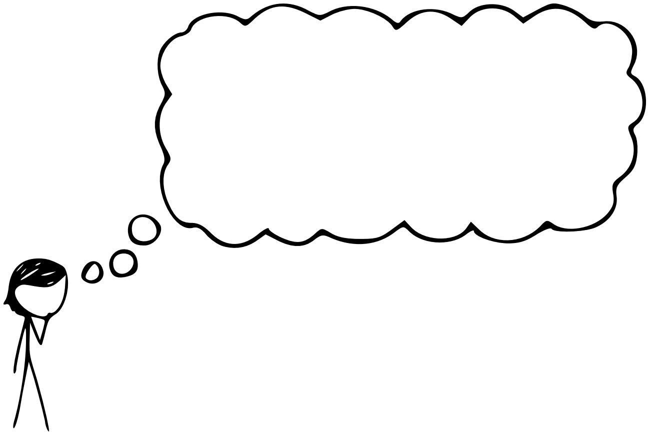
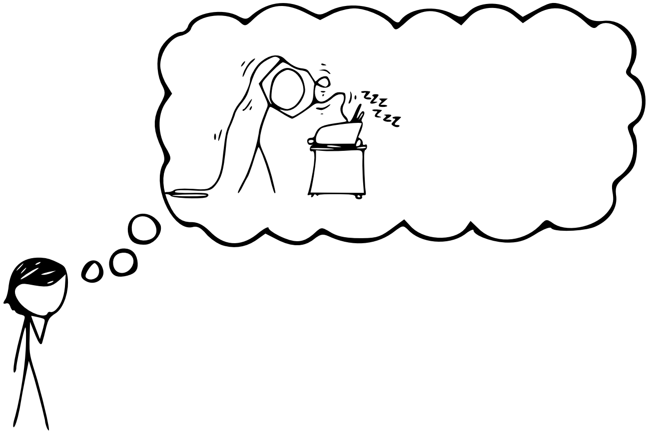
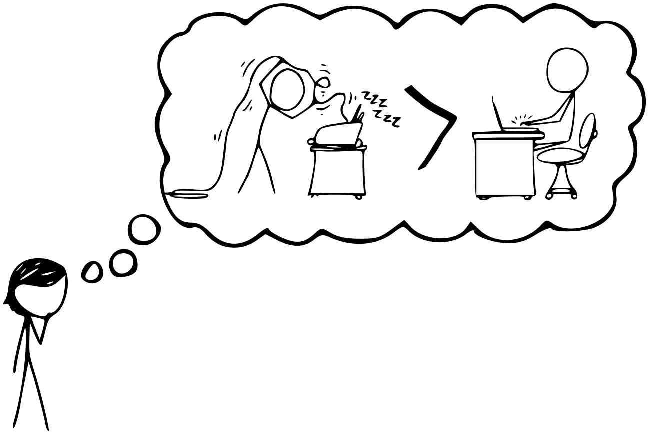
Software-based Side-Channel Attacks and Defenses in Restricted Environments

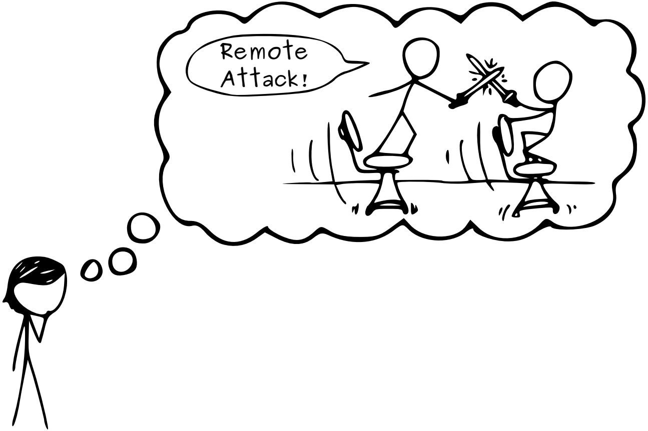
PhD Defense

Michael Schwarz
IAIK, Graz University of Technology
November 5, 2019




zzz
zzz

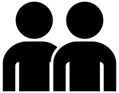

zzz
zzz



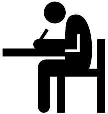
Research Question

How far can we reduce requirements?

Numbers



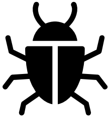
24
(**13** tier 1)



7
(**4** tier 1)

30

7



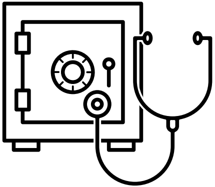
8

10
(**3** co-authors)


Side Channels

$$M \equiv C^d \pmod{n}$$

Description

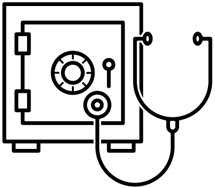


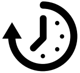
$$M \equiv C^d \pmod{n}$$

Description

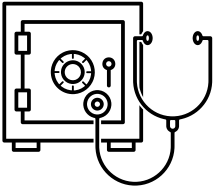
$$C^n = \begin{cases} (C^2)^{\frac{n}{2}} & \text{if } n \equiv 0 \mod 2 \\ C \cdot (C^2)^{\frac{n-1}{2}} & \text{if } n \equiv 1 \mod 2 \end{cases}$$

Software





Execution time



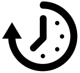
$$M \equiv C^d \pmod{n}$$

Description

$$C^n = \begin{cases} (C^2)^{\frac{n}{2}} & \text{if } n \equiv 0 \mod 2 \\ C \cdot (C^2)^{\frac{n-1}{2}} & \text{if } n \equiv 1 \mod 2 \end{cases}$$

Software

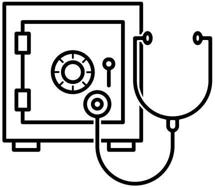
Hardware



Execution time

Power consumption

Side-channel Attacks



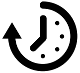
$$M \equiv C^d \pmod{n}$$

Description

$$C^n = \begin{cases} (C^2)^{\frac{n}{2}} & \text{if } n \equiv 0 \mod 2 \\ C \cdot (C^2)^{\frac{n-1}{2}} & \text{if } n \equiv 1 \mod 2 \end{cases}$$

Software

Hardware

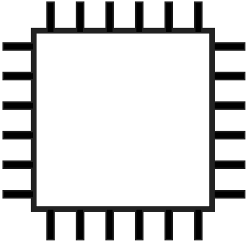


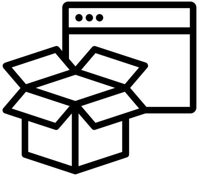
Execution time

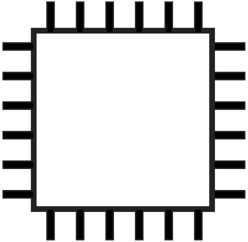
Power consumption

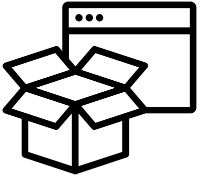
CPU caches

μ-arch

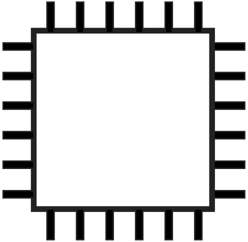


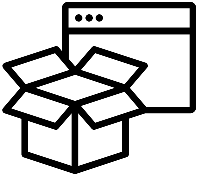




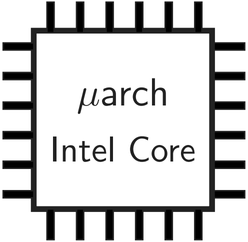



ISA
x86

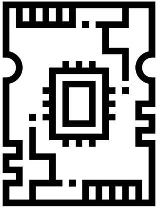




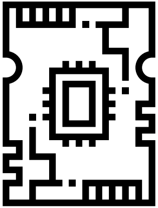

ISA
x86

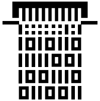

μarch
Intel Core

- Modern CPUs contain multiple microarchitectural elements



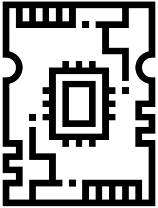
- Modern CPUs contain multiple microarchitectural elements

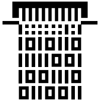




Decoder

- Modern CPUs contain multiple microarchitectural elements

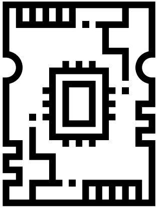


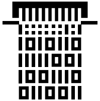


Decoder

Predictor

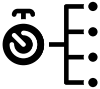
- Modern CPUs contain multiple microarchitectural elements





Decoder

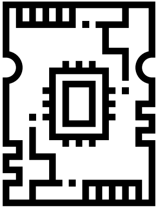
Predictor

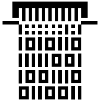


Scheduler

Microarchitectural Elements

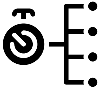
- Modern CPUs contain multiple microarchitectural elements



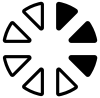


Decoder

Predictor

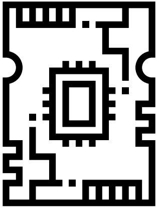


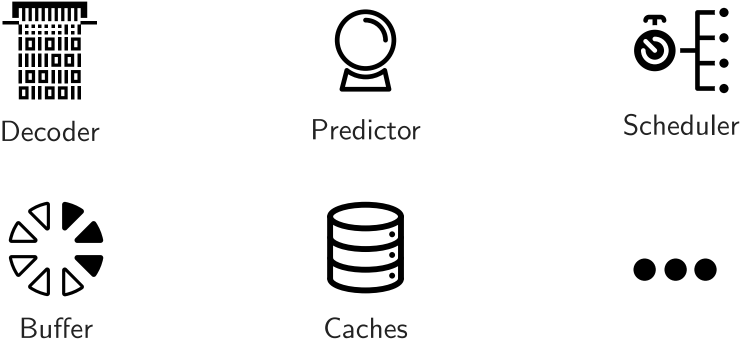
Scheduler

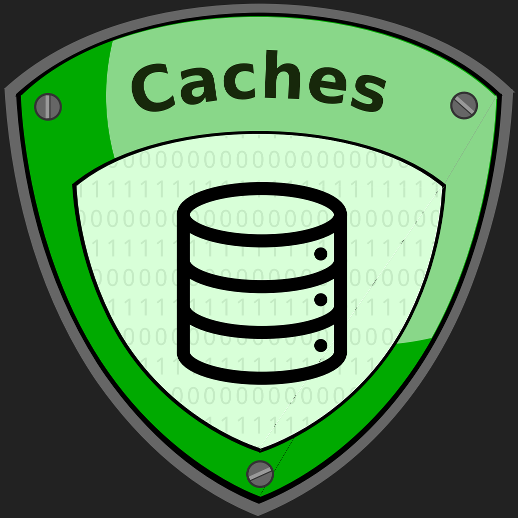


Buffer

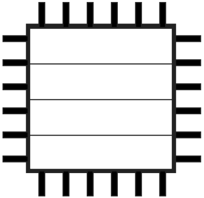
- Modern CPUs contain multiple microarchitectural elements

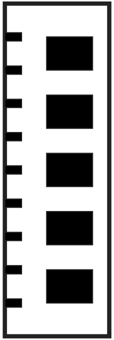




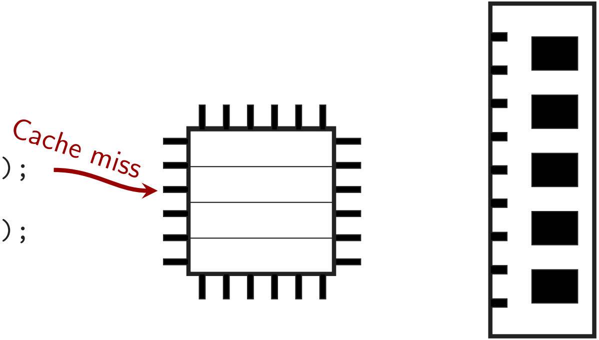

Caches

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

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



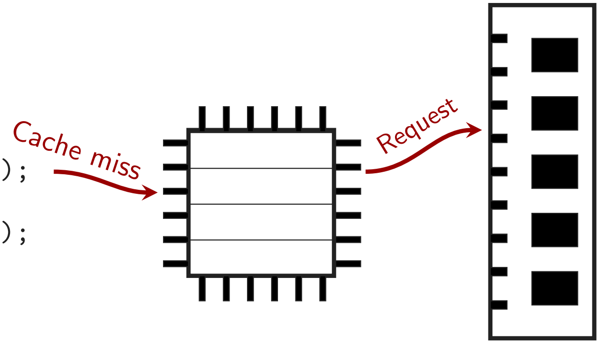


```
printf("%d", i);

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

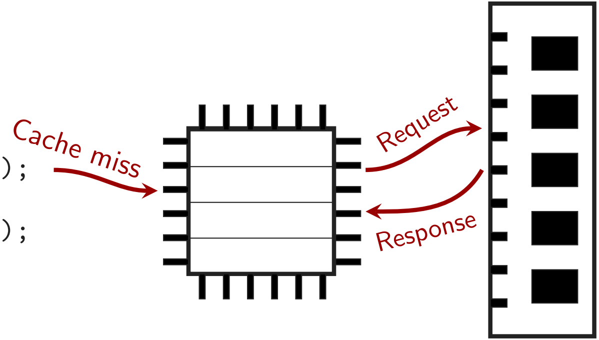


CPU Cache

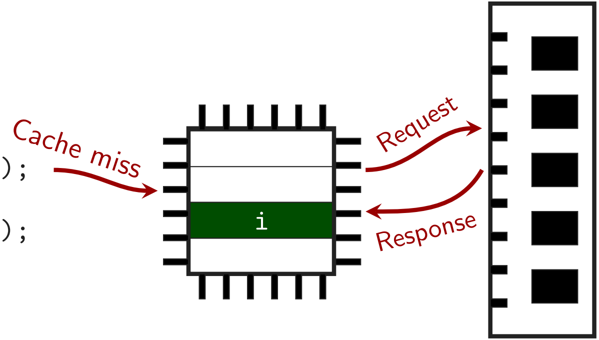
```
printf("%d", i);

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

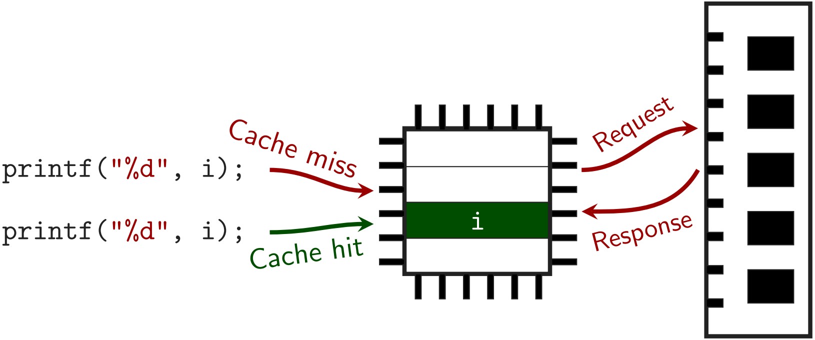


```
printf("%d", i);

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

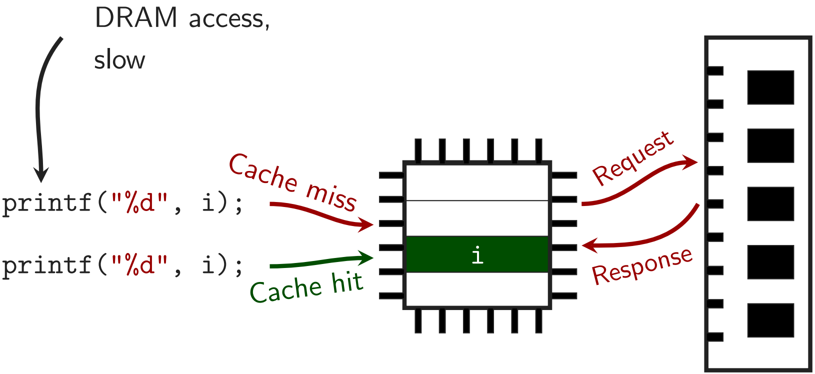


```
printf("%d", i);

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

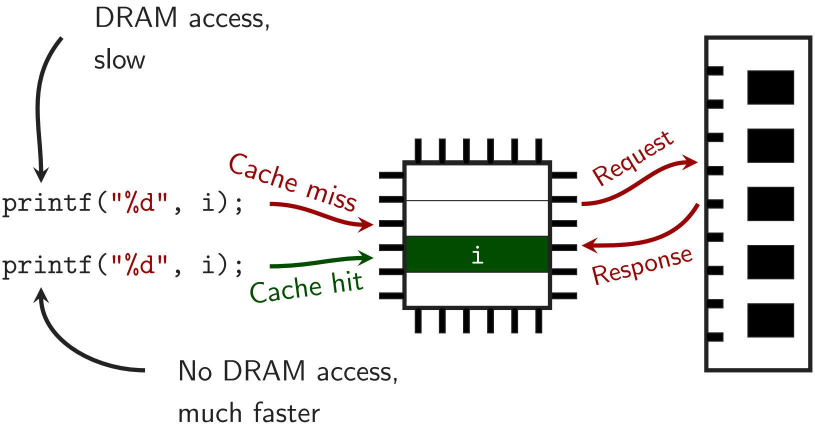




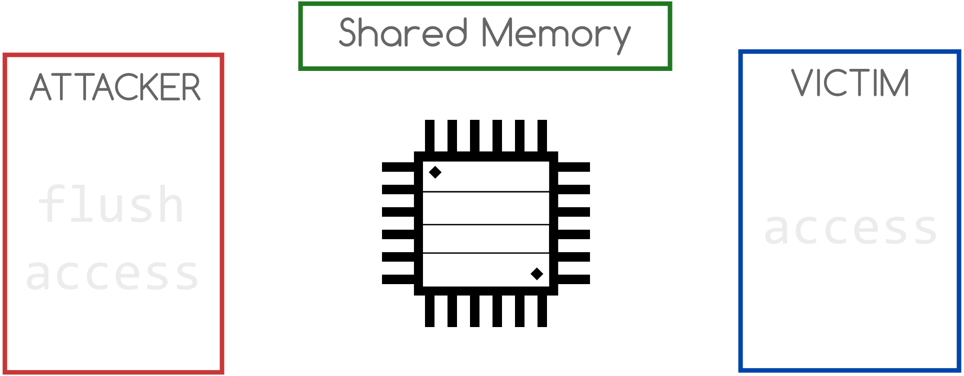
printf("%d", i);
printf("%d", i);
Cache miss
Cache hit
i
Request
Response

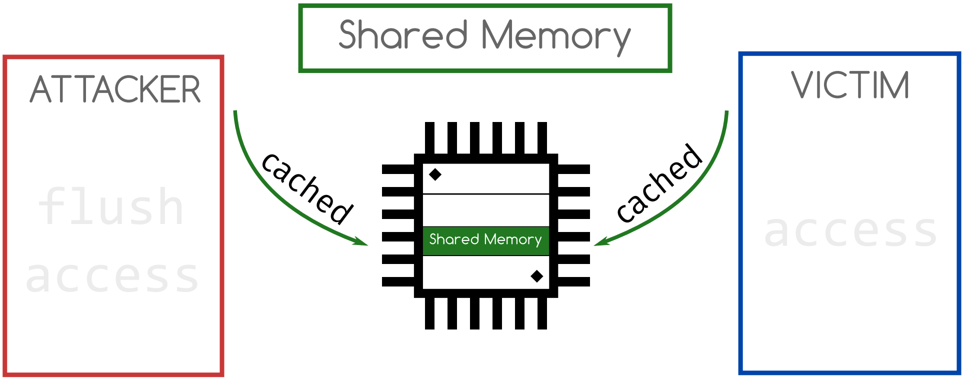


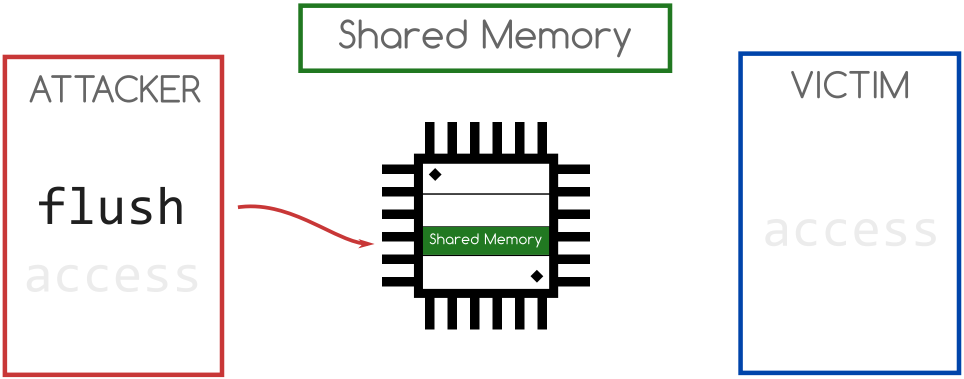
DRAM access,
slow
printf("%d", i);
printf("%d", i);
Cache miss
Cache hit
i
Request
Response

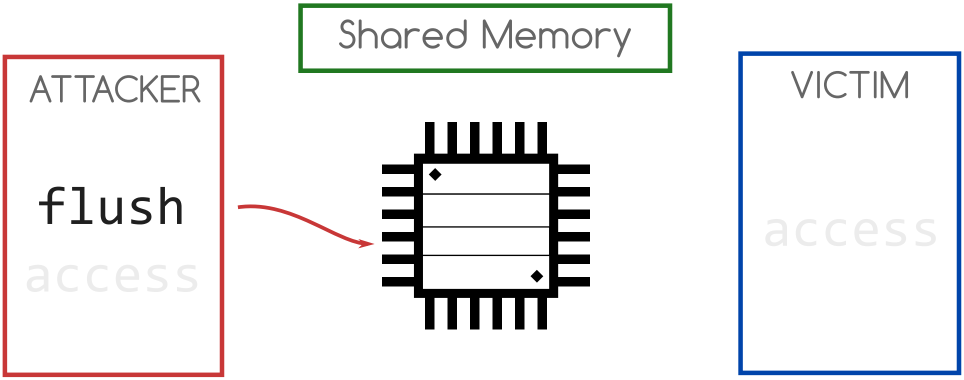


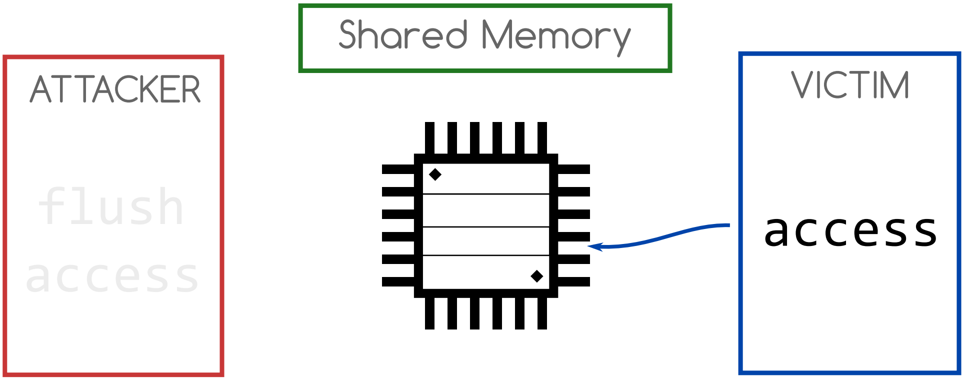
DRAM access,
slow
printf("%d", i);
printf("%d", i);
Cache miss
Cache hit
i
Request
Response
No DRAM access,
much faster

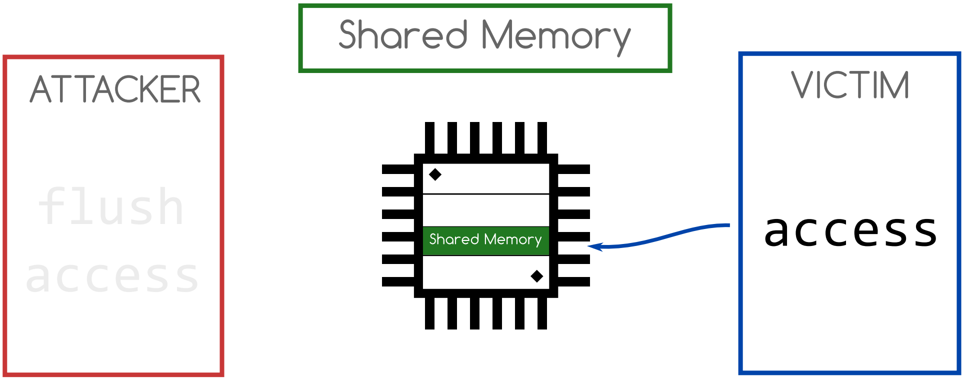

Shared Memory
ATTACKER
flush
access
VICTIM
access

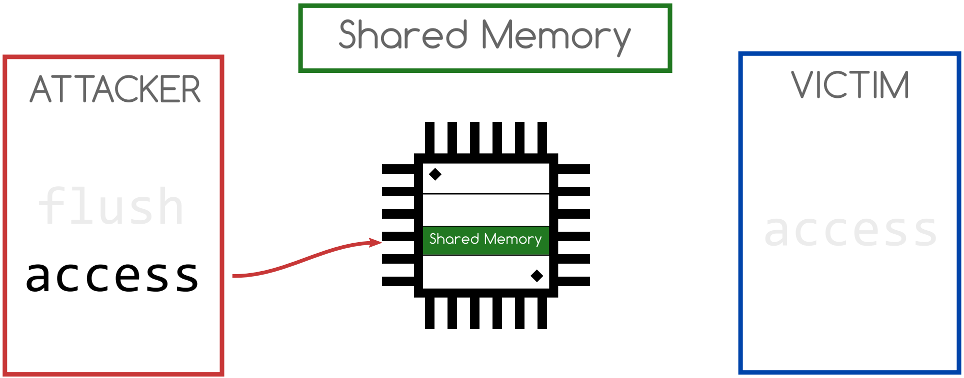

Shared Memory
ATTACKER
flush
access
cached
Shared Memory
cached
VICTIM
access

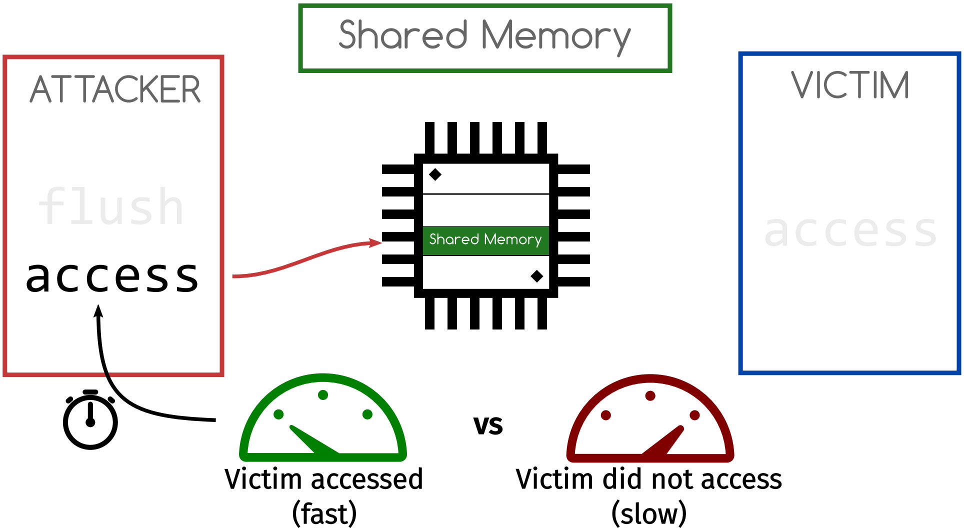

Shared Memory
ATTACKER
flush
access
Shared Memory
VICTIM
access

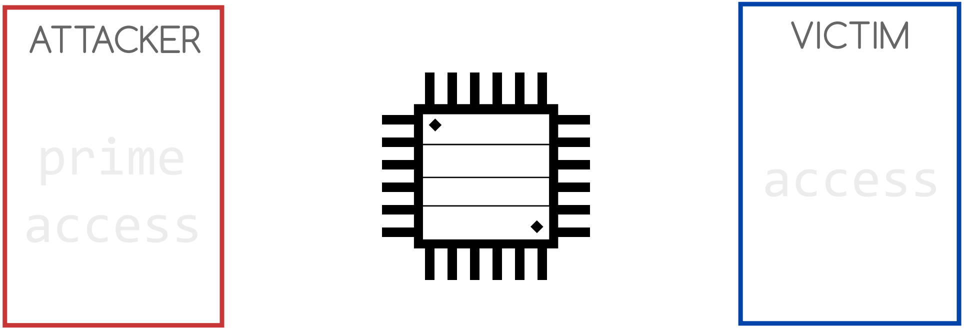

Shared Memory
ATTACKER
flush
access
VICTIM
access

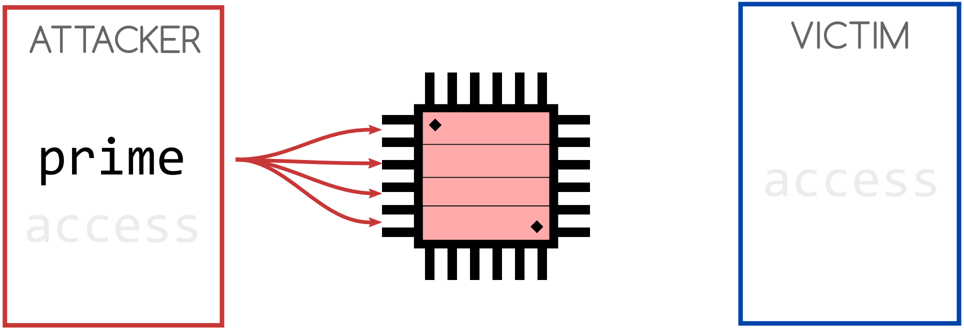

Shared Memory
ATTACKER
flush
access
VICTIM
access

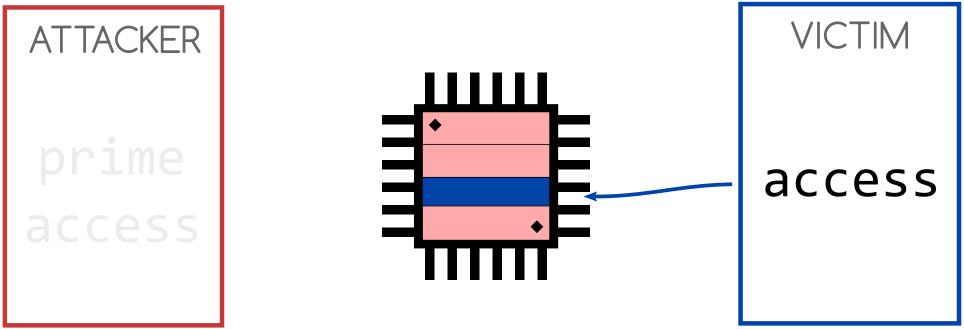

Shared Memory
ATTACKER
flush
access
Shared Memory
VICTIM
access

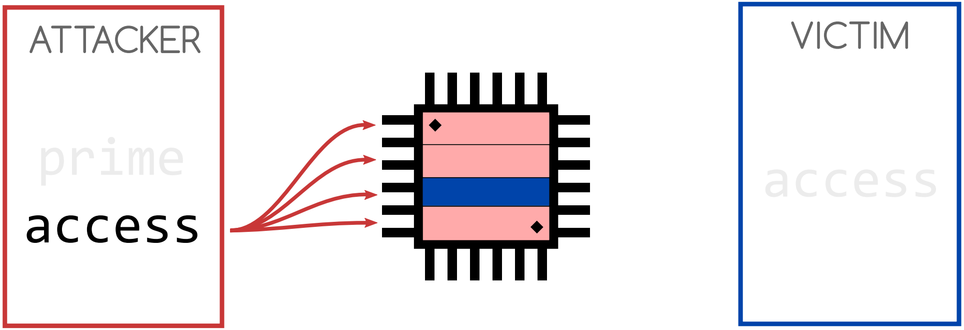

Shared Memory
ATTACKER
flush
access
Shared Memory
VICTIM
access

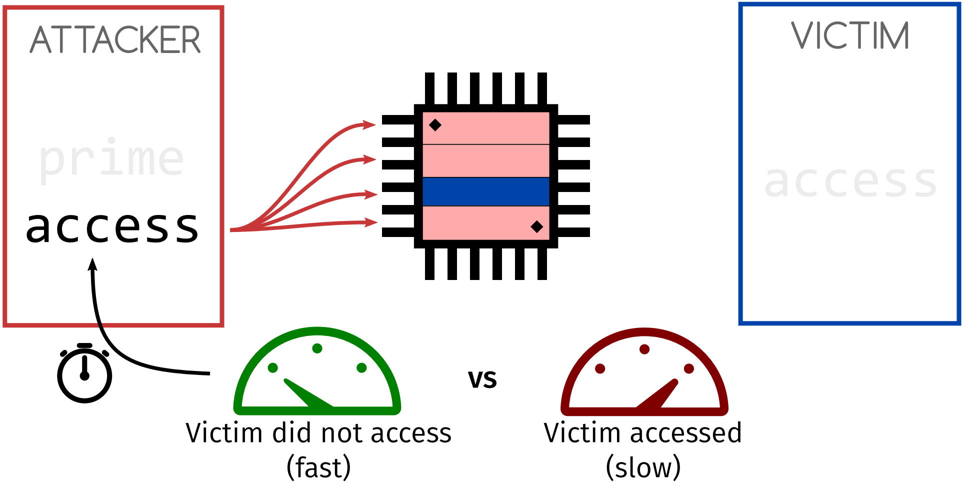

Shared Memory
ATTACKER
flush
access
Shared Memory
VICTIM
access
vs
Victim accessed (fast)
Victim did not access (slow)

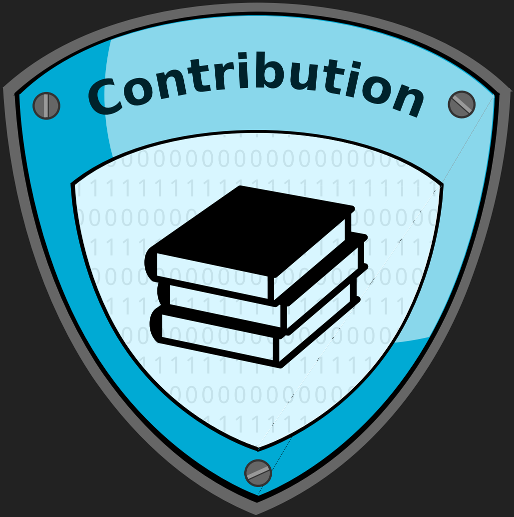

ATTACKER
prime
access
VICTIM
access


ATTACKER
prime
access
VICTIM
access


ATTACKER
prime
access
VICTIM
access


ATTACKER
prime
access
VICTIM
access


ATTACKER
prime
access
VICTIM
access
Victim did not access
(fast)
vs
Victim accessed
(slow)


Contribution

Publications

FANTASTIC TIMERS
AND WHERE TO FIND THEM
HIGH-RESOLUTION MICROARCHITECTURAL ATTACKS IN JAVASCRIPT

malware inside
SGX

KeyDrown

JavaScript zero
REAL JavaScript AND ZERO SIDE-CHANNEL ATTACKS

DECAF

JS
Template Attacks

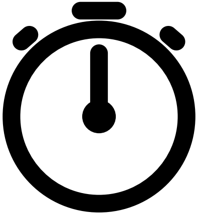
ZOMBIELOAD

FANTASTIC TIMERS

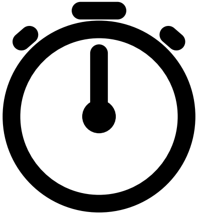
AND WHERE TO FIND THEM

HIGH-RESOLUTION MICROARCHITECTURAL ATTACKS IN JAVASCRIPT

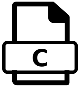
FC'17



- High-resolution timer to measure microarchitectural effects

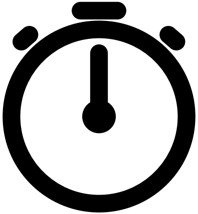


- High-resolution timer to measure microarchitectural effects

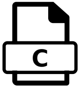


`rdtsc`

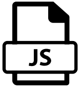
Timers in JavaScript



- High-resolution timer to measure microarchitectural effects

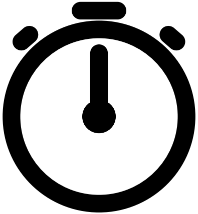


`rdtsc`

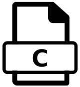


`performance.now()`

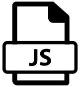
Timers in JavaScript



- High-resolution timer to measure microarchitectural effects



`rdtsc`

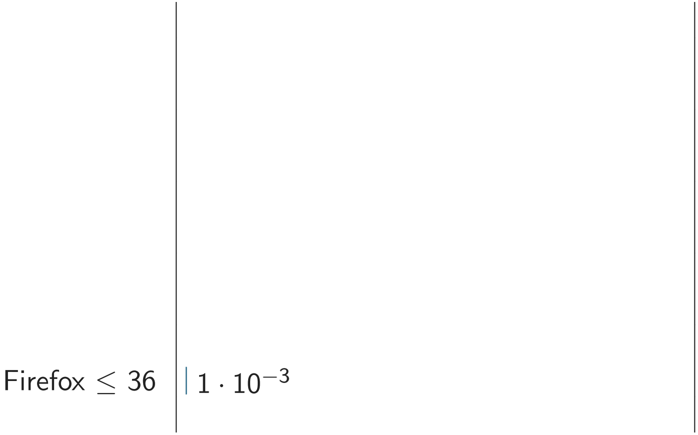


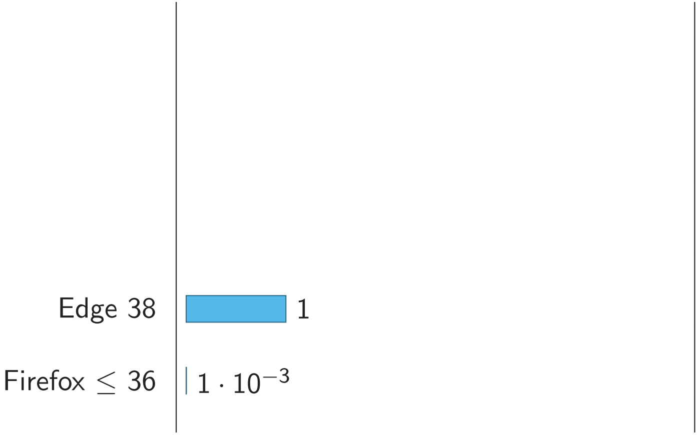
`performance.now()`

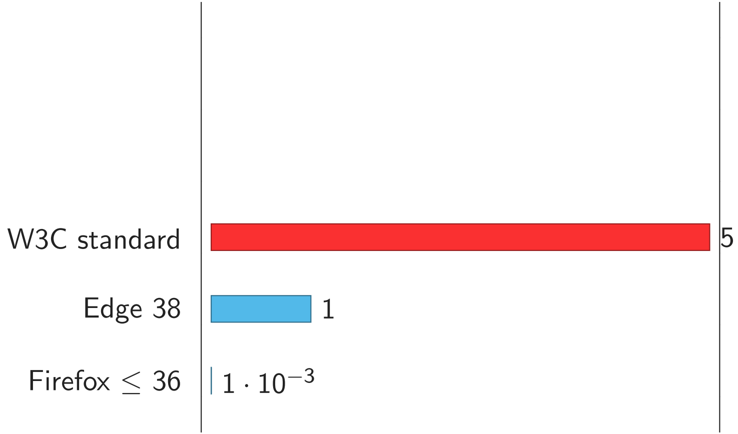
performance.now()

> [...] represent times as floating-point numbers with up to microsecond precision.
>
> — Mozilla Developer Network

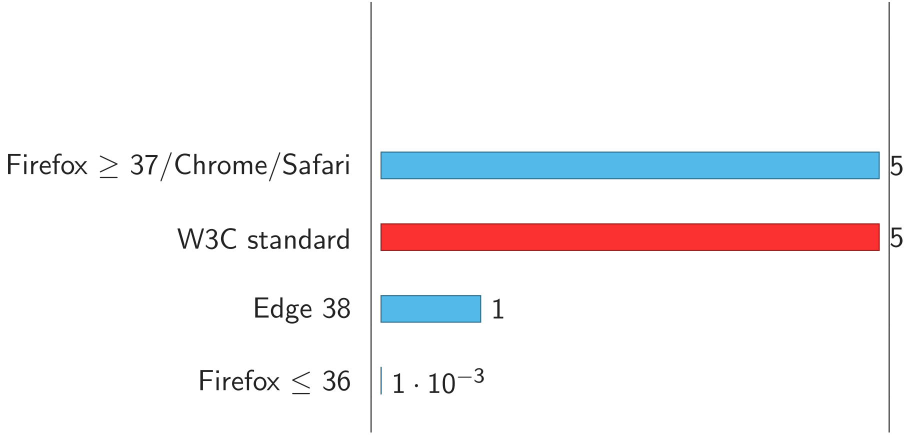
...up to microsecond precision?



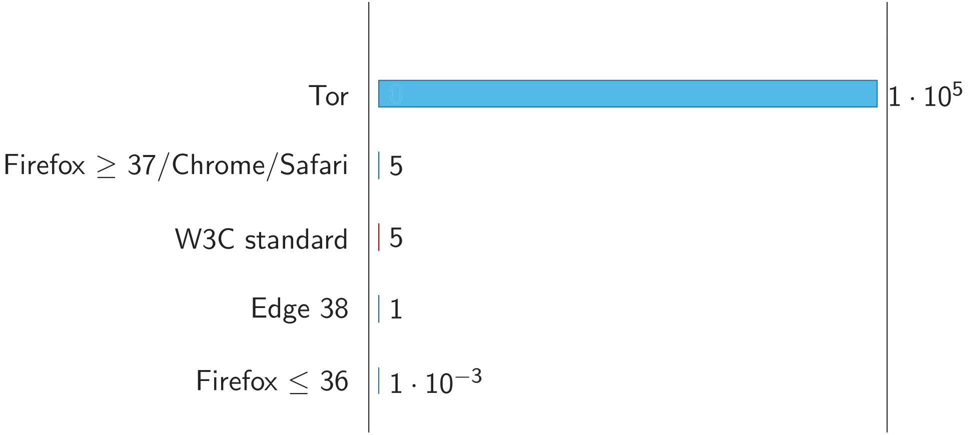

Edge 38
1
Firefox ≤ 36
1 · 10^−3

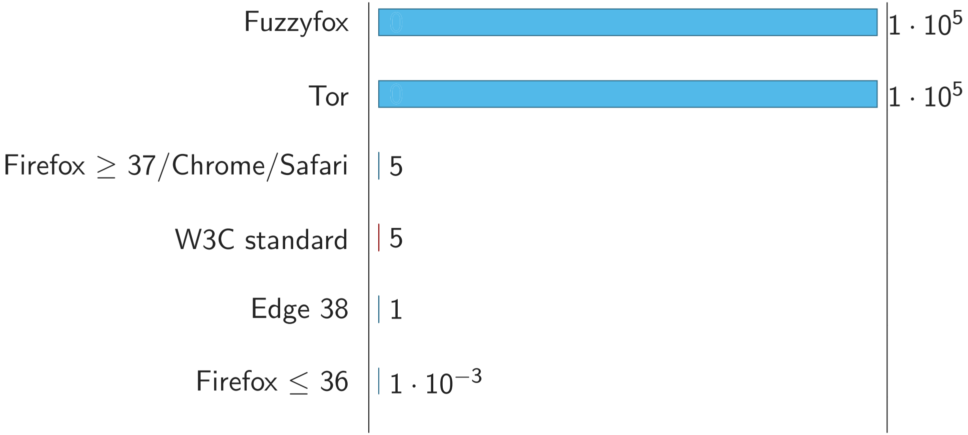

W3C standard
5
Edge 38
1
Firefox $\leq$ 36
$1 \cdot 10^{-3}$

...up to microsecond precision?

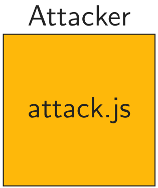


...up to microsecond precision?

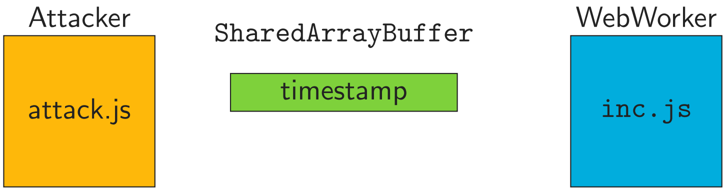


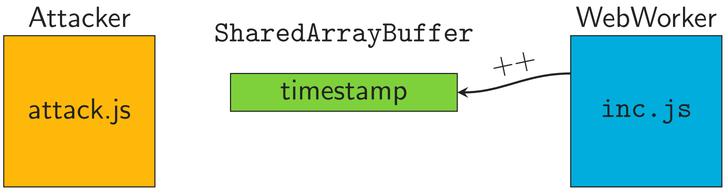

Fuzzyfox
1 · 10^5
Tor
1 · 10^5
Firefox ≥ 37/Chrome/Safari
5
W3C standard
5
Edge 38
1
Firefox ≤ 36
1 · 10^-3

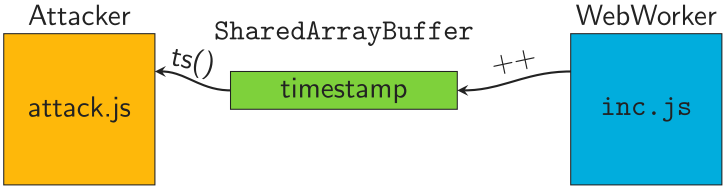
Building a timer - Web worker



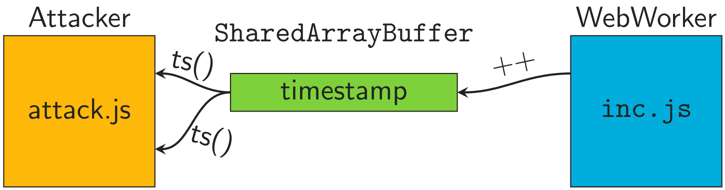
Building a timer - Web worker

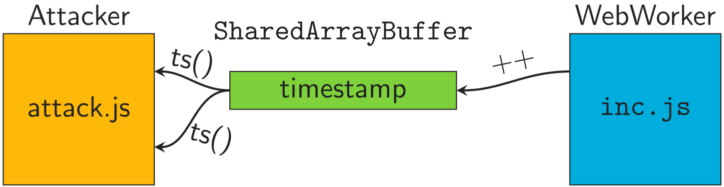

Attacker
attack.js
SharedArrayBuffer
timestamp
WebWorker
inc.js


Attacker
attack.js
SharedArrayBuffer
timestamp
++
WebWorker
inc.js


Attacker
attack.js
ts()
SharedArrayBuffer
timestamp
++
WebWorker
inc.js

Building a timer - Web worker




Attacker
attack.js
ts()
ts()
SharedArrayBuffer
timestamp
++
WebWorker
inc.js


2 ns
15 ns

Summary

- Rounding timers is not a solution

FC'17

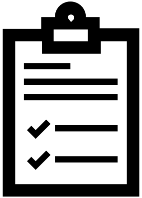
Michael Schwarz, Clémentine Maurice, Daniel Gruss and Stefan Mangard.
Fantastic Timers and Where to Find Them: High-Resolution Microarchitectural Attacks in JavaScript.

Summary

- Rounding timers is not a solution
- Multithreading + shared data → new timers

FC'17

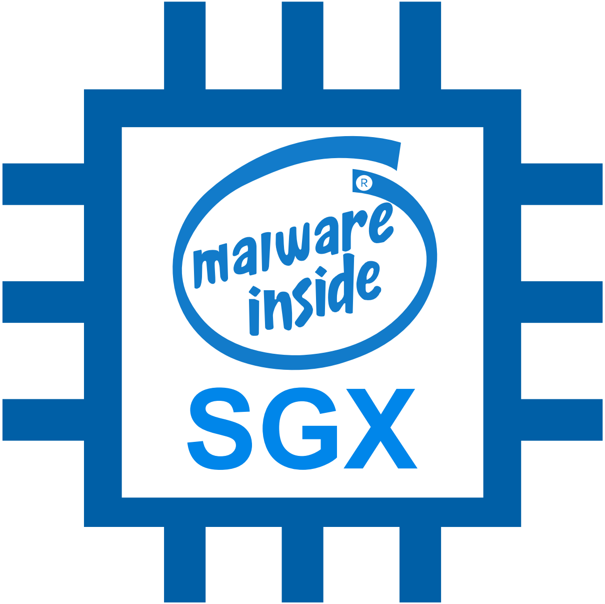
Michael Schwarz, Clémentine Maurice, Daniel Gruss and Stefan Mangard.
Fantastic Timers and Where to Find Them: High-Resolution Microarchitectural Attacks in JavaScript.



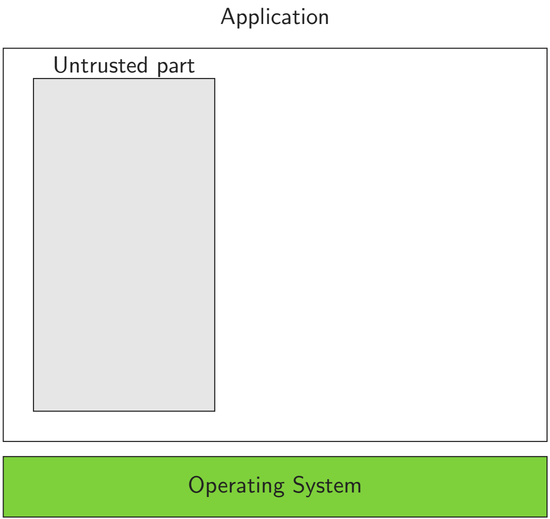
- Rounding timers is not a solution
- Multithreading + shared data → new timers
- Microarchitectural attacks in the browser are possible again

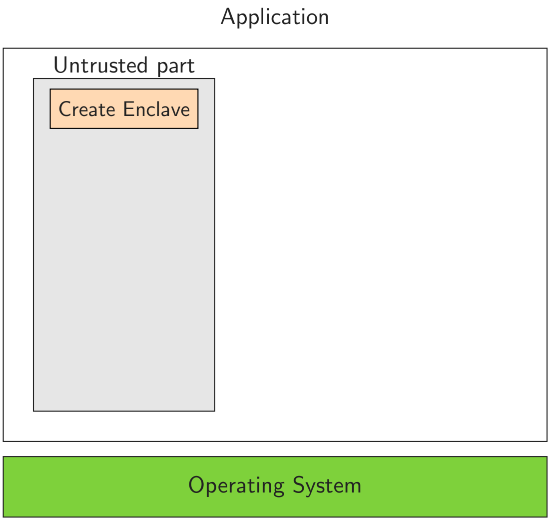
FC'17

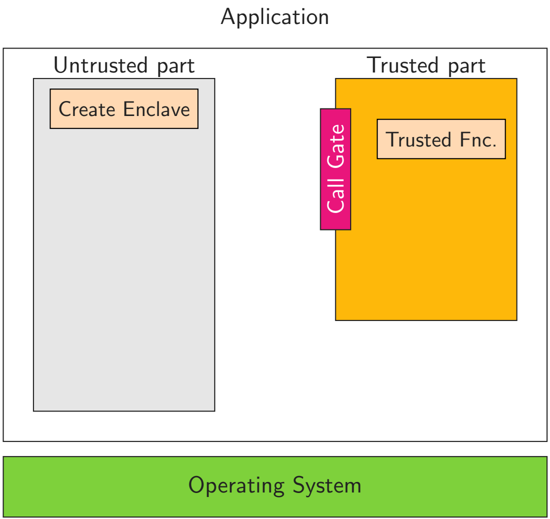
Michael Schwarz, Clémentine Maurice, Daniel Gruss and Stefan Mangard.
Fantastic Timers and Where to Find Them: High-Resolution Microarchitectural Attacks in JavaScript.

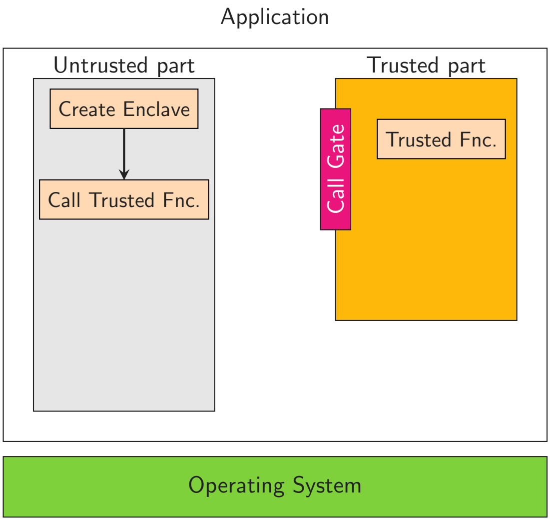


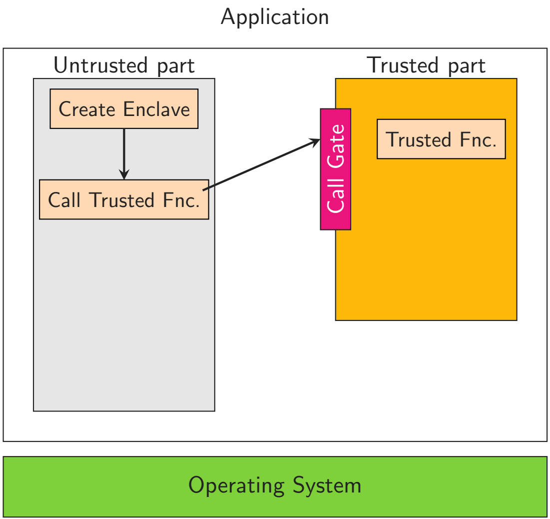
DIMVA'17

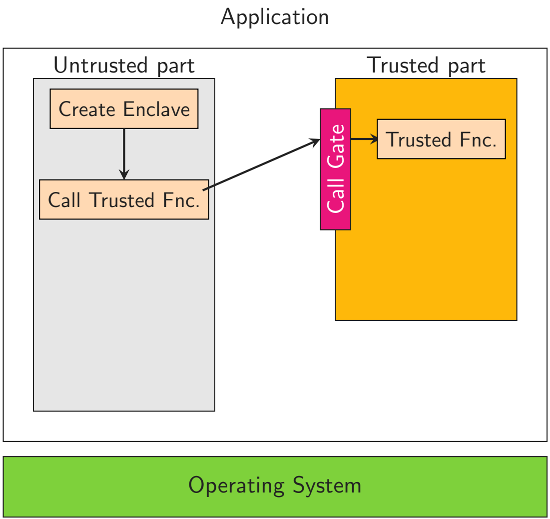

Application
Untrusted part
Operating System

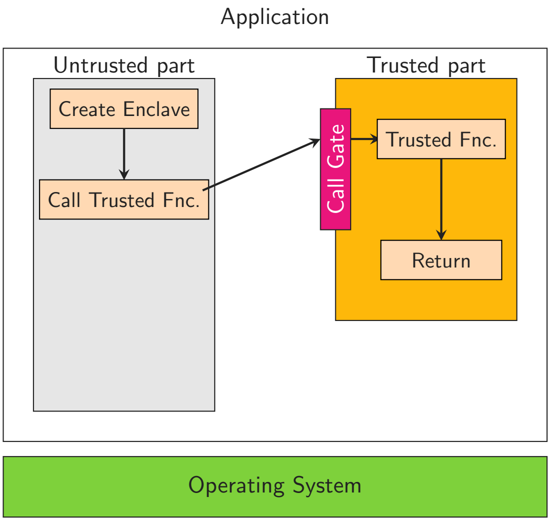

Application
Untrusted part
Create Enclave
Operating System

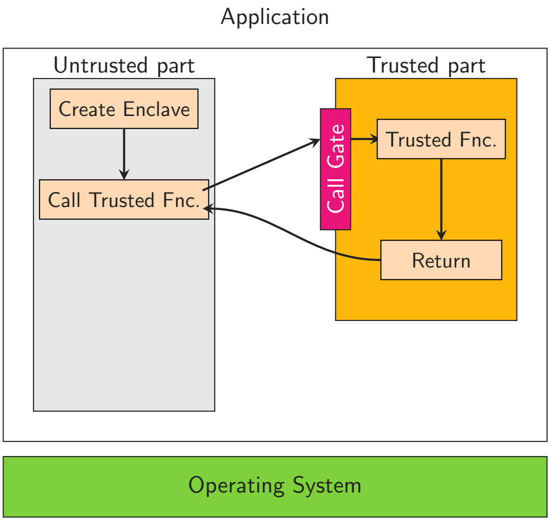

Application
Untrusted part
Create Enclave
Trusted part
Call Gate
Trusted Fnc.
Operating System

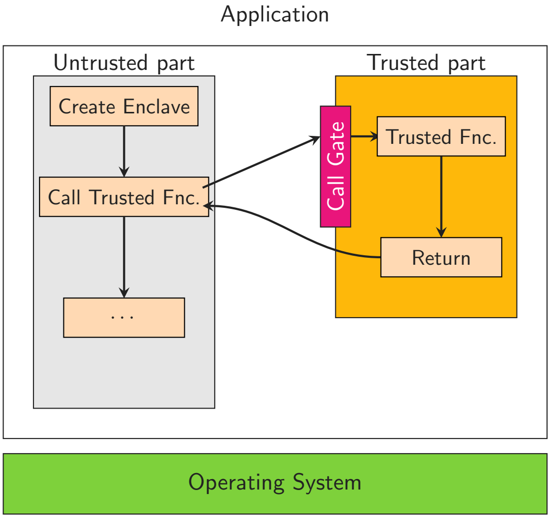

Application
Untrusted part
Create Enclave
Call Trusted Fnc.
Trusted part
Call Gate
Trusted Fnc.
Operating System

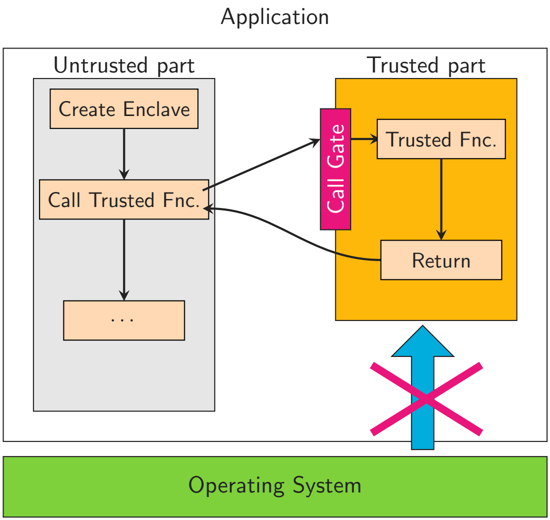

Application
Untrusted part
Create Enclave
Call Trusted Fnc.
Trusted part
Call Gate
Trusted Fnc.
Operating System


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


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

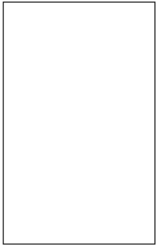

Application
Untrusted part
Trusted part
Create Enclave
Call Trusted Fnc.
Call Gate
Trusted Fnc.
Return
Operating System

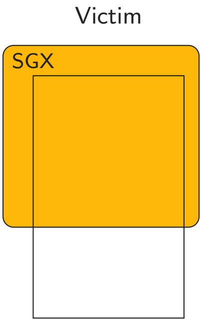

Application
Untrusted part
Trusted part
Create Enclave
Call Trusted Fnc.
...
Call Gate
Trusted Fnc.
Return
Operating System

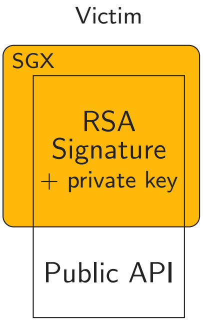

Application
Untrusted part
Trusted part
Create Enclave
Call Trusted Fnc.
...
Call Gate
Trusted Fnc.
Return
Operating System

Attack Settings

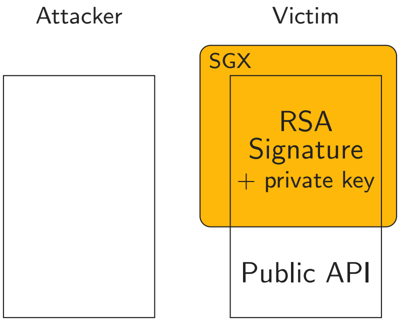
Victim

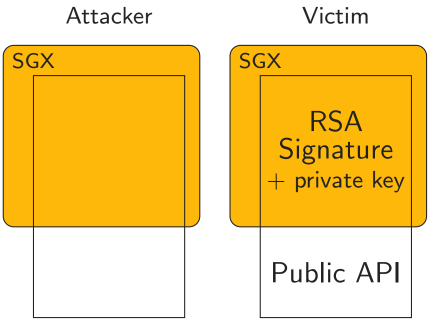



Victim
SGX

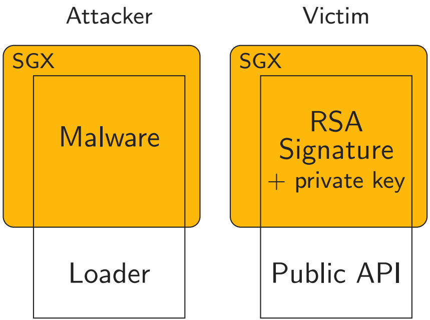

Victim
SGX
RSA
Signature
+ private key
Public API

Attack Settings

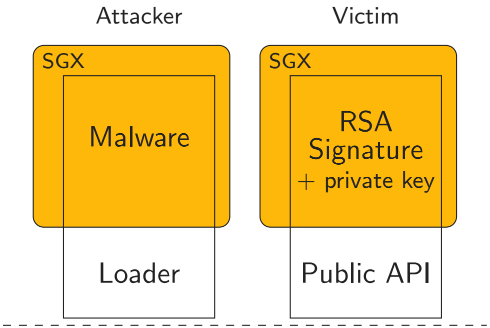



Attacker
Victim
SGX
SGX
RSA Signature
+ private key
Public API

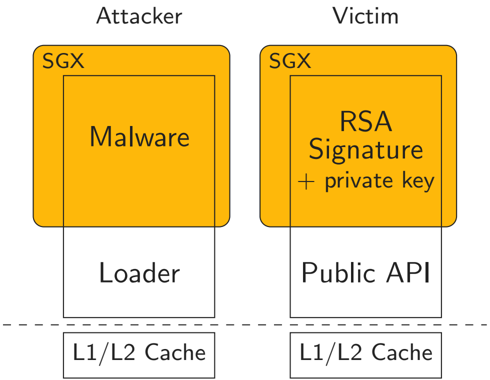
Attack Settings



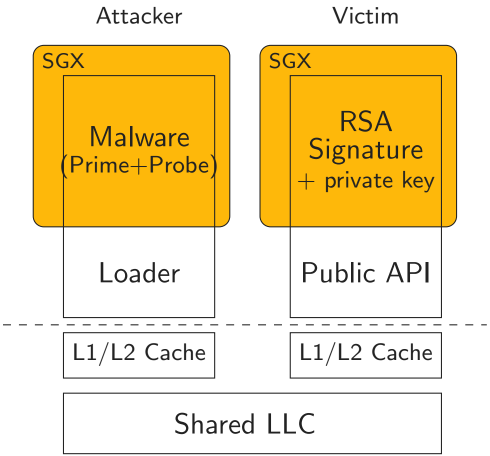
Attack Settings



Attack Settings



Attack Settings

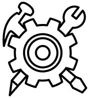


SGX Limitations

No privileges

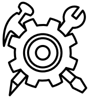
SGX Limitations

No privileges

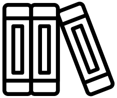


No syscall

No privileges



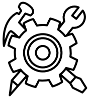
No syscall



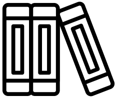
No shared memory

SGX Limitations

No privileges



No syscall

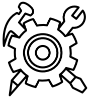


No shared memory

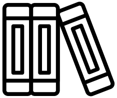
No rdtsc

SGX Limitations

No privileges



No syscall



No shared memory

No rdtsc

No physical addresses

SGX Limitations

No privileges

No syscall

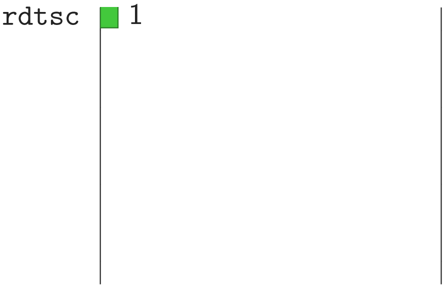
No shared memory

No rdtsc

No physical addresses

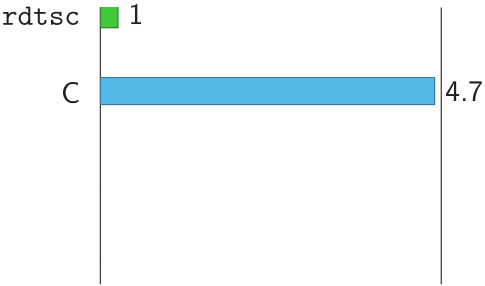
No 2 MB pages

CPU cycles per increment



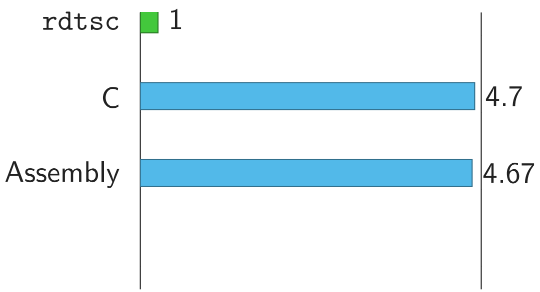

```
timestamp = rdtsc();
```

CPU cycles per increment



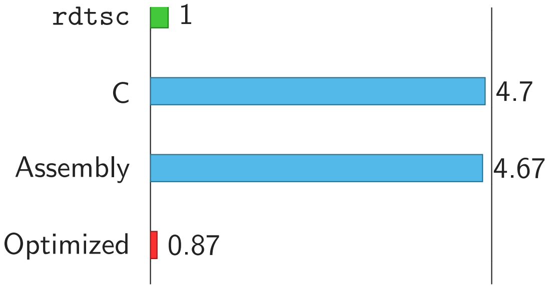

```
while(1) {
  timestamp++;
}
```

CPU cycles per increment



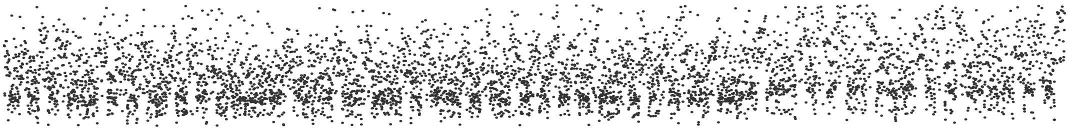

```
mov &timestamp, %rcx
1: incl (%rcx)
jmp 1b
```

CPU cycles per increment



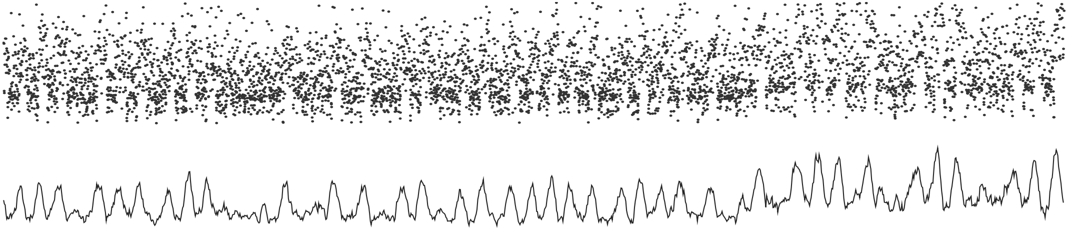

```
mov &timestamp, %rcx
1: inc %rax
mov %rax, (%rcx)
jmp 1b
```

Raw Prime+Probe trace...

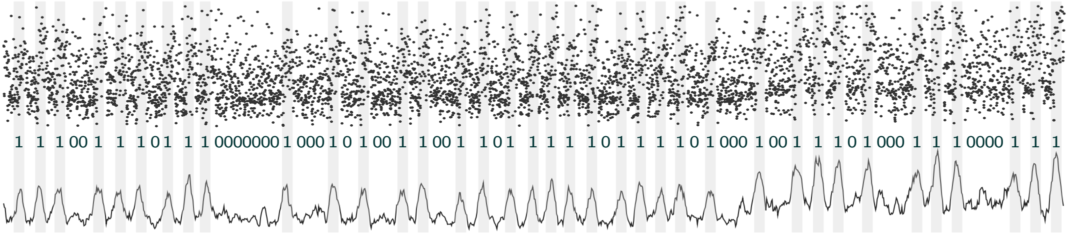


Measured Trace

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



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



- Attacks from within SGX possible

DIMVA'17

Michael Schwarz, Samuel Weiser, Daniel Gruss, Clémentine Maurice, Stefan Mangard.
Malware Guard Extension: Using SGX to Conceal Cache Attacks.

Summary

- Attacks from within SGX possible
- New high-resolution timer

DIMVA'17

Michael Schwarz, Samuel Weiser, Daniel Gruss, Clémentine Maurice, Stefan Mangard.
Malware Guard Extension: Using SGX to Conceal Cache Attacks.

Summary

- Attacks from within SGX possible
- New high-resolution timer
- Leaked 4096-bit RSA key (11 traces)

DIMVA'17

Michael Schwarz, Samuel Weiser, Daniel Gruss, Clémentine Maurice, Stefan Mangard.
Malware Guard Extension: Using SGX to Conceal Cache Attacks.

Summary

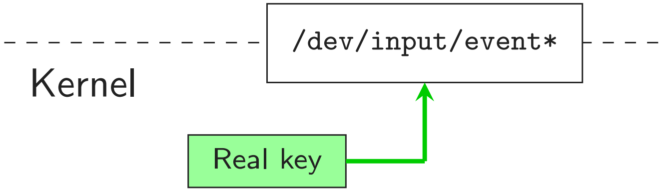
- Attacks from within SGX possible
- New high-resolution timer
- Leaked 4096-bit RSA key (11 traces)
- SGX allows hiding an attack

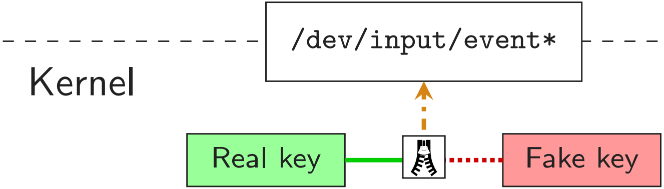
DIMVA'17

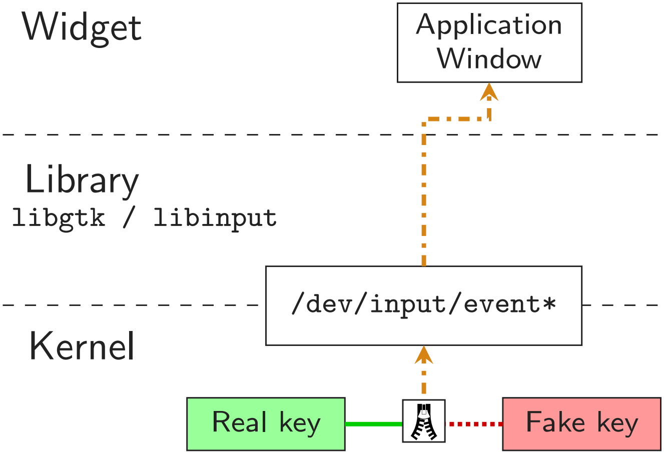
Michael Schwarz, Samuel Weiser, Daniel Gruss, Clémentine Maurice, Stefan Mangard.
Malware Guard Extension: Using SGX to Conceal Cache Attacks.

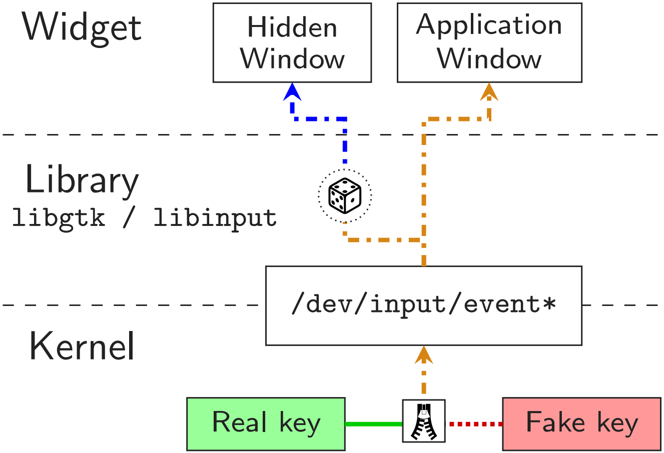
KeyDrown

NDSS'18

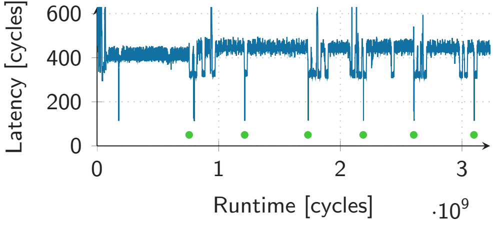

/dev/input/event*
Kernel
Real key


/dev/input/event*
Kernel
Real key
Fake key

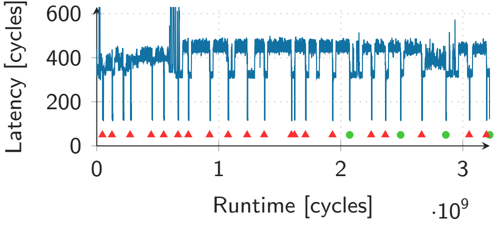

Widget
Application Window
Library
libgtk / libinput
/dev/input/event*
Kernel
Real key
Fake key


Widget
Hidden Window
Application Window
Library
libgtk / libinput
/dev/input/event*
Kernel
Real key
Fake key

Flush+Reload on `libgdk-3.so`

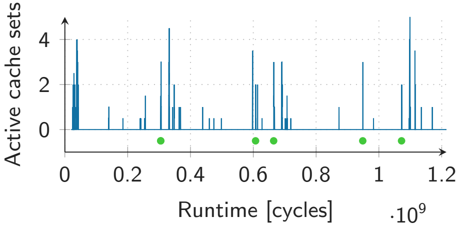


Without Keydrown, F-Score 0.99

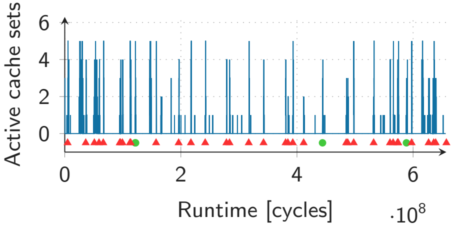


With Keydrown, F-Score 0.09

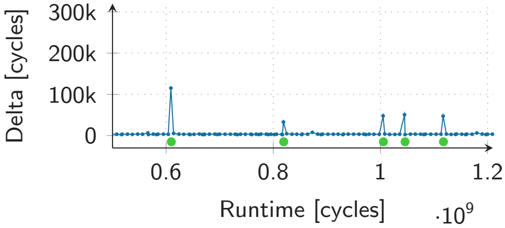
Prime+Probe on `i8042_interrupt`



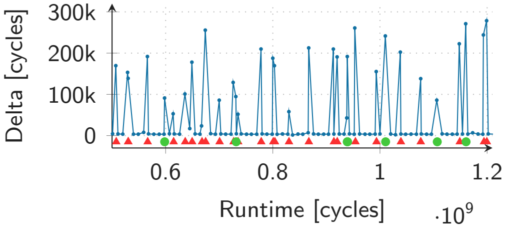
Without Keydrown, F-Score 0.81



With Keydrown, F-Score 0.11



Without Keydrown, F-Score 0.94



With Keydrown, F-Score 0.14

- Keystroke-timing attack only with high-resolution timer

NDSS'18

Michael Schwarz, Moritz Lipp, Daniel Gruss, Samuel Weiser, Clémentine Maurice, Raphael Spreitzer, Stefan Mangard.
KeyDrown: Eliminating Software-Based Keystroke Timing Side-Channel Attacks.

Summary

- Keystroke-timing attack only with high-resolution timer
- First accurate Prime+Probe attack on kernel

NDSS'18

Michael Schwarz, Moritz Lipp, Daniel Gruss, Samuel Weiser, Clémentine Maurice, Raphael Spreitzer, Stefan Mangard.
KeyDrown: Eliminating Software-Based Keystroke Timing Side-Channel Attacks.

Summary

- Keystroke-timing attack only with high-resolution timer
- First accurate Prime+Probe attack on kernel
- First generic countermeasure against keystroke-timing attack

NDSS'18

Michael Schwarz, Moritz Lipp, Daniel Gruss, Samuel Weiser, Clémentine Maurice, Raphael Spreitzer, Stefan Mangard.
KeyDrown: Eliminating Software-Based Keystroke Timing Side-Channel Attacks.

Summary

- Keystroke-timing attack only with high-resolution timer
- First accurate Prime+Probe attack on kernel
- First generic countermeasure against keystroke-timing attack
- PoC for Linux and Android

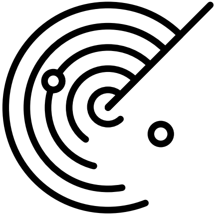
NDSS'18

Michael Schwarz, Moritz Lipp, Daniel Gruss, Samuel Weiser, Clémentine Maurice, Raphael Spreitzer, Stefan Mangard.
KeyDrown: Eliminating Software-Based Keystroke Timing Side-Channel Attacks.

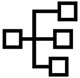
JavaScript zero

REAL JavaScript AND ZERO SIDE-CHANNEL ATTACKS

NDSS'18

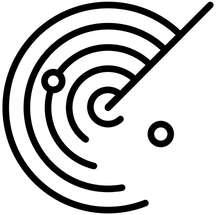


11 known attacks

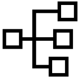


5 identified categories

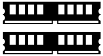
Landscape of Microarchitectural Attacks



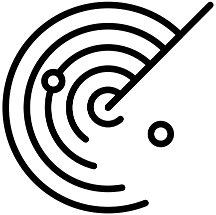
11 known attacks



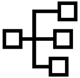
5 identified categories



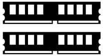
Memory addresses



11 known attacks



5 identified categories



Memory addresses

Accurate timing

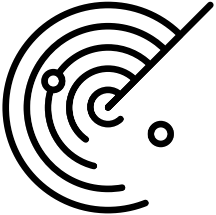
11 known attacks

5 identified categories

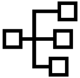
Memory addresses

Accurate timing

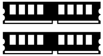
Multithreading



11 known attacks



5 identified categories



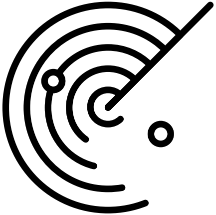
Memory addresses

Accurate timing

Multithreading

Shared data

Landscape of Microarchitectural Attacks



11 known attacks

5 identified categories

Memory addresses

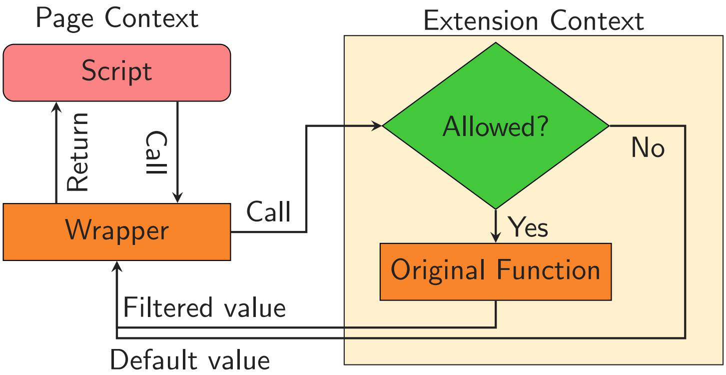
Accurate timing

Multithreading

Shared data

Sensor API

- Functions and properties are replaced by wrappers



Defense \ Prevents	Rowhammer.js	Page Deduplication	DRAM Covert Channel	Anti-ASLR	Cache Eviction	Keystroke Timing	Browser
Buffer ASLR	○	◐	○	●	●	○	○
Array preloading	●	○	●	○	○	○	○
Non-deterministic array	●	◐	◐	●	●	○	○
Array index randomization	○	●	○	●	○	○	○
Low-resolution timestamp	○	◐	○	○	○	◐	◐
Fuzzy time	○	◐*	○	○*	○	●*	●*
`WebWorker` polyfill	○	○	●	●	●	●	○
Message delay	○	○	○	○	○	◐	◐
Slow `SharedArrayBuffer`	○	○	●	◐	●	○	○
No `SharedArrayBuffer`	○	○*	●	●*	●	○*	○*
Summary	●	●	●	●	●	●	●

Fully prevented (●), partly prevented (◐), not prevented (○). Policies must be combined (*).

- Just rounding timers is not sufficient

NDSS'18

Michael Schwarz, Moritz Lipp, Daniel Gruss.
JavaScript Zero: Real JavaScript and Zero Side-Channel Attacks.

- Just rounding timers is not sufficient
- Microarchitectural attacks in the browser are still possible

NDSS'18

Michael Schwarz, Moritz Lipp, Daniel Gruss.
JavaScript Zero: Real JavaScript and Zero Side-Channel Attacks.

- Just rounding timers is not sufficient
- Microarchitectural attacks in the browser are still possible
- Efficient countermeasures can be implemented in browsers

NDSS'18

Michael Schwarz, Moritz Lipp, Daniel Gruss.
JavaScript Zero: Real JavaScript and Zero Side-Channel Attacks.

Summary

- Just rounding timers is not sufficient
- Microarchitectural attacks in the browser are still possible
- Efficient countermeasures can be implemented in browsers
- More microarchitectural attacks in JavaScript will appear

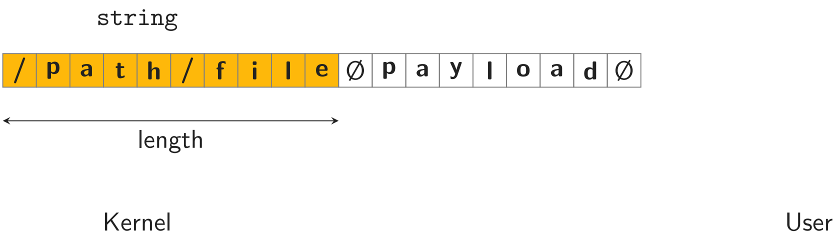
NDSS'18

Michael Schwarz, Moritz Lipp, Daniel Gruss.
JavaScript Zero: Real JavaScript and Zero Side-Channel Attacks.

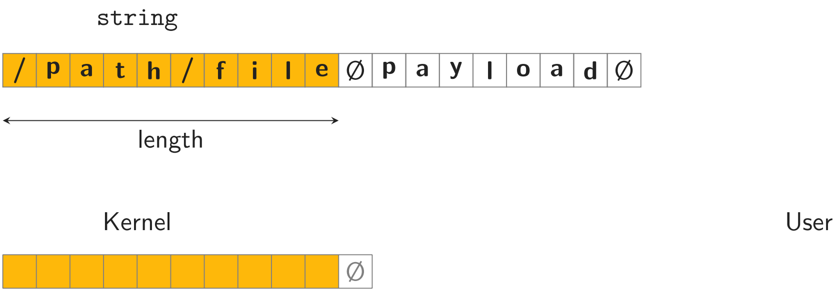
DECAF

AsiaCCS'18

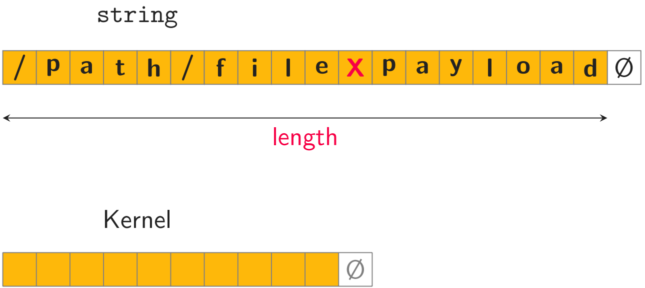
A Double Fetch



A Double Fetch

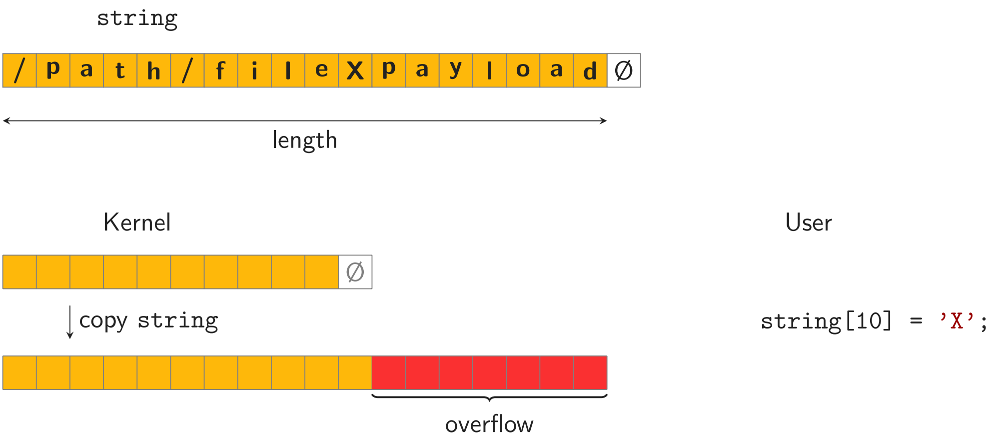


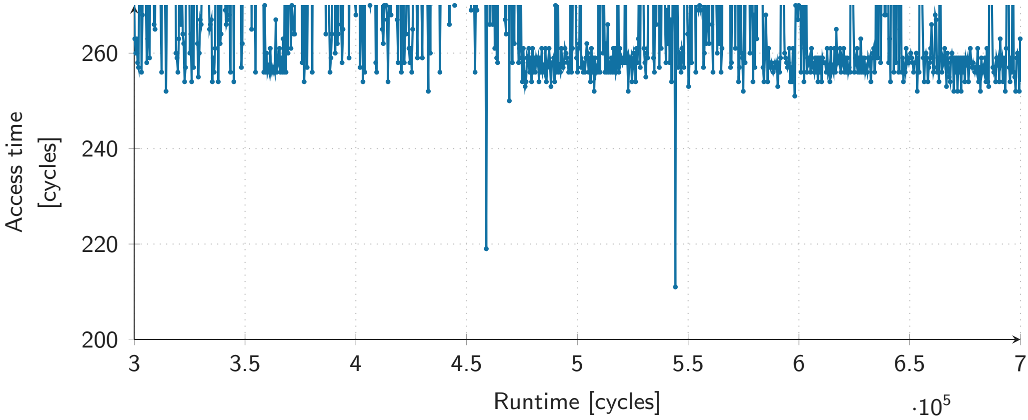
A Double Fetch

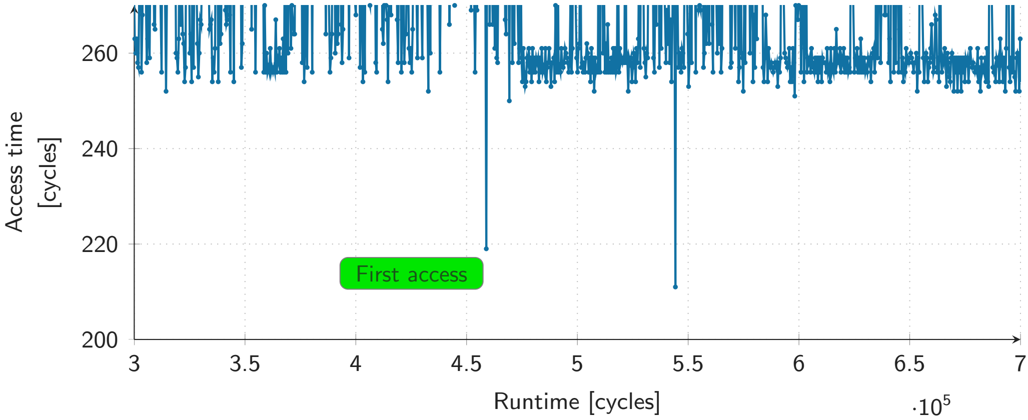


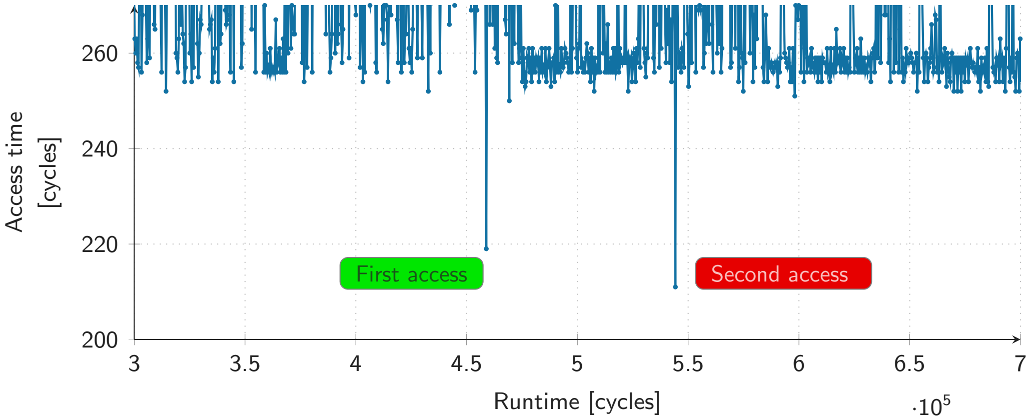

```
string[10] = 'X';
```

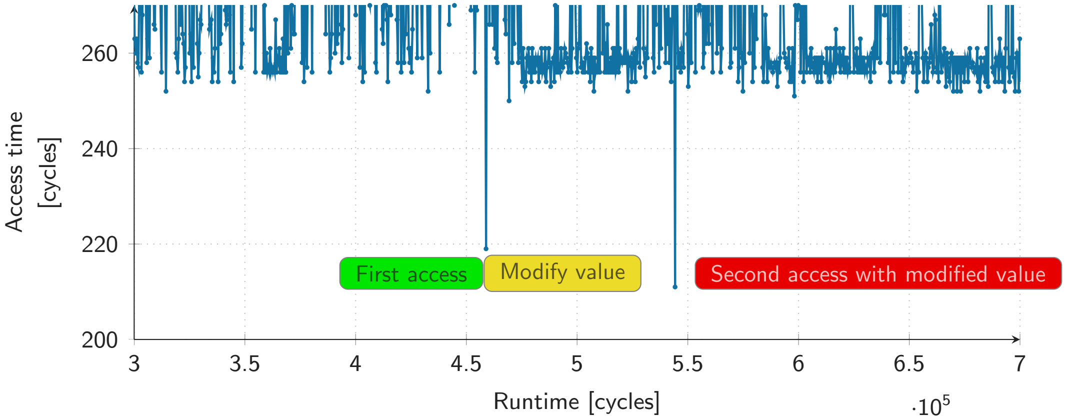
A Double Fetch

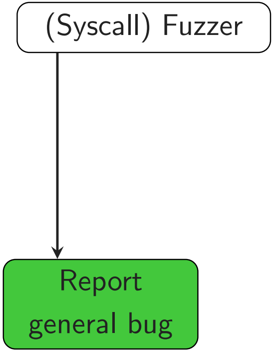


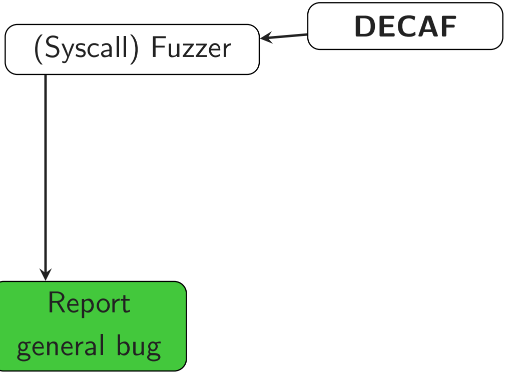

Access time [cycles]
260
240
220
200
3
3.5
4
4.5
5
5.5
6
6.5
7
Runtime [cycles]
$\cdot 10^5$

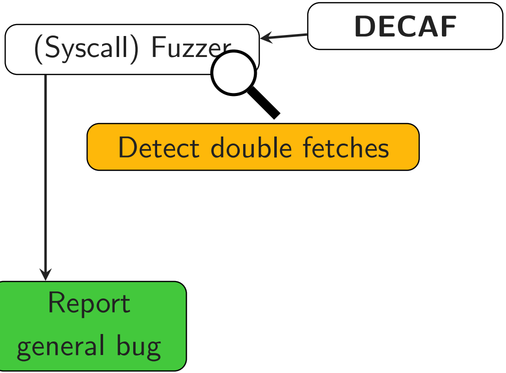

First access
Access time [cycles]
200
220
240
260
3
3.5
4
4.5
5
5.5
6
6.5
7
Runtime [cycles]
$\cdot 10^5$

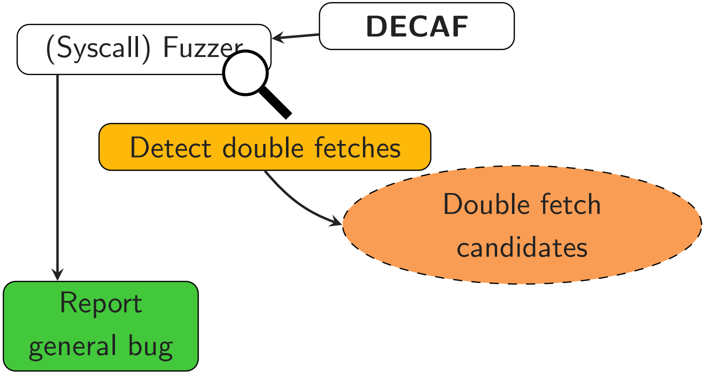

260
240
220
200
Access time [cycles]
First access
Second access
3
3.5
4
4.5
5
5.5
6
6.5
7
Runtime [cycles]
$\cdot 10^5$

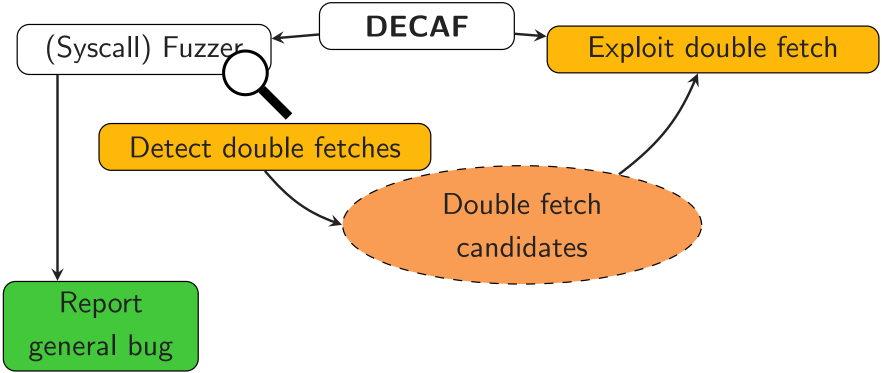

Access time [cycles]
260
240
220
200
First access
Modify value
Second access with modified value
3
3.5
4
4.5
5
5.5
6
6.5
7
Runtime [cycles]
$\cdot 10^5$

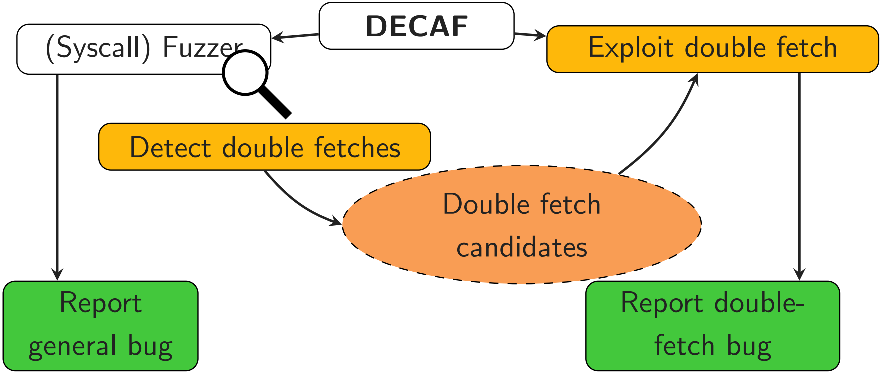

(Syscall) Fuzzer
Report
general bug

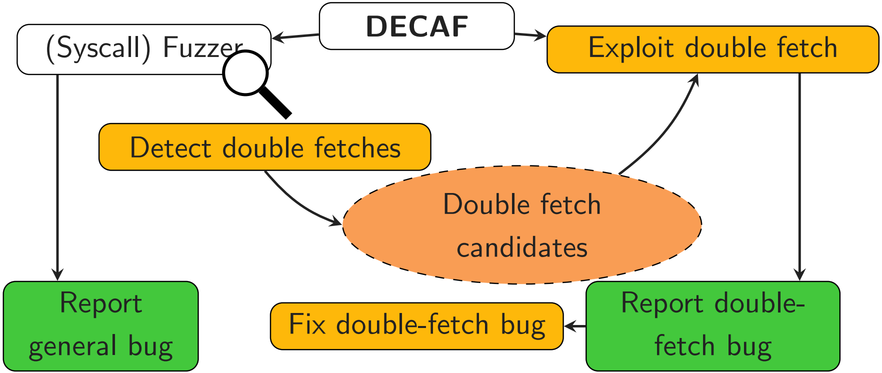

(Syscall) Fuzzer
DECAF
Report
general bug


DECAF
(Syscall) Fuzzer
Detect double fetches
Report
general bug


DECAF
(Syscall) Fuzzer
Detect double fetches
Double fetch candidates
Report general bug


DECAF
(Syscall) Fuzzer
Exploit double fetch
Detect double fetches
Double fetch candidates
Report general bug


DECAF
(Syscall) Fuzzer
Exploit double fetch
Detect double fetches
Double fetch candidates
Report general bug
Report double-fetch bug


DECAF
(Syscall) Fuzzer
Exploit double fetch
Detect double fetches
Double fetch candidates
Report general bug
Fix double-fetch bug
Report double-fetch bug

Summary

- Fuzzing + cache attacks → double-fetch bugs

AsiaCCS'18

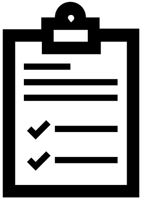
Michael Schwarz, Daniel Gruss, Moritz Lipp, Clémentine Maurice, Thomas Schuster, Anders Fogh, Stefan Mangard.
Automated Detection, Exploitation, and Elimination of Double-Fetch Bugs using Modern CPU Features.

- Fuzzing + cache attacks → double-fetch bugs
- Cache as trigger outperforms state of the art

AsiaCCS'18

Michael Schwarz, Daniel Gruss, Moritz Lipp, Clémentine Maurice, Thomas Schuster, Anders Fogh, Stefan Mangard.
Automated Detection, Exploitation, and Elimination of Double-Fetch Bugs using Modern CPU Features.

Summary



- Fuzzing + cache attacks → double-fetch bugs
- Cache as trigger outperforms state of the art
- Hardware transactional memory prevents exploitation

AsiaCCS'18

Michael Schwarz, Daniel Gruss, Moritz Lipp, Clémentine Maurice, Thomas Schuster, Anders Fogh, Stefan Mangard.
Automated Detection, Exploitation, and Elimination of Double-Fetch Bugs using Modern CPU Features.

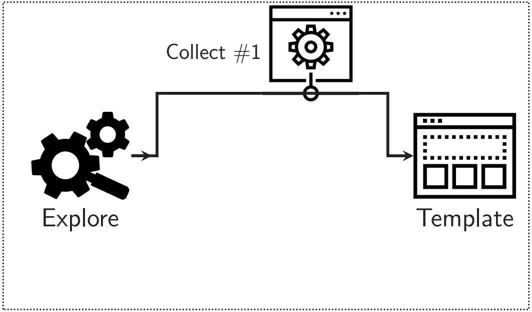
JS Template Attacks

NDSS'19

- Template attack on JavaScript properties

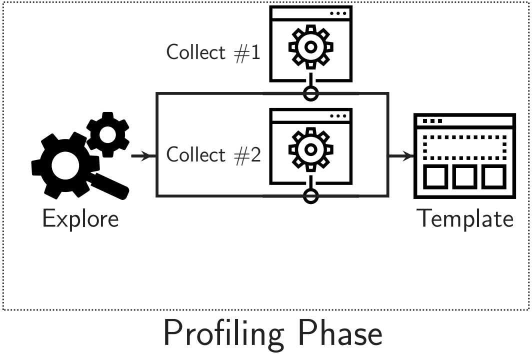
Profiling Phase

- Template attack on JavaScript properties

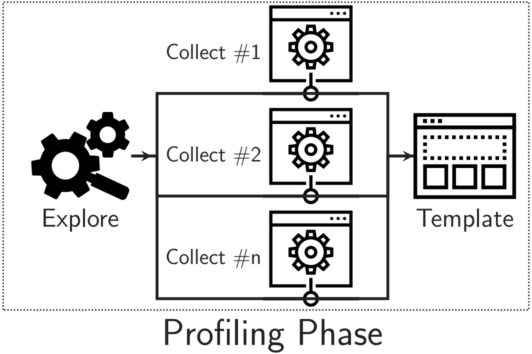


Profiling Phase

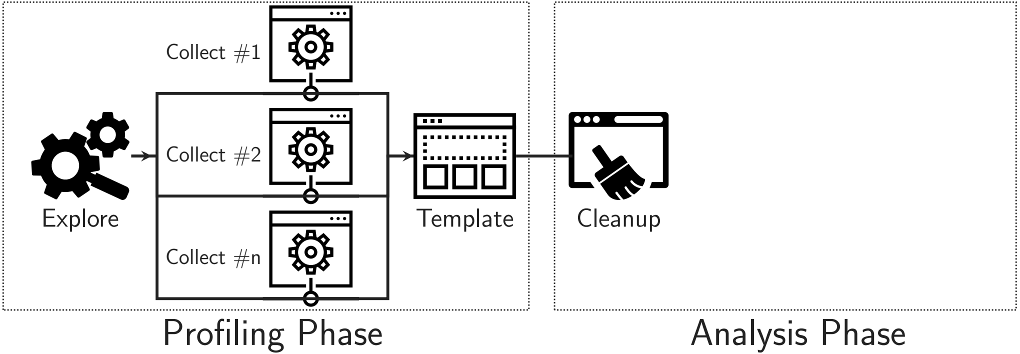
- Template attack on JavaScript properties



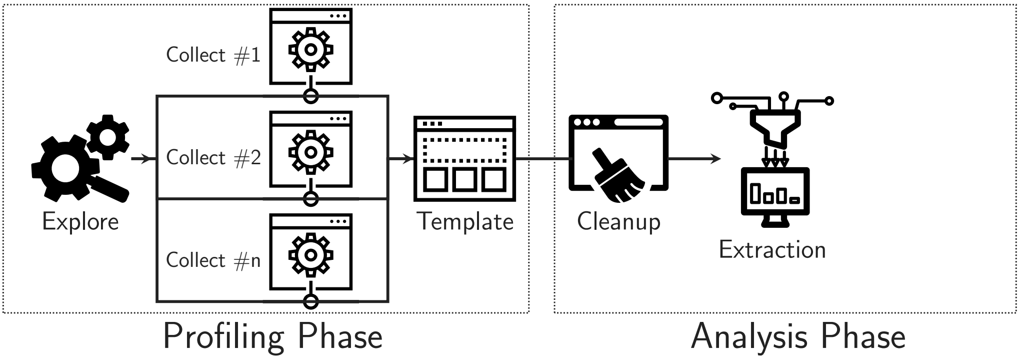
- Template attack on JavaScript properties



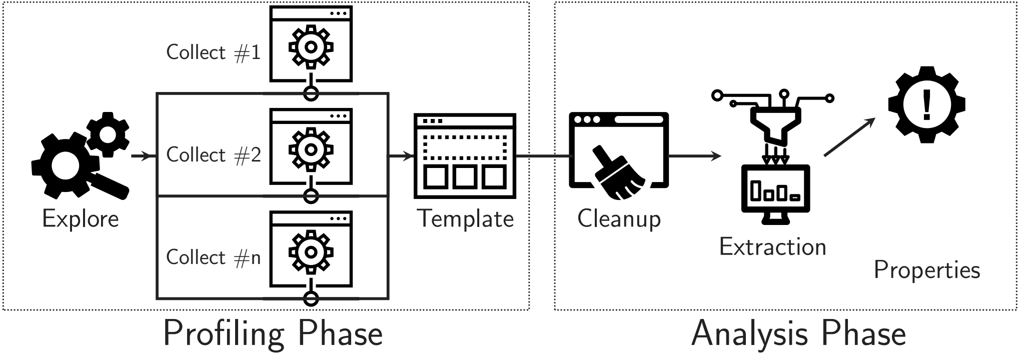
- Template attack on JavaScript properties



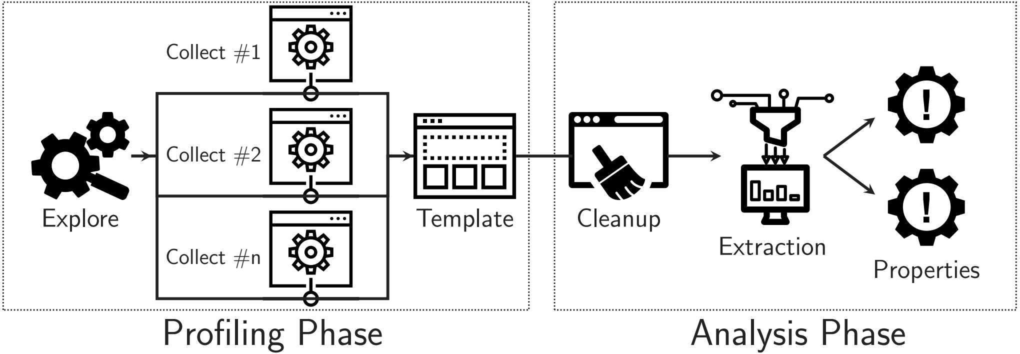
- Template attack on JavaScript properties

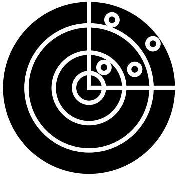


- Template attack on JavaScript properties



- Template attack on JavaScript properties





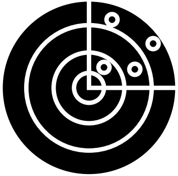
Browser + version

Detection

Browser + version

Privacy extensions

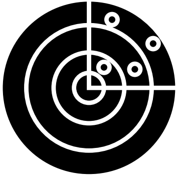
Detection



Browser + version

Privacy extensions

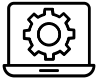
Private mode



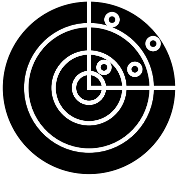
Browser + version

Privacy extensions

Private mode



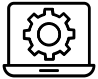
Operating system



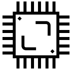
Browser + version

Privacy extensions

Private mode

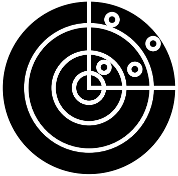


Operating system



CPU vendor

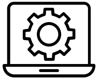
Detection



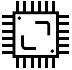
Browser + version

Privacy extensions

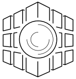
Private mode



Operating system



CPU vendor



Virtual machine

Summary

- JavaScript Template attacks detect environment properties

NDSS'19

Michael Schwarz, Florian Lackner, Daniel Gruss.
JavaScript Template Attacks: Automatically Inferring Host Information for Targeted Exploits.

Summary

- JavaScript Template attacks detect environment properties
- Enables exploits, side-channel attacks and plausible phishing

NDSS'19

Michael Schwarz, Florian Lackner, Daniel Gruss.
JavaScript Template Attacks: Automatically Inferring Host Information for Targeted Exploits.

Summary

- JavaScript Template attacks detect environment properties
- Enables exploits, side-channel attacks and plausible phishing
- Tool for browser vendors to find leakage

NDSS'19

Michael Schwarz, Florian Lackner, Daniel Gruss.
JavaScript Template Attacks: Automatically Inferring Host Information for Targeted Exploits.

Summary

- JavaScript Template attacks detect environment properties
- Enables exploits, side-channel attacks and plausible phishing
- Tool for browser vendors to find leakage
- Advances field of fingerprinting

NDSS'19

Michael Schwarz, Florian Lackner, Daniel Gruss.
JavaScript Template Attacks: Automatically Inferring Host Information for Targeted Exploits.

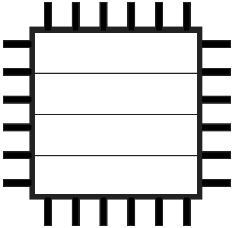
ZombieLoad

CCS'19

User Memory

	A	B
C	D	E
F	G	H
I	J	K
L	M	N
O	P	Q
R	S	T
U	V	W
X	Y	Z

```
char value = kernel[0]
```



User Memory

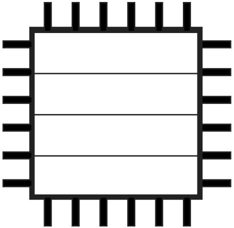
	A	B
C	D	E
F	G	H
I	J	K
L	M	N
O	P	Q
R	S	T
U	V	W
X	Y	Z

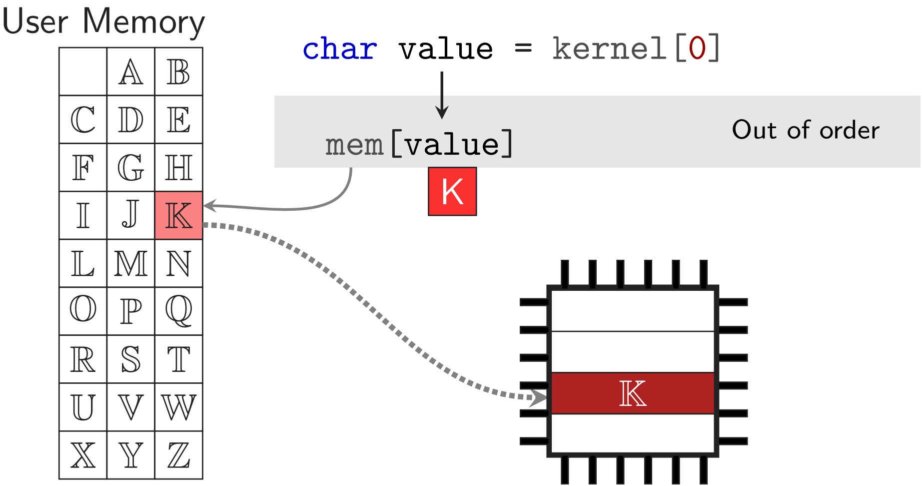
```
char value = kernel[0]
```

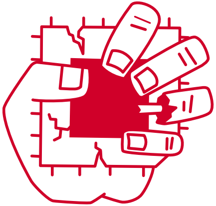
```
mem[value]
```

Out of order

K

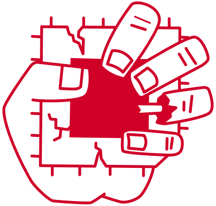



User Memory
A
B
C
D
E
F
G
H
I
J
K
L
M
N
O
P
Q
R
S
T
U
V
W
X
Y
Z
char value = kernel[0]
mem[value]
Out of order
K
K



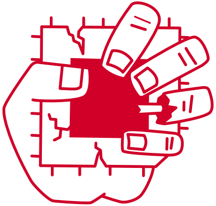
AES-NI key

Results



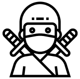
AES-NI key

SGX sealing key

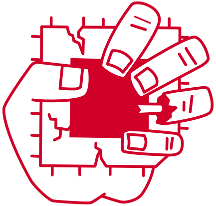


AES-NI key

SGX sealing key

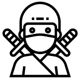


Cross-VM covert channel



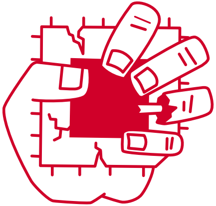
AES-NI key

SGX sealing key



Cross-VM covert channel

Keyword matching



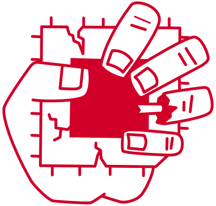
AES-NI key

SGX sealing key

Cross-VM covert channel

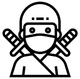
Keyword matching

URL recovery



AES-NI key

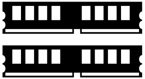
SGX sealing key



Cross-VM covert channel

Keyword matching

URL recovery



Targeted leakage

- Meltdown on the line-fill buffer

CCS'19

Michael Schwarz, Moritz Lipp, Daniel Moghimi, Jo Van Bulck, Julian Stecklina, Thomas Prescher, Daniel Gruss.
ZombieLoad: Cross-Privilege-Boundary Data Sampling.

- Meltdown on the line-fill buffer
- Data sampling attacks are powerful

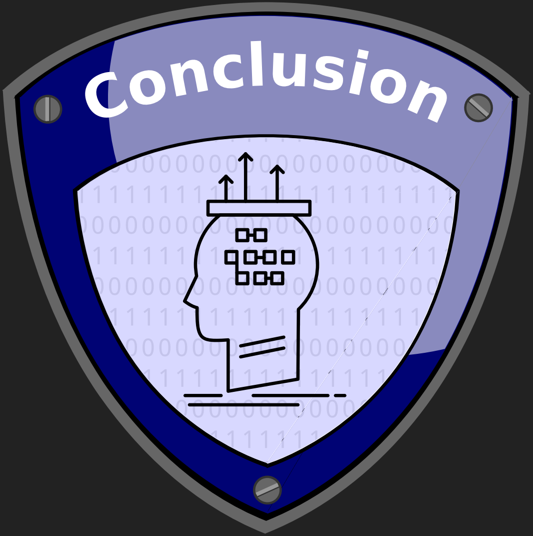
CCS'19

Michael Schwarz, Moritz Lipp, Daniel Moghimi, Jo Van Bulck, Julian Stecklina, Thomas Prescher, Daniel Gruss.
ZombieLoad: Cross-Privilege-Boundary Data Sampling.

- Meltdown on the line-fill buffer
- Data sampling attacks are powerful
- Just the beginning of CPU vulnerabilities

CCS'19

Michael Schwarz, Moritz Lipp, Daniel Moghimi, Jo Van Bulck, Julian Stecklina, Thomas Prescher, Daniel Gruss.
ZombieLoad: Cross-Privilege-Boundary Data Sampling.


Conclusion

- Attack research is necessary for effective defenses

Conclusion

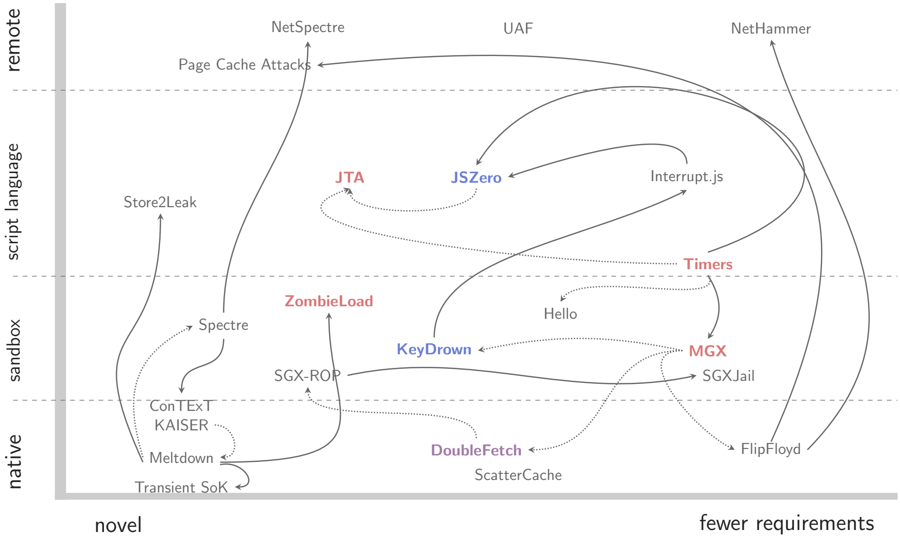
- Attack research is necessary for effective defenses
- Often wrong assumptions on requirements

{···}

- Attack research is necessary for effective defenses
- Often wrong assumptions on requirements
- Data-dependent optimizations often lead to side channel

- Attack research is necessary for effective defenses
- Often wrong assumptions on requirements
- Data-dependent optimizations often lead to side channel
- Trade-off between performance and security

Paper Connections



Software-based Side-Channel Attacks and Defenses in Restricted Environments

PhD Defense

Michael Schwarz
IAIK, Graz University of Technology
November 5, 2019