“LRU eviction” Speicherzugriffe


load
Cache Set
12
5
8
11
7
6
3
4

- Nicht dokumentierte Replacement Policy

Replacement Policy auf neueren CPUs

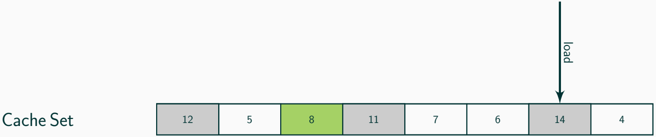
"LRU eviction" Speicherzugriffe


load
Cache Set
12
5
8
11
7
6
13
4

- Nicht dokumentierte Replacement Policy

Replacement Policy auf neueren CPUs

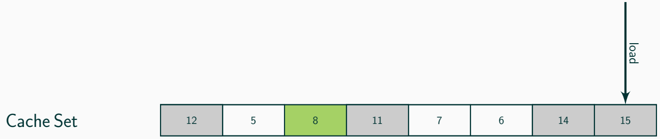
"LRU eviction" Speicherzugriffe


load
Cache Set
12
5
8
11
7
6
14
4

- Nicht dokumentierte Replacement Policy

Replacement Policy auf neueren CPUs

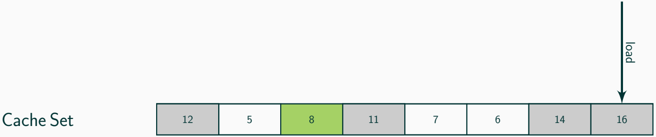
"LRU eviction" Speicherzugriffe


load
Cache Set
12
5
8
11
7
6
14
15

- Nicht dokumentierte Replacement Policy

Replacement Policy auf neueren CPUs

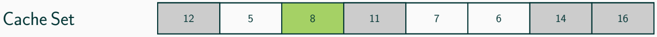
“LRU eviction” Speicherzugriffe


load
Cache Set
12
5
8
11
7
6
14
16

- Nicht dokumentierte Replacement Policy

Replacement Policy auf neueren CPUs

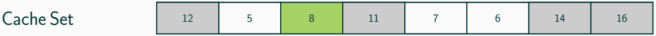
"LRU eviction" Speicherzugriffe


Cache Set
12
5
8
11
7
6
14
16

- Nicht dokumentierte Replacement Policy
- Nur 75% Erfolgsrate auf Haswell

Replacement Policy auf neueren CPUs

“LRU eviction” Speicherzugriffe



Cache Set	12	5	8	11	7	6	14	16

- Nicht dokumentierte Replacement Policy
- Nur 75% Erfolgsrate auf Haswell
- Mehr Zugriffe → mehr Erfolg, aber zu langsam

Cache-Eviction-Strategien: Notation

```
for (s = 0; s <= S-D; s += L)
  for (c = 1; c <= C; c += 1)
    for (d = 1; d <= D; d += 1)
      *a[s+d];
```

Cache-Eviction-Strategien: Notation

```
for (s = 0; s <= S-D; s += L)
  for (c = 1; c <= C; c += 1)
    for (d = 1; d <= D; d += 1)
      *a[s+d];
```

- $\mathscr{P}$-2-2-1-4 → 1, 2, 1, 2, 2, 3, 2, 3, 3, 4, 3, 4

Cache-Eviction-Strategien: Notation

```
for (s = 0; s <= S-D; s += L)
  for (c = 1; c <= C; c += 1)
    for (d = 1; d <= D; d += 1)
      *a[s+d];
```

- $\mathscr{P}$-2-2-1-4 → 1, 2, 1, 2, 2, 3, 2, 3, 3, 4, 3, 4
- $\mathscr{P}$-1-1-1-4 → $1, 2, 3, 4$ → LRU eviction mit Set Größe 4

Cache-Eviction-Strategien: Auswertung

Wir haben über 10 000 Strategien ausgewertet...

Strategie	# Zugriffe	Eviction-Rate	Ausführungszeit
$\mathscr{P}$-1-1-1-17	17		
$\mathscr{P}$-1-1-1-20	20		

Ausführung in einer Schleife, auf einem Haswell mit 16-Wege Last-Level Cache

Cache-Eviction-Strategien: Auswertung

Wir haben über 10 000 Strategien ausgewertet...

Strategie	# Zugriffe	Eviction-Rate	Ausführungszeit
$\mathscr{P}$-1-1-1-17	17	74.46% ✗	
$\mathscr{P}$-1-1-1-20	20	99.82% ✓	

Ausführung in einer Schleife, auf einem Haswell mit 16-Wege Last-Level Cache

Cache-Eviction-Strategien: Auswertung

Wir haben über 10 000 Strategien ausgewertet...

Strategie	# Zugriffe	Eviction-Rate	Ausführungszeit
$\mathscr{P}$-1-1-1-17	17	74.46% ✗	307 ns ✓
$\mathscr{P}$-1-1-1-20	20	99.82% ✓	934 ns ✗

Ausführung in einer Schleife, auf einem Haswell mit 16-Wege Last-Level Cache

Cache-Eviction-Strategien: Auswertung

Wir haben über 10 000 Strategien ausgewertet...

Strategie	# Zugriffe	Eviction-Rate	Ausführungszeit
$\mathscr{P}$-1-1-1-17	17	74.46% ✗	307 ns ✓
$\mathscr{P}$-1-1-1-20	20	99.82% ✓	934 ns ✗
$\mathscr{P}$-2-1-1-17	34		

Ausführung in einer Schleife, auf einem Haswell mit 16-Wege Last-Level Cache

Cache-Eviction-Strategien: Auswertung

Wir haben über 10 000 Strategien ausgewertet...

Strategie	# Zugriffe	Eviction-Rate	Ausführungszeit
$\mathscr{P}$-1-1-1-17	17	74.46% ✗	307 ns ✓
$\mathscr{P}$-1-1-1-20	20	99.82% ✓	934 ns ✗
$\mathscr{P}$-2-1-1-17	34	99.86% ✓	

Ausführung in einer Schleife, auf einem Haswell mit 16-Wege Last-Level Cache

Cache-Eviction-Strategien: Auswertung

Wir haben über 10 000 Strategien ausgewertet...

Strategie	# Zugriffe	Eviction-Rate	Ausführungszeit
$\mathscr{P}$-1-1-1-17	17	74.46% ✗	307 ns ✓
$\mathscr{P}$-1-1-1-20	20	99.82% ✓	934 ns ✗
$\mathscr{P}$-2-1-1-17	34	99.86% ✓	191 ns ✓

Ausführung in einer Schleife, auf einem Haswell mit 16-Wege Last-Level Cache

Cache-Eviction-Strategien: Auswertung

Wir haben über 10 000 Strategien ausgewertet...

Strategie	# Zugriffe	Eviction-Rate	Ausführungszeit
$\mathscr{P}$-1-1-1-17	17	74.46% ✗	307 ns ✓
$\mathscr{P}$-1-1-1-20	20	99.82% ✓	934 ns ✗
$\mathscr{P}$-2-1-1-17	34	99.86% ✓	191 ns ✓
$\mathscr{P}$-2-2-1-17	64		

Ausführung in einer Schleife, auf einem Haswell mit 16-Wege Last-Level Cache

Cache-Eviction-Strategien: Auswertung

Wir haben über 10 000 Strategien ausgewertet...

Strategie	# Zugriffe	Eviction-Rate	Ausführungszeit
$\mathscr{P}$-1-1-1-17	17	74.46% ✗	307 ns ✓
$\mathscr{P}$-1-1-1-20	20	99.82% ✓	934 ns ✗
$\mathscr{P}$-2-1-1-17	34	99.86% ✓	191 ns ✓
$\mathscr{P}$-2-2-1-17	64	99.98% ✓	

Ausführung in einer Schleife, auf einem Haswell mit 16-Wege Last-Level Cache

Cache-Eviction-Strategien: Auswertung

Wir haben über 10 000 Strategien ausgewertet...

Strategie	# Zugriffe	Eviction-Rate	Ausführungszeit
$\mathscr{P}$-1-1-1-17	17	74.46% ✗	307 ns ✓
$\mathscr{P}$-1-1-1-20	20	99.82% ✓	934 ns ✗
$\mathscr{P}$-2-1-1-17	34	99.86% ✓	191 ns ✓
$\mathscr{P}$-2-2-1-17	64	99.98% ✓	180 ns ✓

Ausführung in einer Schleife, auf einem Haswell mit 16-Wege Last-Level Cache

Cache-Eviction-Strategien: Auswertung

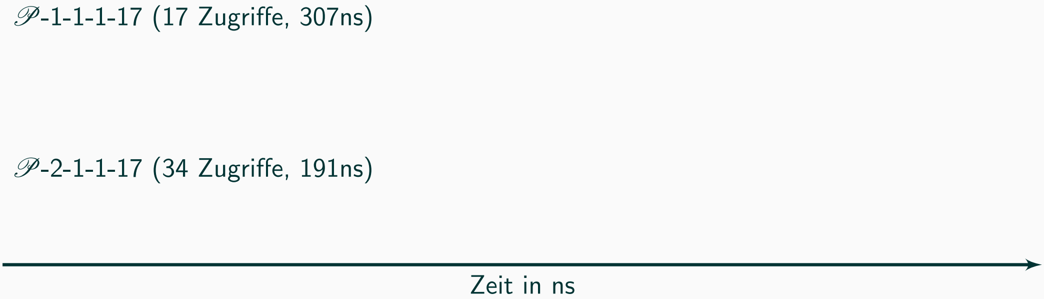
Wir haben über 10 000 Strategien ausgewertet…

Strategie	# Zugriffe	Eviction-Rate	Ausführungszeit
$\mathscr{P}$-1-1-1-17	17	74.46% ✗	307 ns ✓
$\mathscr{P}$-1-1-1-20	20	99.82% ✓	934 ns ✗
$\mathscr{P}$-2-1-1-17	34	99.86% ✓	191 ns ✓
$\mathscr{P}$-2-2-1-17	64	99.98% ✓	180 ns ✓

→ mehr Zugriffe, geringere Ausführungszeit?

Ausführung in einer Schleife, auf einem Haswell mit 16-Wege Last-Level Cache

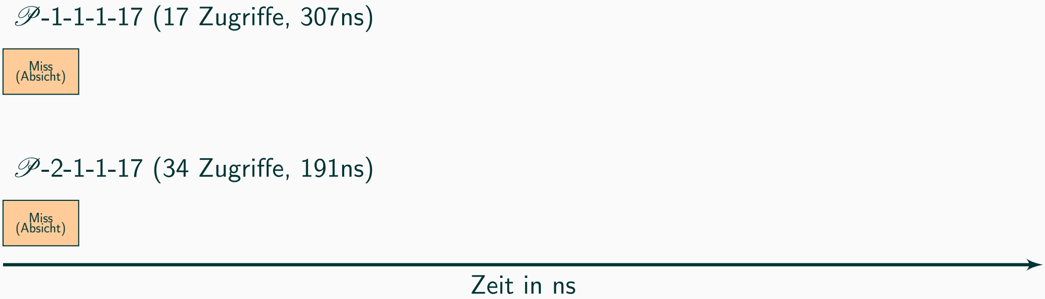
Cache-Eviction-Strategien: Illustration



$\mathscr{P}$-1-1-1-17 (17 Zugriffe, 307ns)

$\mathscr{P}$-2-1-1-17 (34 Zugriffe, 191ns)

Cache-Eviction-Strategien: Illustration



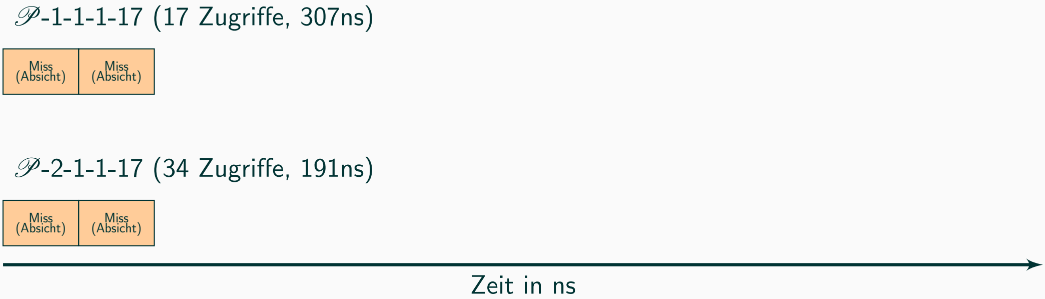
$\mathscr{P}$-1-1-1-17 (17 Zugriffe, 307ns)

Miss
(Absicht)

$\mathscr{P}$-2-1-1-17 (34 Zugriffe, 191ns)

Miss
(Absicht)
Zeit in ns

Cache-Eviction-Strategien: Illustration



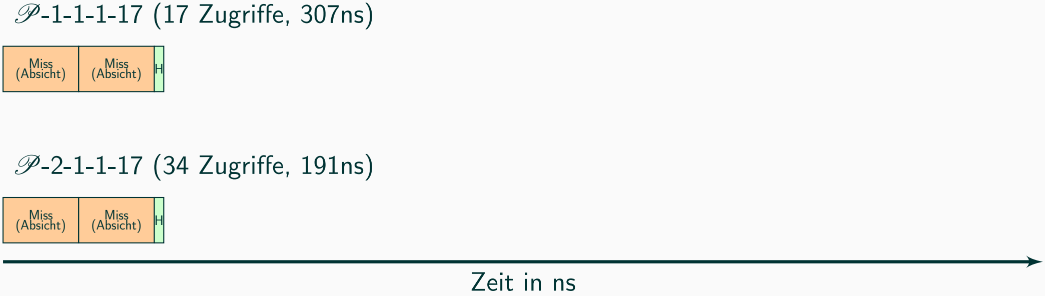
$\mathscr{P}$-1-1-1-17 (17 Zugriffe, 307ns)

Miss
(Absicht)
Miss
(Absicht)

$\mathscr{P}$-2-1-1-17 (34 Zugriffe, 191ns)

Miss
(Absicht)
Miss
(Absicht)
Zeit in ns

Cache-Eviction-Strategien: Illustration

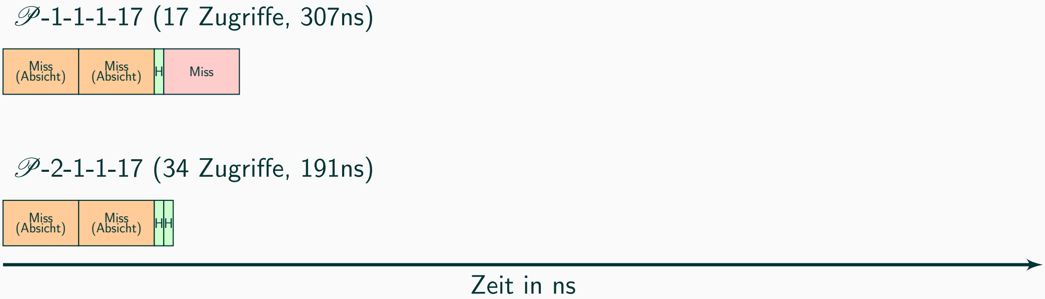

$\mathscr{P}$-1-1-1-17 (17 Zugriffe, 307ns)

Miss (Absicht)
Miss (Absicht)
H

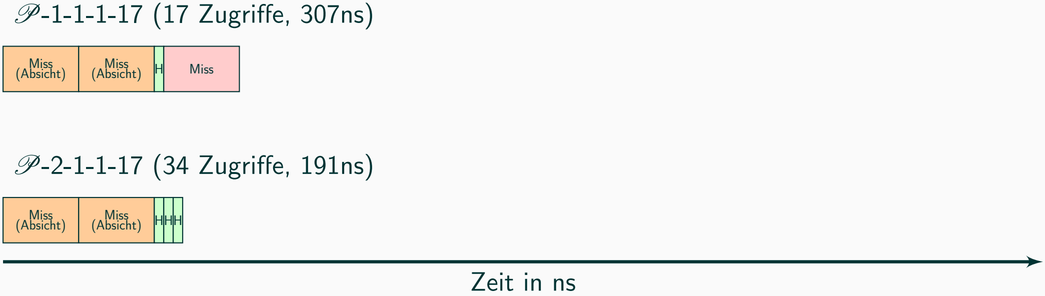
$\mathscr{P}$-2-1-1-17 (34 Zugriffe, 191ns)

Miss (Absicht)
Miss (Absicht)
H
Zeit in ns

Cache-Eviction-Strategien: Illustration


$\mathscr{P}$-1-1-1-17 (17 Zugriffe, 307ns)
Miss (Absicht)
Miss (Absicht)
H
Miss
$\mathscr{P}$-2-1-1-17 (34 Zugriffe, 191ns)
Miss (Absicht)
Miss (Absicht)
H
H
Zeit in ns

Cache-Eviction-Strategien: Illustration



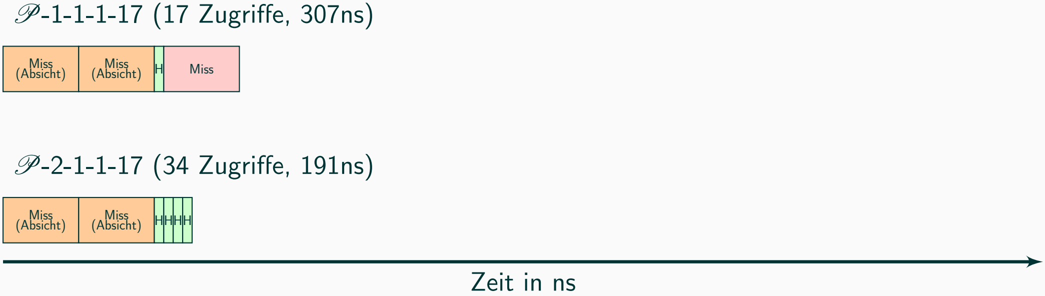
$\mathscr{P}$-1-1-1-17 (17 Zugriffe, 307ns)

Miss
(Absicht)
Miss
(Absicht)
H
Miss

$\mathscr{P}$-2-1-1-17 (34 Zugriffe, 191ns)

Miss
(Absicht)
Miss
(Absicht)
H
H
H
Zeit in ns

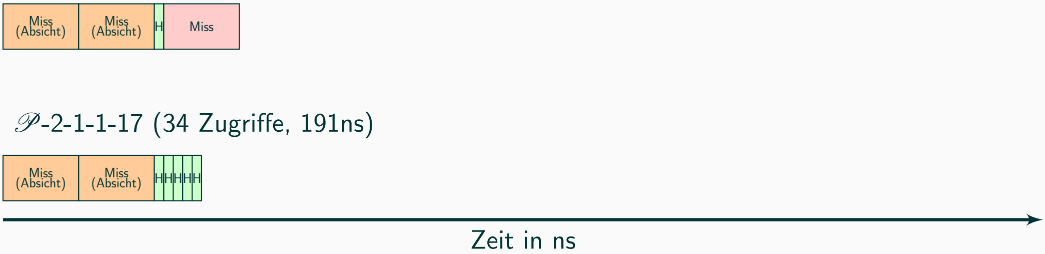
Cache-Eviction-Strategien: Illustration


$\mathscr{P}$-1-1-1-17 (17 Zugriffe, 307ns)
Miss (Absicht)
Miss (Absicht)
H
Miss
$\mathscr{P}$-2-1-1-17 (34 Zugriffe, 191ns)
Miss (Absicht)
Miss (Absicht)
H
H
H
H
Zeit in ns

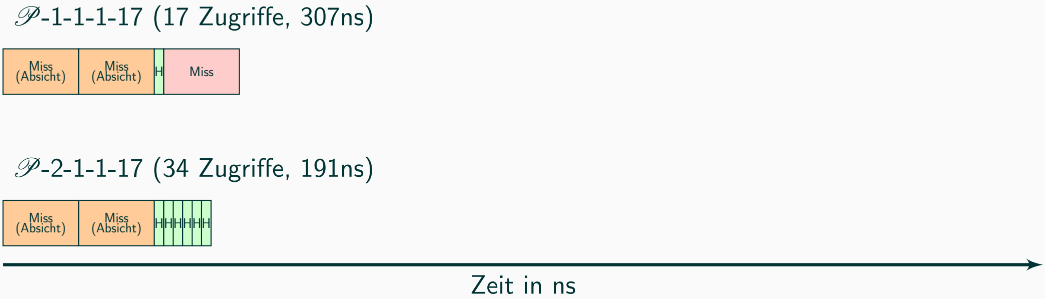
Cache-Eviction-Strategien: Illustration

$\mathscr{P}$-1-1-1-17 (17 Zugriffe, 307ns)

$\mathscr{P}$-2-1-1-17 (34 Zugriffe, 191ns)


Miss (Absicht)
Miss (Absicht)
H
Miss
Miss (Absicht)
Miss (Absicht)
H
H
H
H
H
Zeit in ns

Cache-Eviction-Strategien: Illustration



$\mathscr{P}$-1-1-1-17 (17 Zugriffe, 307ns)

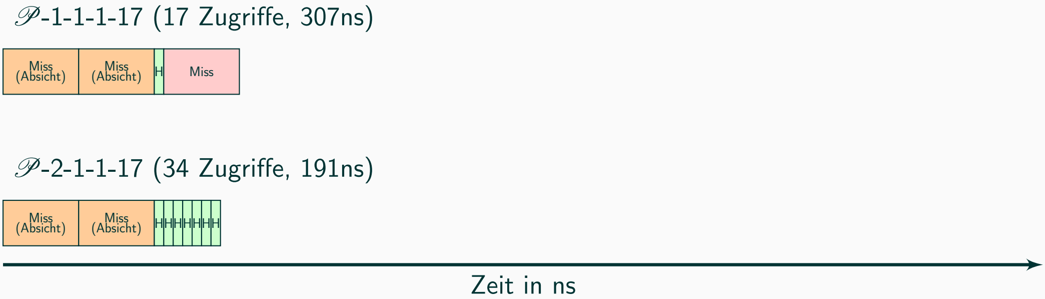
Miss (Absicht) | Miss (Absicht) | H | Miss

$\mathscr{P}$-2-1-1-17 (34 Zugriffe, 191ns)

Miss (Absicht) | Miss (Absicht) | H | H | H | H | H | H

Zeit in ns

Cache-Eviction-Strategien: Illustration



$\mathscr{P}$-1-1-1-17 (17 Zugriffe, 307ns)

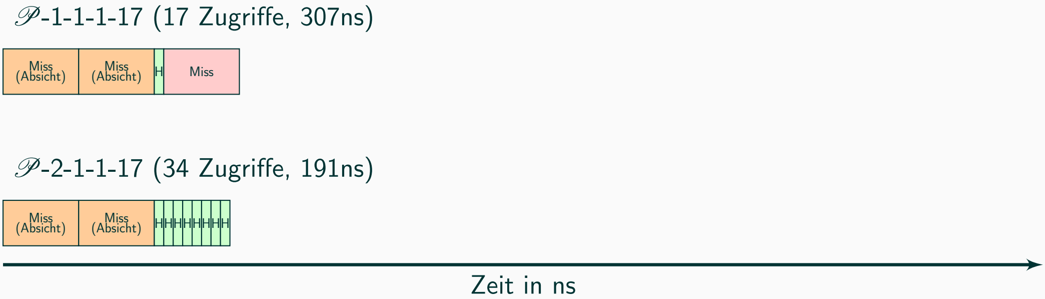
Miss (Absicht) | Miss (Absicht) | H | Miss

$\mathscr{P}$-2-1-1-17 (34 Zugriffe, 191ns)

Miss (Absicht) | Miss (Absicht) | H | H | H | H | H | H | H

Zeit in ns

Cache-Eviction-Strategien: Illustration



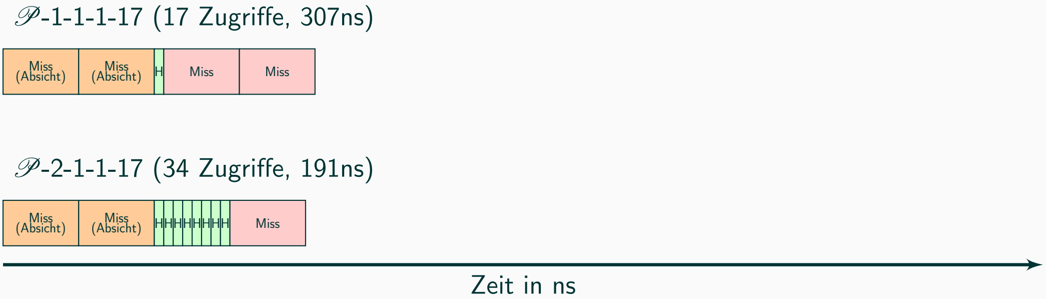
$\mathscr{P}$-1-1-1-17 (17 Zugriffe, 307ns)

Miss
(Absicht)
Miss
(Absicht)
H
Miss

$\mathscr{P}$-2-1-1-17 (34 Zugriffe, 191ns)

Miss
(Absicht)
Miss
(Absicht)
H
H
H
H
H
H
H
H
Zeit in ns

Cache-Eviction-Strategien: Illustration



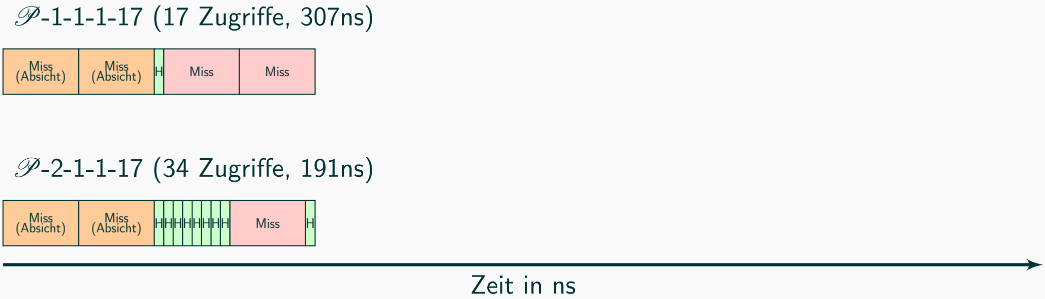
$\mathscr{P}$-1-1-1-17 (17 Zugriffe, 307ns)

Miss (Absicht)
Miss (Absicht)
H
Miss
Miss

$\mathscr{P}$-2-1-1-17 (34 Zugriffe, 191ns)

Miss (Absicht)
Miss (Absicht)
H H H H H H H H
Miss
Zeit in ns

Cache-Eviction-Strategien: Illustration



$\mathscr{P}$-1-1-1-17 (17 Zugriffe, 307ns)

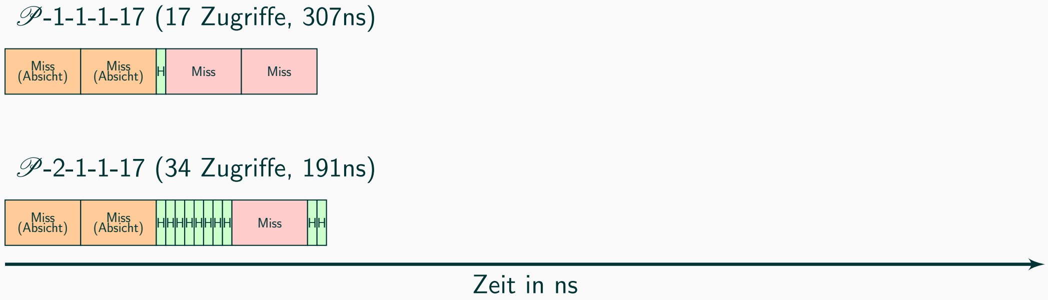
Miss (Absicht) | Miss (Absicht) | H | Miss | Miss

$\mathscr{P}$-2-1-1-17 (34 Zugriffe, 191ns)

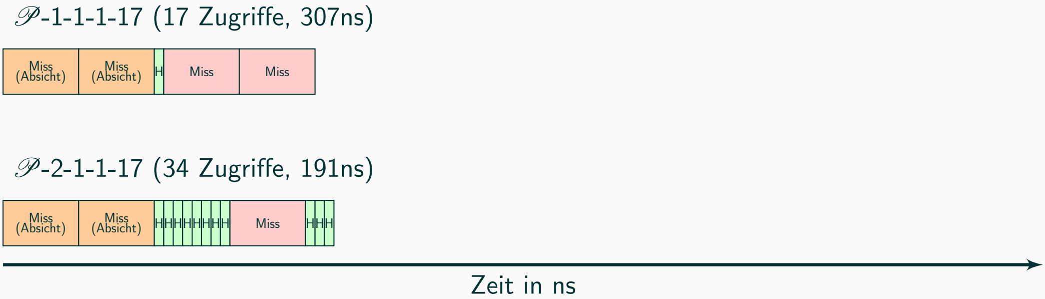
Miss (Absicht) | Miss (Absicht) | H | H | H | H | H | H | H | H | Miss | H

Zeit in ns

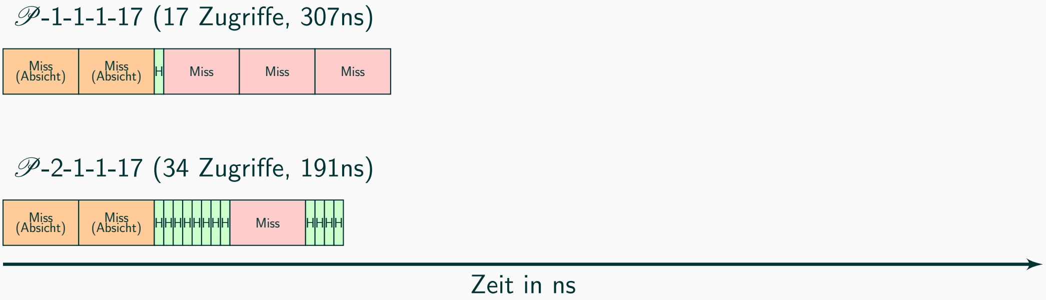
Cache-Eviction-Strategien: Illustration


$\mathscr{P}$-1-1-1-17 (17 Zugriffe, 307ns)
Miss (Absicht)
Miss (Absicht)
H
Miss
Miss
$\mathscr{P}$-2-1-1-17 (34 Zugriffe, 191ns)
Miss (Absicht)
Miss (Absicht)
H H H H H H H H
Miss
H H
Zeit in ns

Cache-Eviction-Strategien: Illustration


𝒫-1-1-1-17 (17 Zugriffe, 307ns)
Miss (Absicht)
Miss (Absicht)
H
Miss
Miss
𝒫-2-1-1-17 (34 Zugriffe, 191ns)
Miss (Absicht)
Miss (Absicht)
H H H H H H H H
Miss
H H H
Zeit in ns

Cache-Eviction-Strategien: Illustration

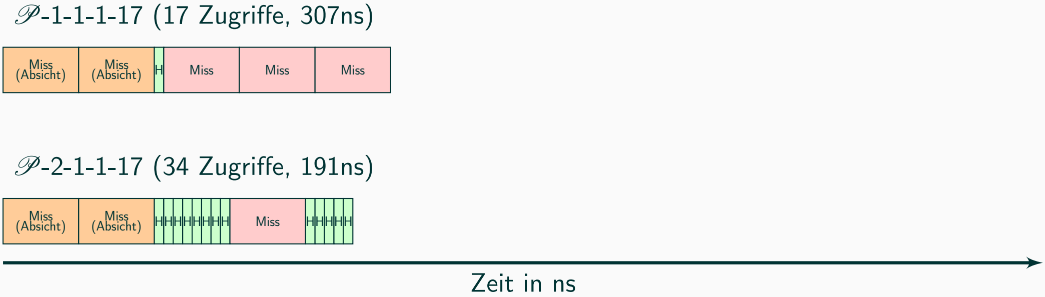


$\mathscr{P}$-1-1-1-17 (17 Zugriffe, 307ns)

$\mathscr{P}$-2-1-1-17 (34 Zugriffe, 191ns)

Miss (Absicht)
Miss (Absicht)
H
Miss
Miss
Miss
Miss (Absicht)
Miss (Absicht)
H H H H H H H H
Miss
H H H H
Zeit in ns

Cache-Eviction-Strategien: Illustration



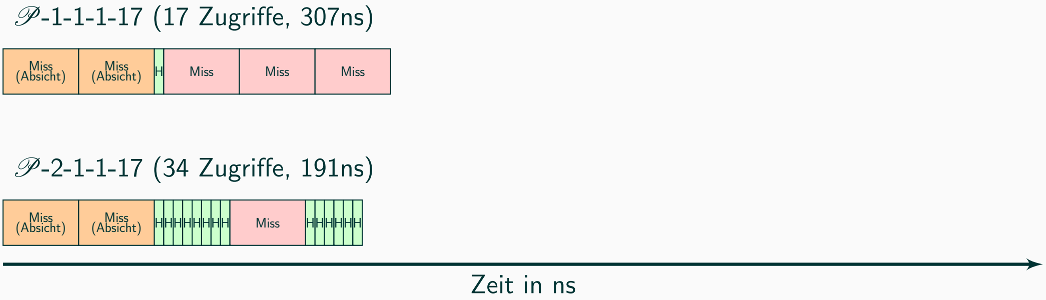
$\mathscr{P}$-1-1-1-17 (17 Zugriffe, 307ns)

Miss (Absicht)
Miss (Absicht)
H
Miss
Miss
Miss

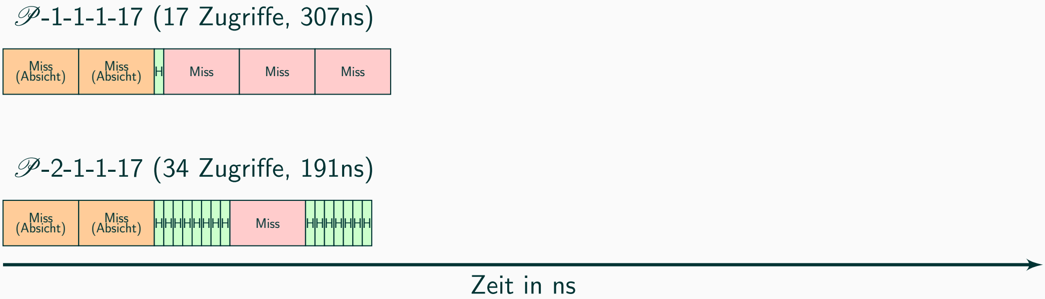
$\mathscr{P}$-2-1-1-17 (34 Zugriffe, 191ns)

Miss (Absicht)
Miss (Absicht)
H H H H H H H H
Miss
H H H H H
Zeit in ns

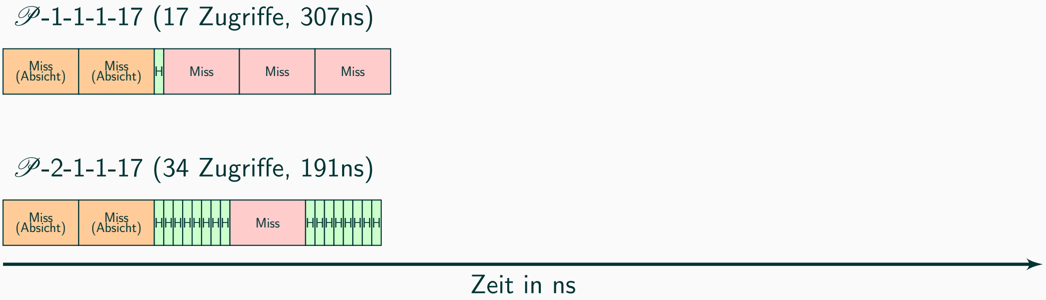
Cache-Eviction-Strategien: Illustration


$\mathscr{P}$-1-1-1-17 (17 Zugriffe, 307ns)
Miss (Absicht)
Miss (Absicht)
H
Miss
Miss
Miss
$\mathscr{P}$-2-1-1-17 (34 Zugriffe, 191ns)
Miss (Absicht)
Miss (Absicht)
H H H H H H H H
Miss
H H H H H H
Zeit in ns

Cache-Eviction-Strategien: Illustration


$\mathscr{P}$-1-1-1-17 (17 Zugriffe, 307ns)
Miss (Absicht)
Miss (Absicht)
H
Miss
Miss
Miss
$\mathscr{P}$-2-1-1-17 (34 Zugriffe, 191ns)
Miss (Absicht)
Miss (Absicht)
H H H H H H H H
Miss
H H H H H H H
Zeit in ns

Cache-Eviction-Strategien: Illustration



$\mathscr{P}$-1-1-1-17 (17 Zugriffe, 307ns)

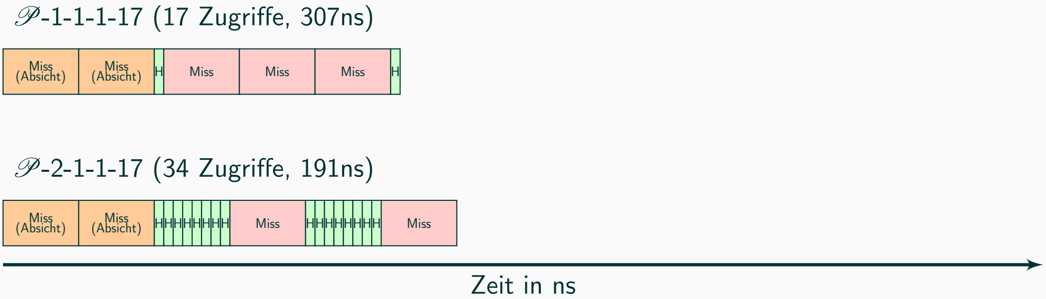
Miss (Absicht) | Miss (Absicht) | H | Miss | Miss | Miss

$\mathscr{P}$-2-1-1-17 (34 Zugriffe, 191ns)

Miss (Absicht) | Miss (Absicht) | H | H | H | H | H | H | H | H | Miss | H | H | H | H | H | H | H | H

Zeit in ns

Cache-Eviction-Strategien: Illustration



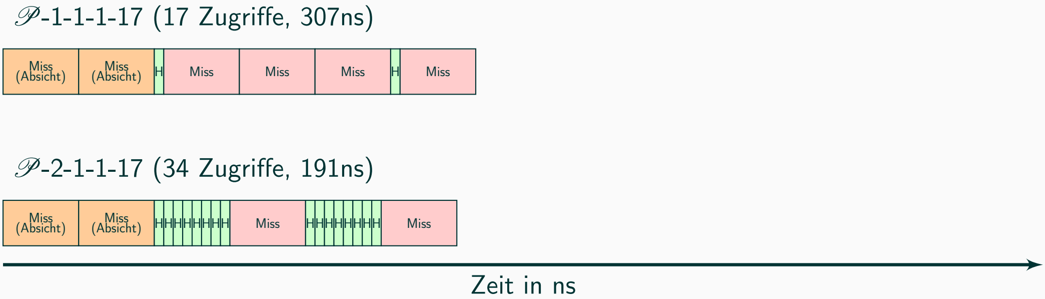
$\mathscr{P}$-1-1-1-17 (17 Zugriffe, 307ns)

Miss (Absicht)
Miss (Absicht)
H
Miss
Miss
Miss
H

$\mathscr{P}$-2-1-1-17 (34 Zugriffe, 191ns)

Miss (Absicht)
Miss (Absicht)
HHHHHHHH
Miss
HHHHHHHH
Miss
Zeit in ns

Cache-Eviction-Strategien: Illustration



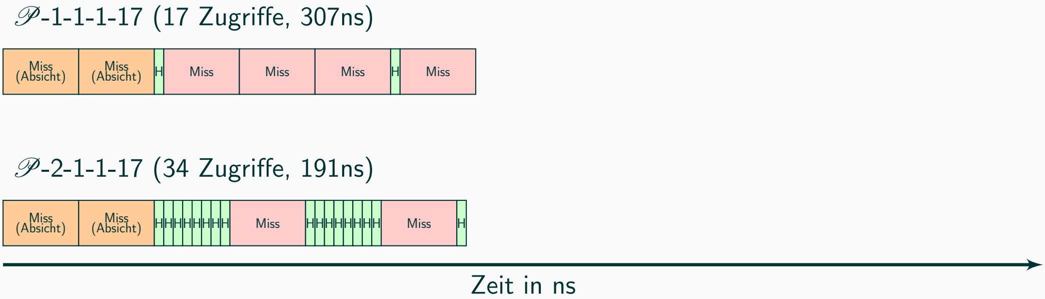
$\mathscr{P}$-1-1-1-17 (17 Zugriffe, 307ns)

Miss (Absicht)
Miss (Absicht)
H
Miss
Miss
Miss
H
Miss

$\mathscr{P}$-2-1-1-17 (34 Zugriffe, 191ns)

Miss (Absicht)
Miss (Absicht)
H H H H H H H H
Miss
H H H H H H H H
Miss
Zeit in ns

Cache-Eviction-Strategien: Illustration



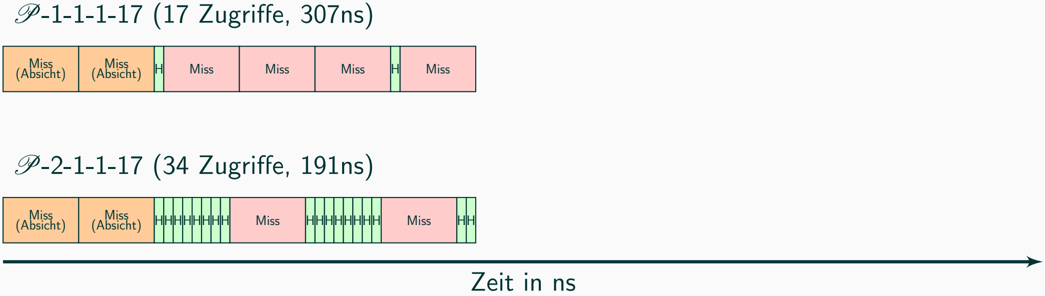
$\mathscr{P}$-1-1-1-17 (17 Zugriffe, 307ns)

Miss (Absicht)
Miss (Absicht)
H
Miss
Miss
Miss
H
Miss

$\mathscr{P}$-2-1-1-17 (34 Zugriffe, 191ns)

Miss (Absicht)
Miss (Absicht)
HHHHHHHH
Miss
HHHHHHHH
Miss
H
Zeit in ns

Cache-Eviction-Strategien: Illustration



$\mathscr{P}$-1-1-1-17 (17 Zugriffe, 307ns)

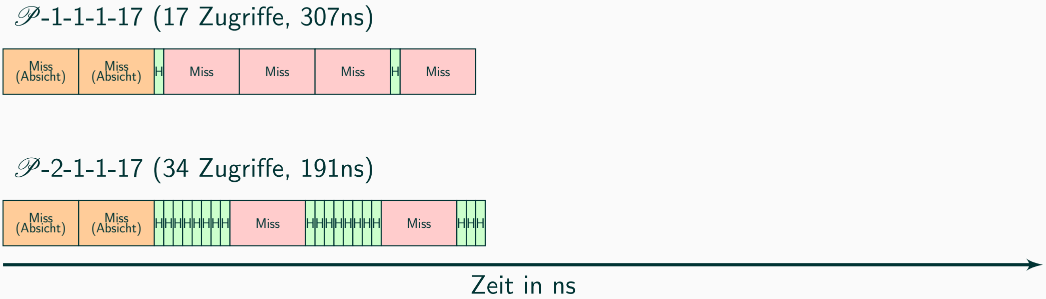
Miss (Absicht)
Miss (Absicht)
H
Miss
Miss
Miss
H
Miss

$\mathscr{P}$-2-1-1-17 (34 Zugriffe, 191ns)

Miss (Absicht)
Miss (Absicht)
HHHHHHHH
Miss
HHHHHHHH
Miss
HH

Zeit in ns

Cache-Eviction-Strategien: Illustration



$\mathscr{P}$-1-1-1-17 (17 Zugriffe, 307ns)

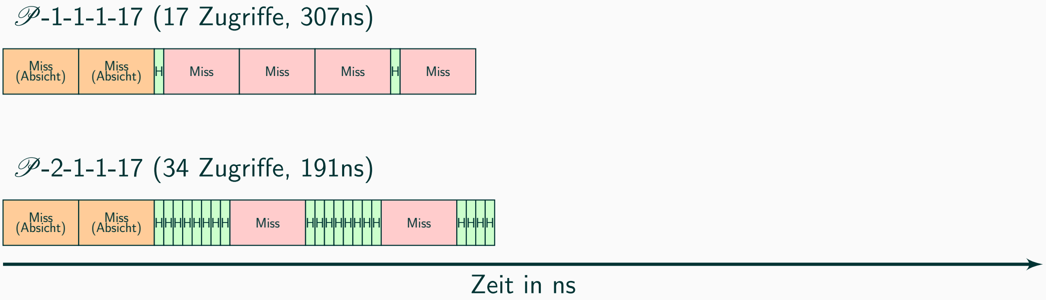
Miss (Absicht)
Miss (Absicht)
H
Miss
Miss
Miss
H
Miss

$\mathscr{P}$-2-1-1-17 (34 Zugriffe, 191ns)

Miss (Absicht)
Miss (Absicht)
HHHHHHHH
Miss
HHHHHHHH
Miss
HHH

Zeit in ns

Cache-Eviction-Strategien: Illustration



$\mathscr{P}$-1-1-1-17 (17 Zugriffe, 307ns)

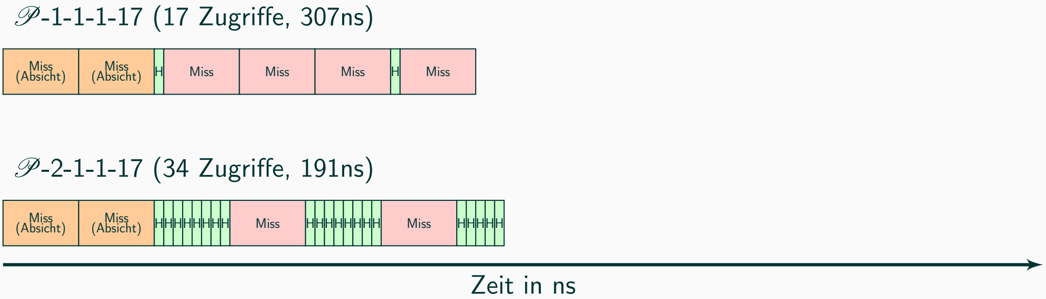
Miss (Absicht)
Miss (Absicht)
H
Miss
Miss
Miss
H
Miss

$\mathscr{P}$-2-1-1-17 (34 Zugriffe, 191ns)

Miss (Absicht)
Miss (Absicht)
HHHHHHHH
Miss
HHHHHHHH
Miss
HHHH

Zeit in ns

Cache-Eviction-Strategien: Illustration



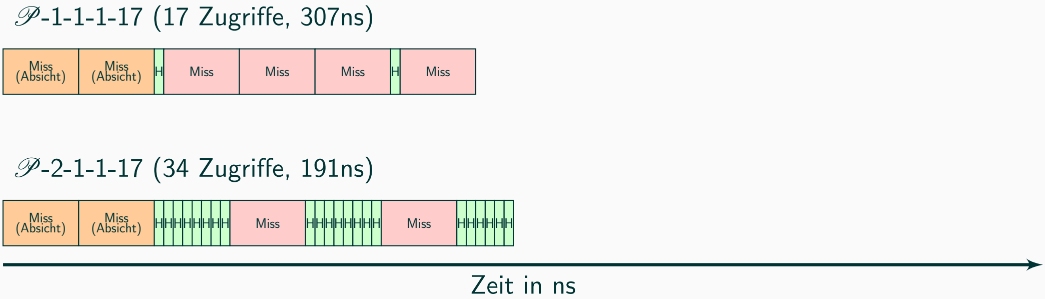
$\mathscr{P}$-1-1-1-17 (17 Zugriffe, 307ns)

Miss (Absicht)
Miss (Absicht)
H
Miss
Miss
Miss
H
Miss

$\mathscr{P}$-2-1-1-17 (34 Zugriffe, 191ns)

Miss (Absicht)
Miss (Absicht)
HHHHHHHH
Miss
HHHHHHHH
Miss
HHHHH
Zeit in ns

Cache-Eviction-Strategien: Illustration



$\mathscr{P}$-1-1-1-17 (17 Zugriffe, 307ns)

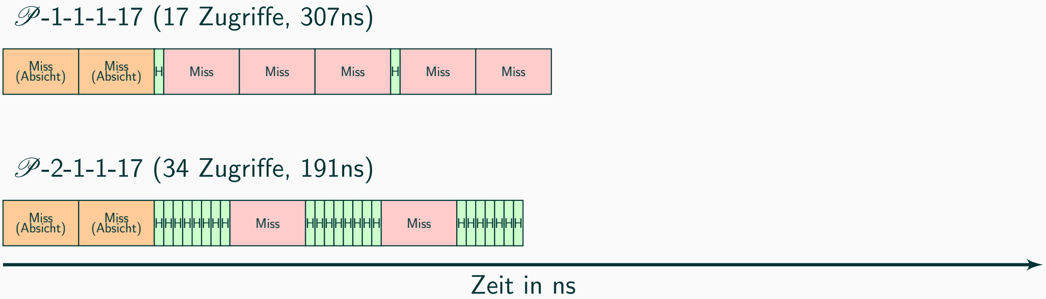
Miss (Absicht) | Miss (Absicht) | H | Miss | Miss | Miss | H | Miss

$\mathscr{P}$-2-1-1-17 (34 Zugriffe, 191ns)

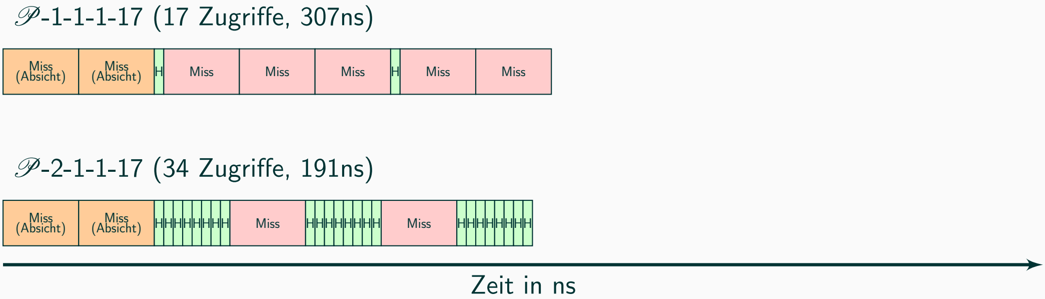
Miss (Absicht) | Miss (Absicht) | H H H H H H H H | Miss | H H H H H H H H | Miss | H H H H H H

Zeit in ns

Cache-Eviction-Strategien: Illustration


𝒫-1-1-1-17 (17 Zugriffe, 307ns)
Miss (Absicht)
Miss (Absicht)
H
Miss
Miss
Miss
H
Miss
Miss
𝒫-2-1-1-17 (34 Zugriffe, 191ns)
Miss (Absicht)
Miss (Absicht)
HHHHHHHH
Miss
HHHHHHHH
Miss
HHHHHHH
Zeit in ns

Cache-Eviction-Strategien: Illustration



$\mathscr{P}$-1-1-1-17 (17 Zugriffe, 307ns)

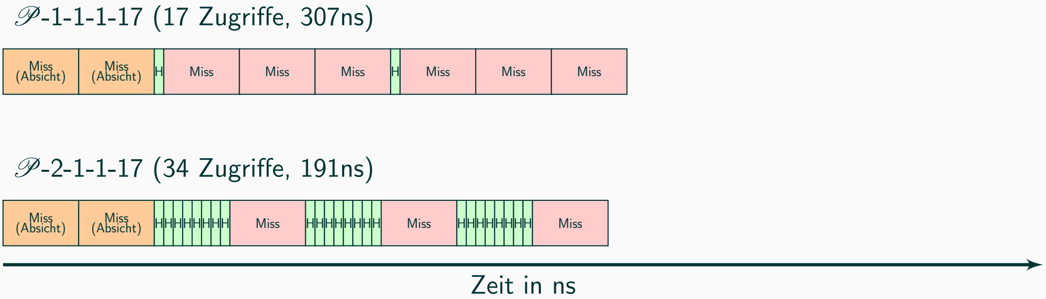
Miss (Absicht) | Miss (Absicht) | H | Miss | Miss | Miss | H | Miss | Miss

$\mathscr{P}$-2-1-1-17 (34 Zugriffe, 191ns)

Miss (Absicht) | Miss (Absicht) | H H H H H H H H | Miss | H H H H H H H H | Miss | H H H H H H H H

Zeit in ns

Cache-Eviction-Strategien: Illustration

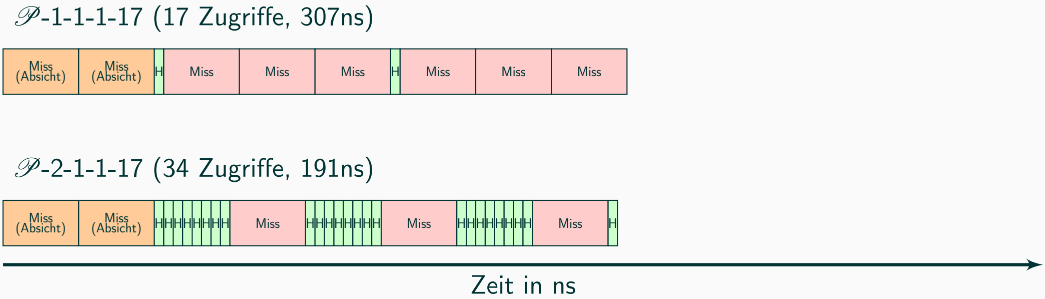


$\mathscr{P}$-1-1-1-17 (17 Zugriffe, 307ns)

$\mathscr{P}$-2-1-1-17 (34 Zugriffe, 191ns)

Miss (Absicht)
Miss (Absicht)
H
Miss
Miss
Miss
H
Miss
Miss
Miss
Miss (Absicht)
Miss (Absicht)
HHHHHHHH
Miss
HHHHHHHH
Miss
HHHHHHHH
Miss
Zeit in ns

Cache-Eviction-Strategien: Illustration



$\mathscr{P}$-1-1-1-17 (17 Zugriffe, 307ns)

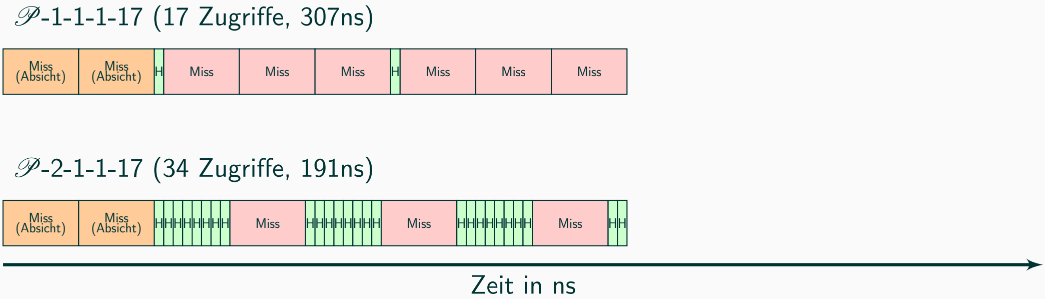
Miss (Absicht) | Miss (Absicht) | H | Miss | Miss | Miss | H | Miss | Miss | Miss

$\mathscr{P}$-2-1-1-17 (34 Zugriffe, 191ns)

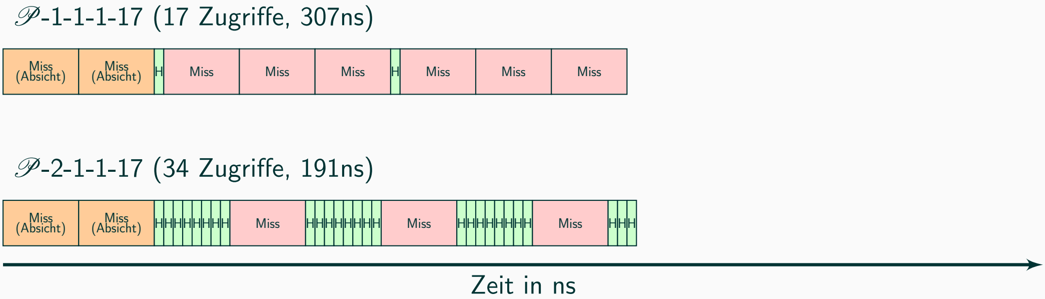
Miss (Absicht) | Miss (Absicht) | HHHHHHHH | Miss | HHHHHHHH | Miss | HHHHHHHH | Miss | H

Zeit in ns

Cache-Eviction-Strategien: Illustration


$\mathscr{P}$-1-1-1-17 (17 Zugriffe, 307ns)
Miss (Absicht)
Miss (Absicht)
H
Miss
Miss
Miss
H
Miss
Miss
Miss
$\mathscr{P}$-2-1-1-17 (34 Zugriffe, 191ns)
Miss (Absicht)
Miss (Absicht)
HHHHHHHH
Miss
HHHHHHHH
Miss
HHHHHHHH
Miss
HH
Zeit in ns

Cache-Eviction-Strategien: Illustration



$\mathscr{P}$-1-1-1-17 (17 Zugriffe, 307ns)

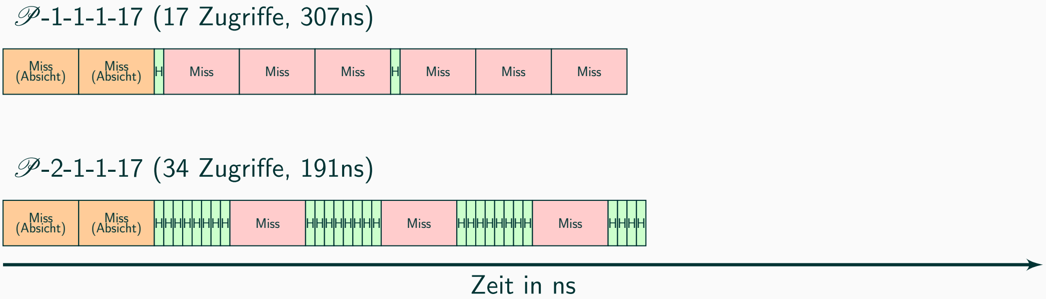
Miss (Absicht) | Miss (Absicht) | H | Miss | Miss | Miss | H | Miss | Miss | Miss

$\mathscr{P}$-2-1-1-17 (34 Zugriffe, 191ns)

Miss (Absicht) | Miss (Absicht) | H | H | H | H | H | H | H | H | Miss | H | H | H | H | H | H | H | H | Miss | H | H | H | H | H | H | H | H | Miss | H | H | H

Zeit in ns

Cache-Eviction-Strategien: Illustration



𝒫-1-1-1-17 (17 Zugriffe, 307ns)

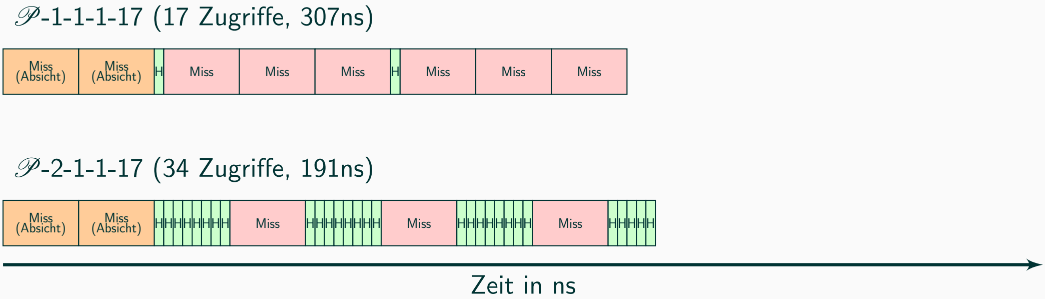
Miss (Absicht) | Miss (Absicht) | H | Miss | Miss | Miss | H | Miss | Miss | Miss

𝒫-2-1-1-17 (34 Zugriffe, 191ns)

Miss (Absicht) | Miss (Absicht) | H | H | H | H | H | H | H | H | Miss | H | H | H | H | H | H | H | H | Miss | H | H | H | H | H | H | H | H | Miss | H | H | H | H

Zeit in ns

Cache-Eviction-Strategien: Illustration



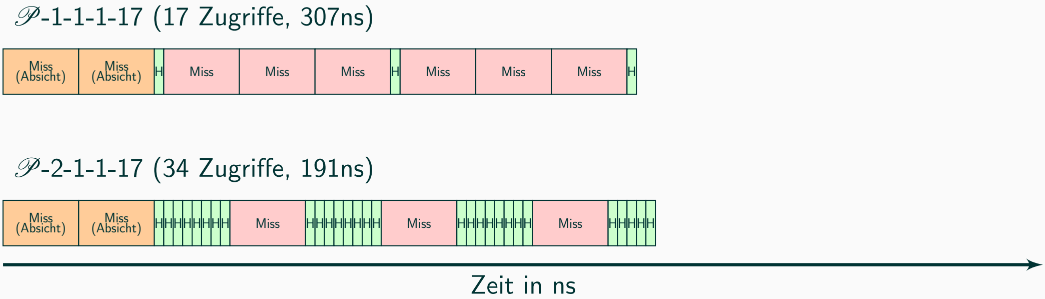
$\mathscr{P}$-1-1-1-17 (17 Zugriffe, 307ns)

Miss (Absicht)
Miss (Absicht)
H
Miss
Miss
Miss
H
Miss
Miss
Miss

$\mathscr{P}$-2-1-1-17 (34 Zugriffe, 191ns)

Miss (Absicht)
Miss (Absicht)
HHHHHHHH
Miss
HHHHHHHH
Miss
HHHHHHHH
Miss
HHHHH
Zeit in ns

Cache-Eviction-Strategien: Illustration



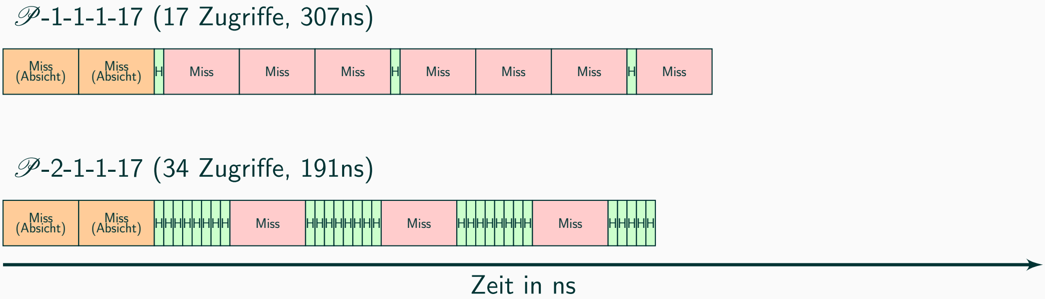
$\mathscr{P}$-1-1-1-17 (17 Zugriffe, 307ns)

Miss (Absicht)
Miss (Absicht)
H
Miss
Miss
Miss
H
Miss
Miss
Miss
H

$\mathscr{P}$-2-1-1-17 (34 Zugriffe, 191ns)

Miss (Absicht)
Miss (Absicht)
HHHHHHHH
Miss
HHHHHHHH
Miss
HHHHHHHH
Miss
HHHHH
Zeit in ns

Cache-Eviction-Strategien: Illustration



$\mathscr{P}$-1-1-1-17 (17 Zugriffe, 307ns)

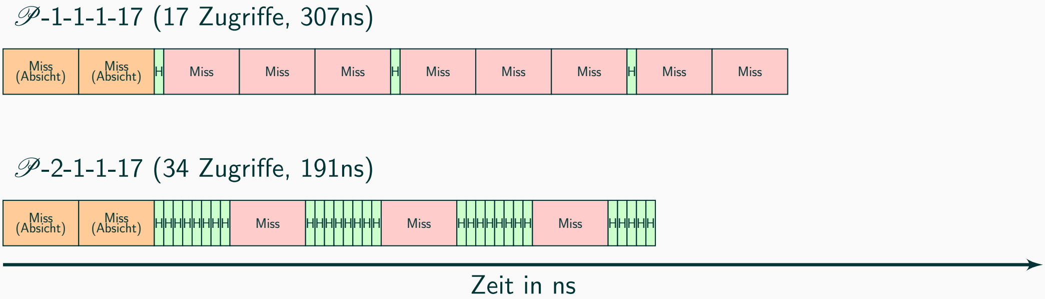
Miss (Absicht) | Miss (Absicht) | H | Miss | Miss | Miss | H | Miss | Miss | Miss | H | Miss

$\mathscr{P}$-2-1-1-17 (34 Zugriffe, 191ns)

Miss (Absicht) | Miss (Absicht) | H H H H H H H H | Miss | H H H H H H H H | Miss | H H H H H H H H | Miss | H H H H H

Zeit in ns

Cache-Eviction-Strategien: Illustration



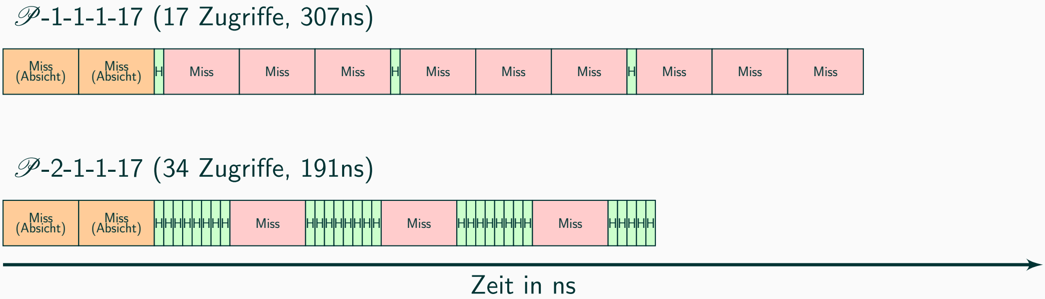
𝒫-1-1-1-17 (17 Zugriffe, 307ns)

Miss (Absicht)
Miss (Absicht)
H
Miss
Miss
Miss
H
Miss
Miss
Miss
H
Miss
Miss

𝒫-2-1-1-17 (34 Zugriffe, 191ns)

Miss (Absicht)
Miss (Absicht)
HHHHHHHH
Miss
HHHHHHHH
Miss
HHHHHHHH
Miss
HHHHH
Zeit in ns

Cache-Eviction-Strategien: Illustration



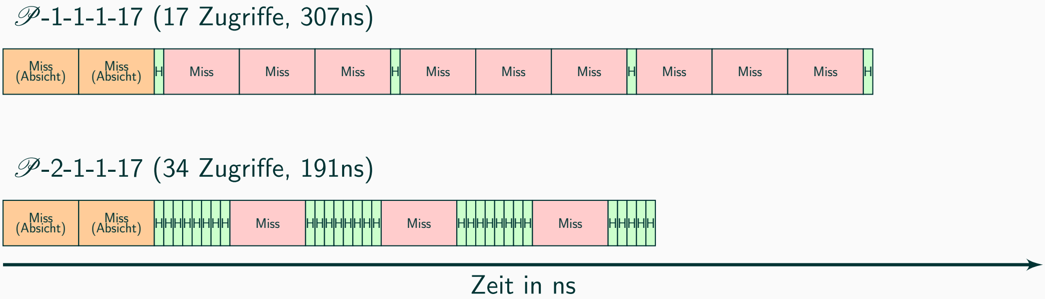
$\mathscr{P}$-1-1-1-17 (17 Zugriffe, 307ns)

Miss (Absicht)
Miss (Absicht)
H
Miss
Miss
Miss
H
Miss
Miss
Miss
H
Miss
Miss
Miss

$\mathscr{P}$-2-1-1-17 (34 Zugriffe, 191ns)

Miss (Absicht)
Miss (Absicht)
HHHHHHHH
Miss
HHHHHHHH
Miss
HHHHHHHH
Miss
HHHHH
Zeit in ns

Cache-Eviction-Strategien: Illustration



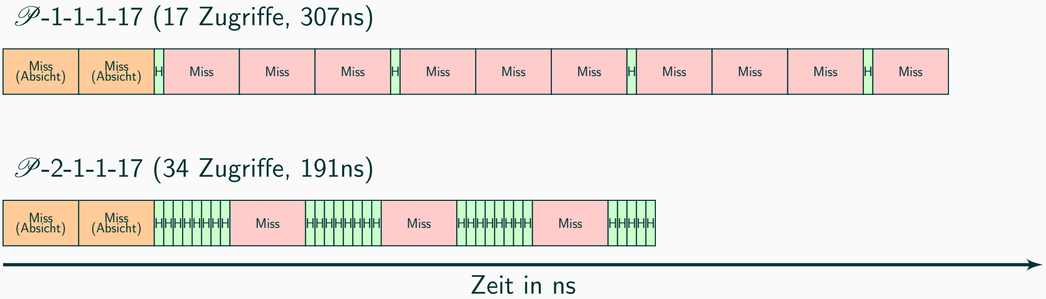
$\mathscr{P}$-1-1-1-17 (17 Zugriffe, 307ns)

Miss (Absicht)
Miss (Absicht)
H
Miss
Miss
Miss
H
Miss
Miss
Miss
H
Miss
Miss
Miss
H

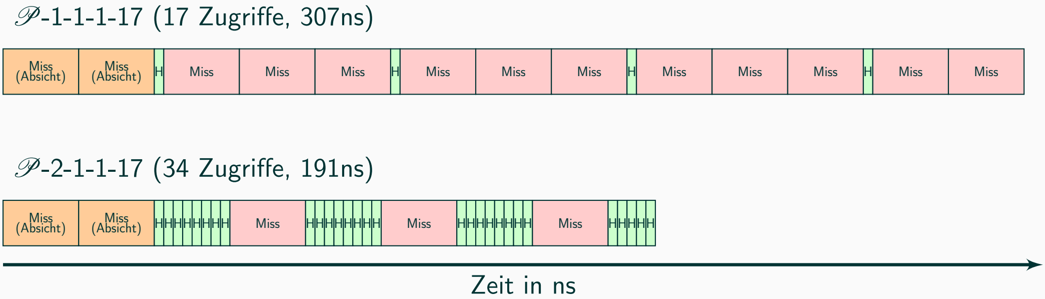
$\mathscr{P}$-2-1-1-17 (34 Zugriffe, 191ns)

Miss (Absicht)
Miss (Absicht)
HHHHHHHH
Miss
HHHHHHHH
Miss
HHHHHHHH
Miss
HHHHH
Zeit in ns

Cache-Eviction-Strategien: Illustration


𝒫-1-1-1-17 (17 Zugriffe, 307ns)
Miss (Absicht)
Miss (Absicht)
H
Miss
Miss
Miss
H
Miss
Miss
Miss
H
Miss
Miss
Miss
H
Miss
𝒫-2-1-1-17 (34 Zugriffe, 191ns)
Miss (Absicht)
Miss (Absicht)
HHHHHHHH
Miss
HHHHHHHH
Miss
HHHHHHHH
Miss
HHHHH
Zeit in ns

Cache-Eviction-Strategien: Illustration



$\mathscr{P}$-1-1-1-17 (17 Zugriffe, 307ns)

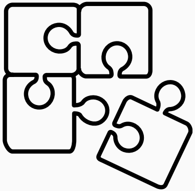
Miss (Absicht)
Miss (Absicht)
H
Miss
Miss
Miss
H
Miss
Miss
Miss
H
Miss
Miss
Miss
H
Miss
Miss

$\mathscr{P}$-2-1-1-17 (34 Zugriffe, 191ns)

Miss (Absicht)
Miss (Absicht)
HHHHHHHH
Miss
HHHHHHHH
Miss
HHHHHHHH
Miss
HHHHH

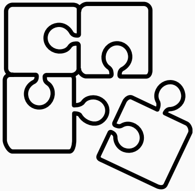
Zeit in ns

Rowhammer Anforderungen



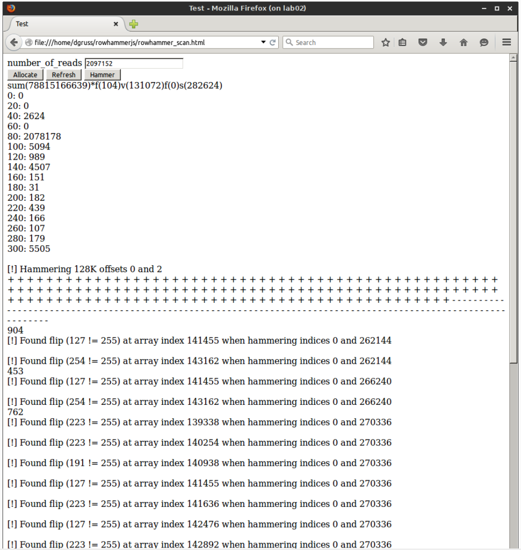
- ✓ Genaue Zeitmessungen (in JS)
- ✓ Physikalische Adressen (in JS)
- ✓ Adressauswahl
- ? Zugriffsreihenfolge

Rowhammer Anforderungen



- ✓ Genaue Zeitmessungen (in JS)
- ✓ Physikalische Adressen (in JS)
- ✓ Adressauswahl
- ✓ Zugriffsreihenfolge

Bit-Flips in JavaScript


Test - Mozilla Firefox (on lab02)
Test
file:///home/dgruss/rowhammerjs/rowhammer_scan.html
Search

number_of_reads 2097152

Allocate | Refresh | Hammer

```
sum(78815166639)*f(104)v(131072)f(0)s(282624)
0: 0
20: 0
40: 2624
60: 0
80: 2078178
100: 5094
120: 989
140: 4507
160: 151
180: 31
200: 182
220: 439
240: 166
260: 107
280: 179
300: 5505

[!] Hammering 128K offsets 0 and 2
+ + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + +
+ + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + +
+ + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + - - - - - - - - - - -
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
- - - - - - - - -
904
[!] Found flip (127 != 255) at array index 141455 when hammering indices 0 and 262144

[!] Found flip (254 != 255) at array index 143162 when hammering indices 0 and 262144
453
[!] Found flip (127 != 255) at array index 141455 when hammering indices 0 and 266240

[!] Found flip (254 != 255) at array index 143162 when hammering indices 0 and 266240
762
[!] Found flip (223 != 255) at array index 139338 when hammering indices 0 and 270336

[!] Found flip (223 != 255) at array index 140254 when hammering indices 0 and 270336

[!] Found flip (191 != 255) at array index 140938 when hammering indices 0 and 270336

[!] Found flip (127 != 255) at array index 141455 when hammering indices 0 and 270336

[!] Found flip (223 != 255) at array index 141636 when hammering indices 0 and 270336

[!] Found flip (127 != 255) at array index 142476 when hammering indices 0 and 270336

[!] Found flip (223 != 255) at array index 142892 when hammering indices 0 and 270336
```

Evaluierung auf Haswell

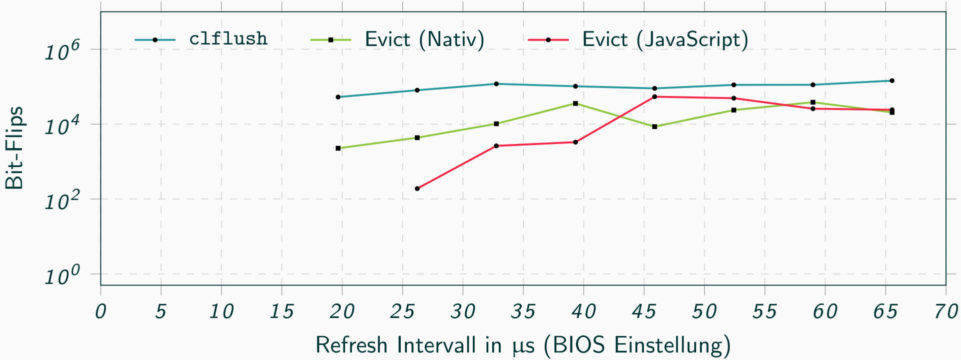


Abbildung 1: Anzahl der Bit-Flips nach 15 Minuten.

Flush+Flush

Flush+Flush: Motivation

Flush+Flush: Motivation

- Cache Angriffe → viele Cache Misses

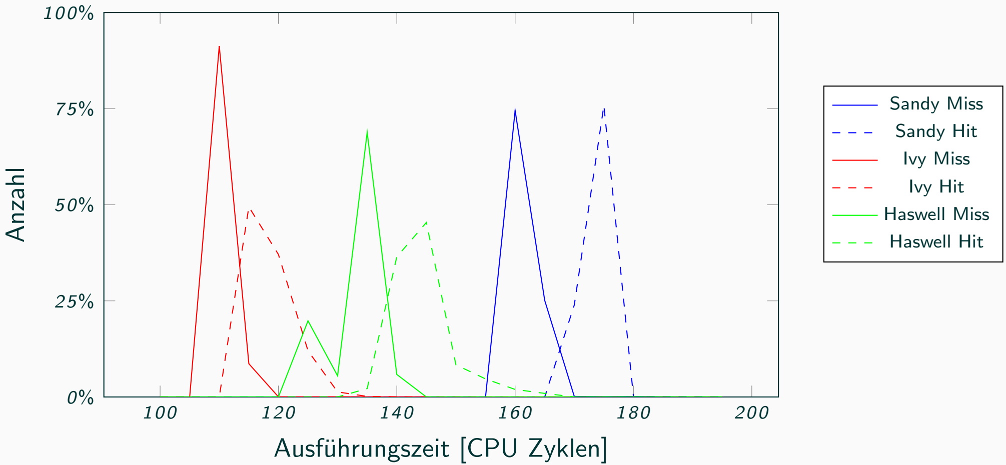
Flush+Flush: Motivation

- Cache Angriffe → viele Cache Misses
- Idee: Cache Misses mit Performance Counters detektieren

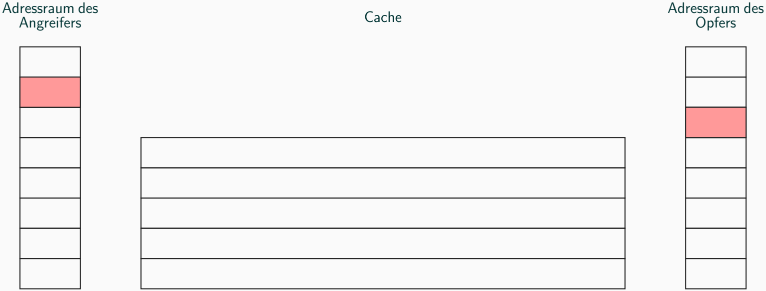
Flush+Flush: Motivation

- Cache Angriffe → viele Cache Misses
- Idee: Cache Misses mit Performance Counters detektieren
- Problem: Cache Flush $\neq$ Cache Miss

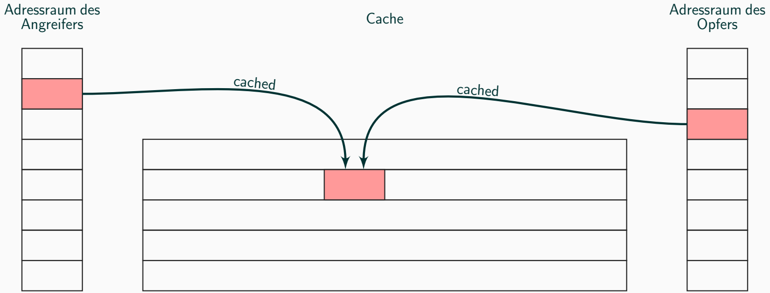
Flush+Flush Histogramm



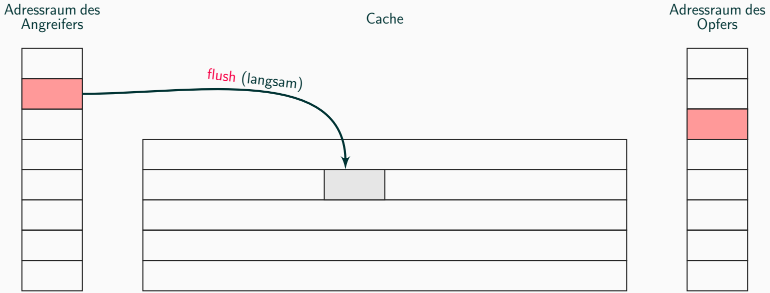
Flush+Flush


Adressraum des Angreifers
Cache
Adressraum des Opfers

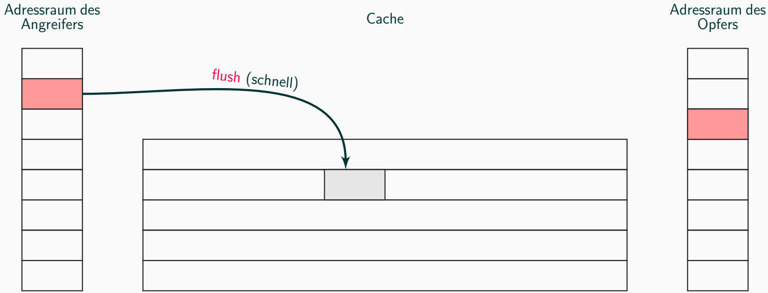
Flush+Flush


Adressraum des Angreifers
Cache
Adressraum des Opfers
cached
cached

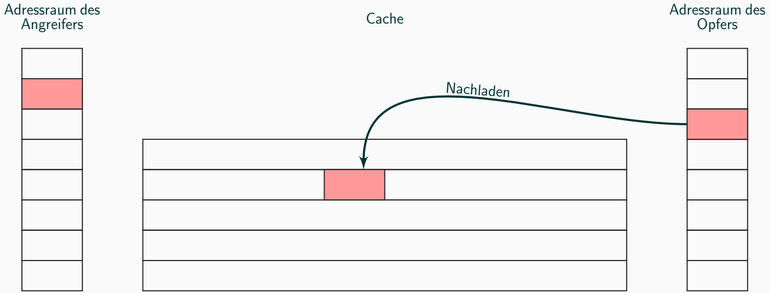
Flush+Flush


Adressraum des Angreifers
Cache
Adressraum des Opfers
flush (langsam)

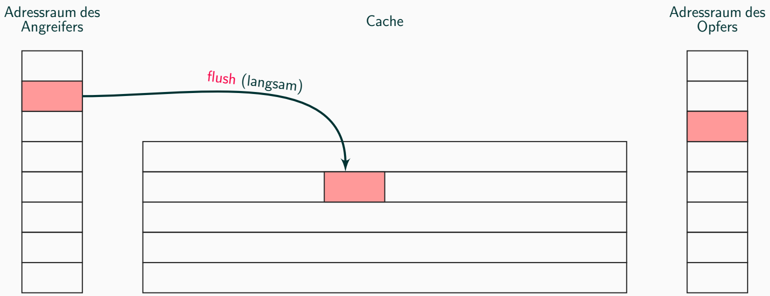
Flush+Flush


Adressraum des Angreifers
Cache
Adressraum des Opfers
flush (schnell)

Flush+Flush


Adressraum des Angreifers
Cache
Adressraum des Opfers
Nachladen

Flush+Flush


Adressraum des Angreifers
Cache
Adressraum des Opfers
flush (langsam)

Flush+Flush: Zusammenfassung

Flush+Flush: Zusammenfassung

- Der Angreifer verursacht selbst keine Cache Misses

Flush+Flush: Zusammenfassung

- Der Angreifer verursacht selbst keine Cache Misses

  → Schnell

Flush+Flush: Zusammenfassung

- Der Angreifer verursacht selbst keine Cache Misses

  → Schnell (496 KB/s Covert Channel)

Flush+Flush: Zusammenfassung

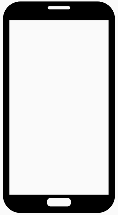
- Der Angreifer verursacht selbst keine Cache Misses
  - → Schnell (496 KB/s Covert Channel)
  - → Schwer zu detektieren

Flush+Flush: Zusammenfassung

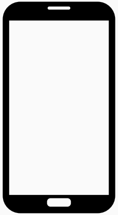
- Der Angreifer verursacht selbst keine Cache Misses
  - → Schnell (496 KB/s Covert Channel)
  - → Schwer zu detektieren
- Gleiche Seitenkanalangriffe möglich wie mit Flush+Reload

ARMageddon

Cache Angriffe auf Mobilgeräten?

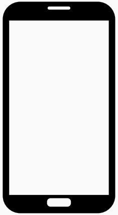


Cache Angriffe auf Mobilgeräten?



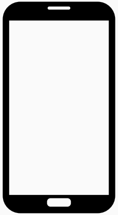
- Viele Cache Angriffe auf Intel x86 Prozessoren in den vergangenen 10 Jahren

Cache Angriffe auf Mobilgeräten?



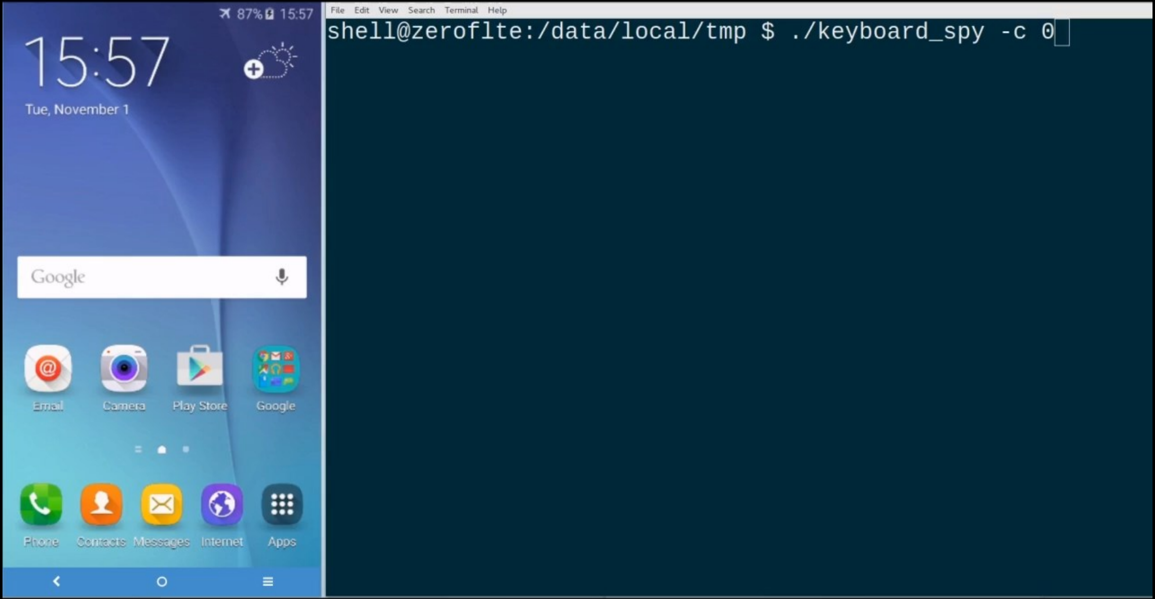
- Viele Cache Angriffe auf Intel x86 Prozessoren in den vergangenen 10 Jahren
- Aber nichts von einer Bedeutung wie Flush+Reload oder Prime+Probe auf Mobilgeräten

Cache Angriffe auf Mobilgeräten?



- Viele Cache Angriffe auf Intel x86 Prozessoren in den vergangenen 10 Jahren
- Aber nichts von einer Bedeutung wie Flush+Reload oder Prime+Probe auf Mobilgeräten

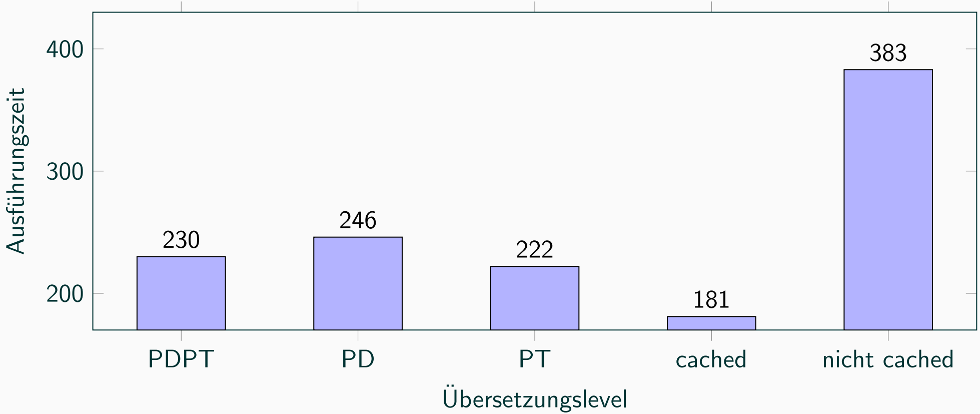
→ Warum nicht?


87% 15:57
15:57
Tue, November 1
Google
Email
Camera
Play Store
Google
Phone
Contacts
Messages
Internet
Apps
File
Edit
View
Search
Terminal
Help

```
shell@zeroflte:/data/local/tmp $ ./keyboard_spy -c 0
```

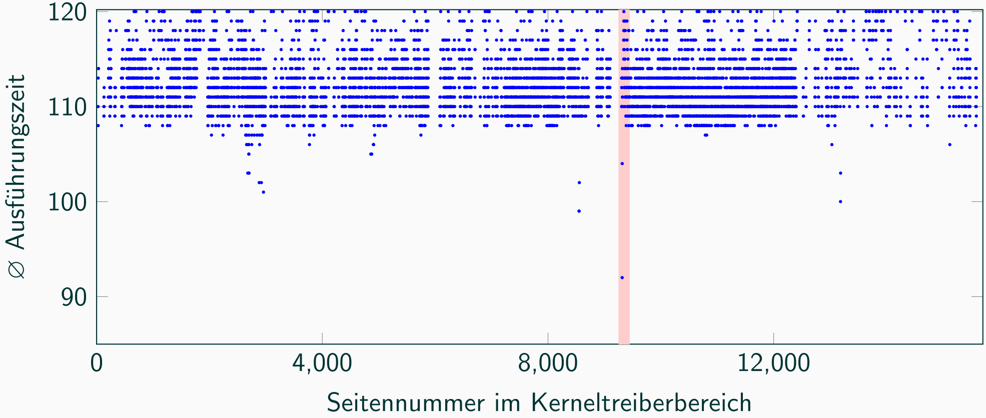
Prefetch-Seitenkanalangriff

Prefetch: Motivation



Idee: Würde das auch auf nicht-zugreifbaren Kernelspeicher funktionieren?

Prefetch: ASLR umgehen (Adresse eines Kerneltreibers finden)



Prefetch-Seitenkanal und Meltdown

Prefetch-Seitenkanal und Meltdown

- Der gleiche zugrundeliegende Fehler

Prefetch-Seitenkanal und Meltdown

- Der gleiche zugrundeliegende Fehler
- Prefetch-Instruktion verändert keine Register

Prefetch-Seitenkanal und Meltdown

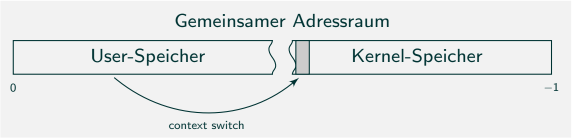
- Der gleiche zugrundeliegende Fehler
- Prefetch-Instruktion verändert keine Register
- Meltdown liest geheimen Wert in Register

Prefetch-Seitenkanal und Meltdown

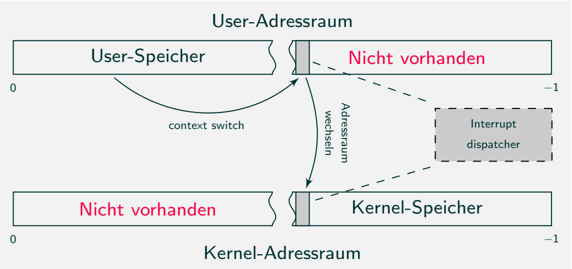
- Der gleiche zugrundeliegende Fehler
- Prefetch-Instruktion verändert keine Register
- Meltdown liest geheimen Wert in Register
- Gleiche Gegenmaßnahme: KAISER

KAISER (Stronger Kernel Isolation)

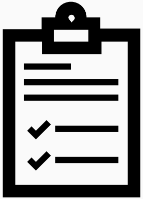
Betriebssysteme bisher:


Gemeinsamer Adressraum
User-Speicher
Kernel-Speicher
0
−1
context switch

KAISER (Stronger Kernel Isolation):


User-Adressraum
User-Speicher
Nicht vorhanden
0
−1
context switch
Adressraum wechseln
Interrupt dispatcher
Nicht vorhanden
Kernel-Speicher
0
−1
Kernel-Adressraum

Schlussfolgerungen



Schlussfolgerungen

1. Mikroarchitekturangriffe können weitgehend automatisiert werden

Schlussfolgerungen

1. Mikroarchitekturangriffe können weitgehend automatisiert werden
2. Noch viele unbekannte und neue Angriffe

Schlussfolgerungen

1. Mikroarchitekturangriffe können weitgehend automatisiert werden
2. Noch viele unbekannte und neue Angriffe (z.B. Meltdown und Spectre)

Schlussfolgerungen

1. Mikroarchitekturangriffe können weitgehend automatisiert werden
2. Noch viele unbekannte und neue Angriffe (z.B. Meltdown und Spectre)
3. Oft minimale Anforderungen → Angriffe durch Webseiten

Schlussfolgerungen

1. Mikroarchitekturangriffe können weitgehend automatisiert werden
2. Noch viele unbekannte und neue Angriffe (z.B. Meltdown und Spectre)
3. Oft minimale Anforderungen → Angriffe durch Webseiten
4. Gegenmaßnahmen zu konstruieren ist schwierig und erfordert ein solides Verständnis der Angriffe

Software-basierte Mikroarchitekturangriffe

Daniel Gruss

7.5.2018

Technische Universität Graz

Publikationen in dieser Dissertation

1. Daniel Gruss, Raphael Spreitzer und Stefan Mangard (2015). Cache Template Attacks: Automating Attacks on Inclusive Last-Level Caches. In: USENIX Security
2. Daniel Gruss, David Bidner und Stefan Mangard (2015). Practical Memory Deduplication Attacks in Sandboxed JavaScript. In: ESORICS'15
3. Daniel Gruss, Clémentine Maurice, Klaus Wagner und Stefan Mangard (2016). Flush+Flush: A Fast and Stealthy Cache Attack. In: DIMVA'16
4. Daniel Gruss, Clémentine Maurice und Stefan Mangard (2016). Rowhammer.js: A Remote Software-Induced Fault Attack in JavaScript. In: DIMVA'16
5. Moritz Lipp, Daniel Gruss, Raphael Spreitzer, Clémentine Maurice und Stefan Mangard (2016). ARMageddon: Cache Attacks on Mobile Devices. In: USENIX Security
6. Daniel Gruss, Clémentine Maurice, Anders Fogh, Moritz Lipp und Stefan Mangard (2016). Prefetch Side-Channel Attacks: Bypassing SMAP and Kernel ASLR. In: CCS'16

Weitere Publikationen im Dissertationszeitraum i

1. Peter Pessl, Daniel Gruss, Clémentine Maurice, Michael Schwarz und Stefan Mangard (2016). DRAMA: Exploiting DRAM Addressing for Cross-CPU Attacks. In: USENIX Security
2. Victor van der Veen, Yanick Fratantonio, Martina Lindorfer, Daniel Gruss, Clémentine Maurice, Giovanni Vigna, Herbert Bos, Kaveh Razavi und Cristiano Giuffrida (2016). Drammer: Deterministic Rowhammer Attacks on Mobile Platforms. In: CCS'16
3. Clémentine Maurice, Manuel Weber, Michael Schwarz, Lukas Giner, Daniel Gruss, Carlo Alberto Boano, Stefan Mangard und Kay Römer (2017). Hello from the Other Side: SSH over Robust Cache Covert Channels in the Cloud. In: NDSS'17
4. Michael Schwarz, Clémentine Maurice, Daniel Gruss und Stefan Mangard (2017). Fantastic Timers and Where to Find Them: High-Resolution Microarchitectural Attacks in JavaScript. In: Financial Cryptography 2017
5. Daniel Gruss, Moritz Lipp, Michael Schwarz, Richard Fellner, Clémentine Maurice und Stefan Mangard (2017). KASLR is Dead: Long Live KASLR. In: ESSoS'17
6. Michael Schwarz, Daniel Gruss, Samuel Weiser, Clémentine Maurice und Stefan Mangard (2017). Malware Guard Extension: Using SGX to Conceal Cache Attacks. In: DIMVA'17
7. Daniel Gruss, Julian Lettner, Felix Schuster, Olga Ohrimenko, Istvan Haller und Manuel Costa (2017). Strong and Efficient Cache Side-Channel Protection using Hardware Transactional Memory. In: USENIX Security