black hat ASIA 2019

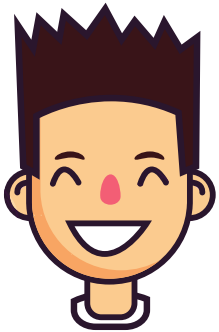
NetSpectre

A Truly Remote Spectre Variant

Michael Schwarz @misc0110

Martin Schwarzl @marv0x90

Who am I?



Michael Schwarz

PhD candidate @ Graz University of Technology

@misc0110

michael.schwarz@iaik.tugraz.at

Who am I?

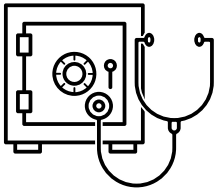
Martin Schwarzl

Master student @ Graz University of Technology

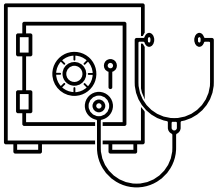
@marv0x90

m.schwarzl@student.tugraz.at

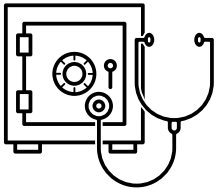
- Bug-free software does not mean safe execution



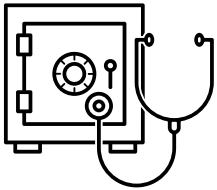
- Bug-free software does not mean safe execution
- Information leaks due to underlying hardware



Side-Channel Attacks



- Bug-free software does not mean safe execution
- Information leaks due to underlying hardware
- Exploit leakage through side-effects



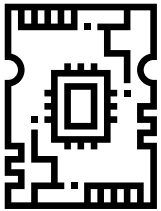
- Bug-free software does not mean safe execution
- Information leaks due to underlying hardware
- Exploit leakage through side-effects

Power consumption

Execution time

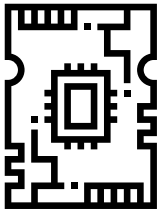
CPU caches

Microarchitectural Components



- Modern CPUs contain multiple microarchitectural elements

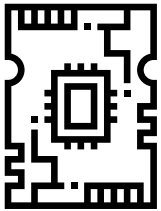
Microarchitectural Components



- Modern CPUs contain multiple microarchitectural elements

Caches and buffers

Predictors

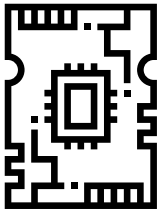


- Modern CPUs contain multiple microarchitectural elements

Caches and buffers

Predictors

- Transparent for the programmer



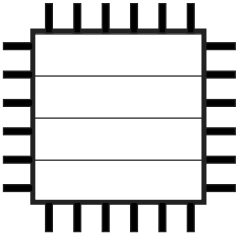
- Modern CPUs contain multiple microarchitectural elements

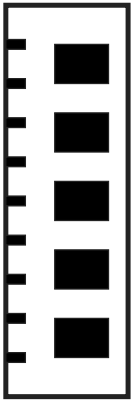
Caches and buffers

Predictors

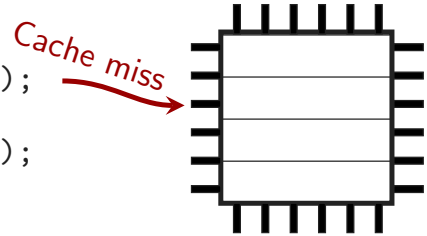
- Transparent for the programmer
- Timing optimizations → side-channel leakage

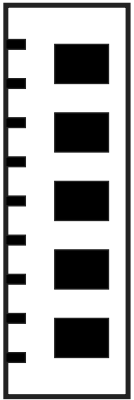
```
printf("%d", i);

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



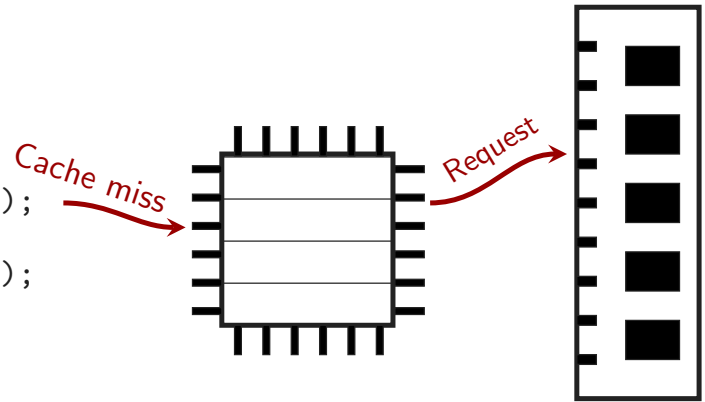


```
printf("%d", i);

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

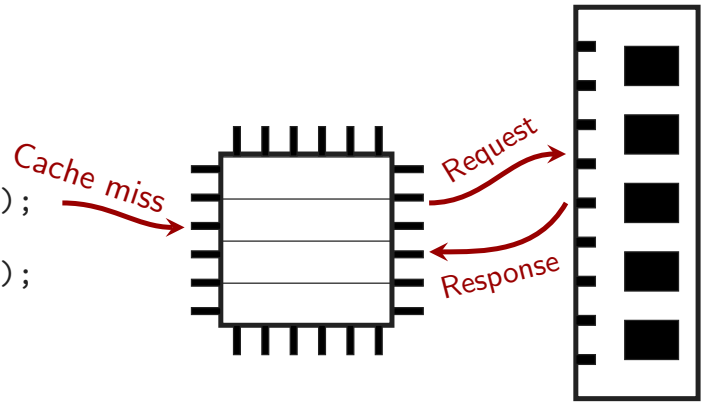




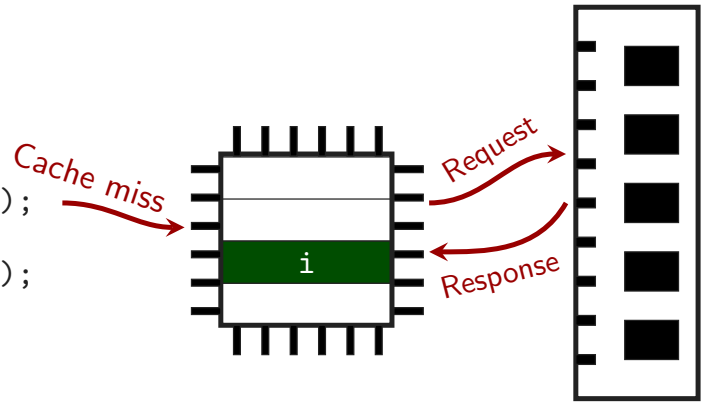
```
printf("%d", i);

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

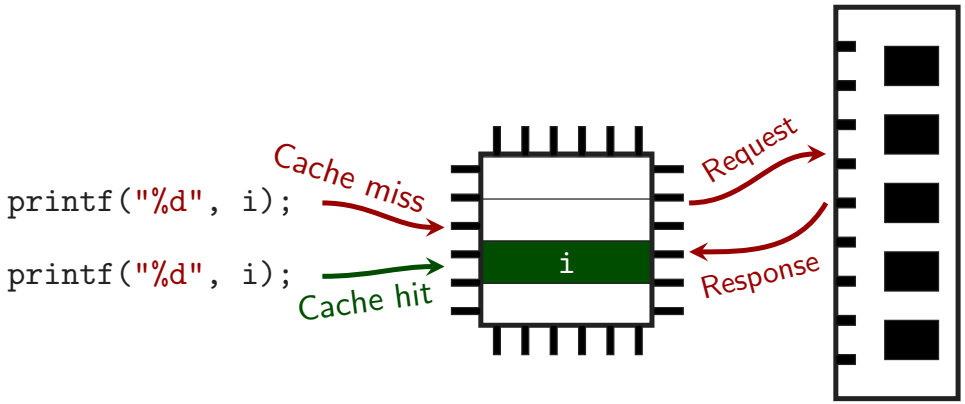


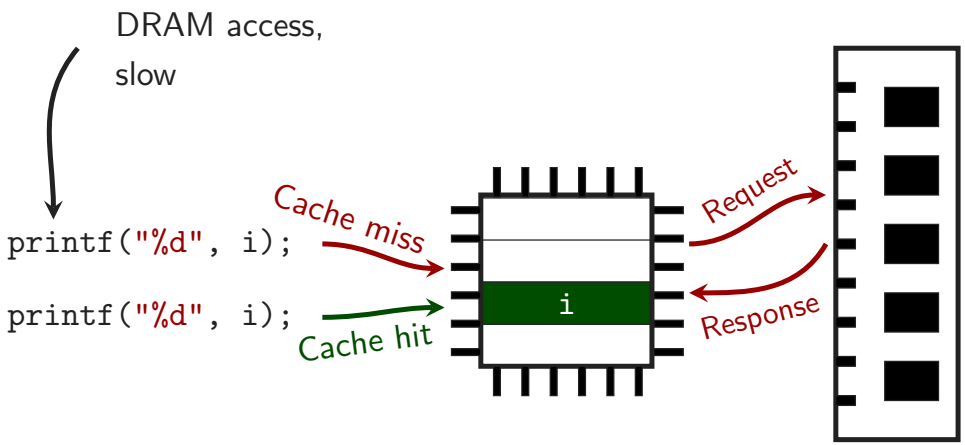
```
printf("%d", i);

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

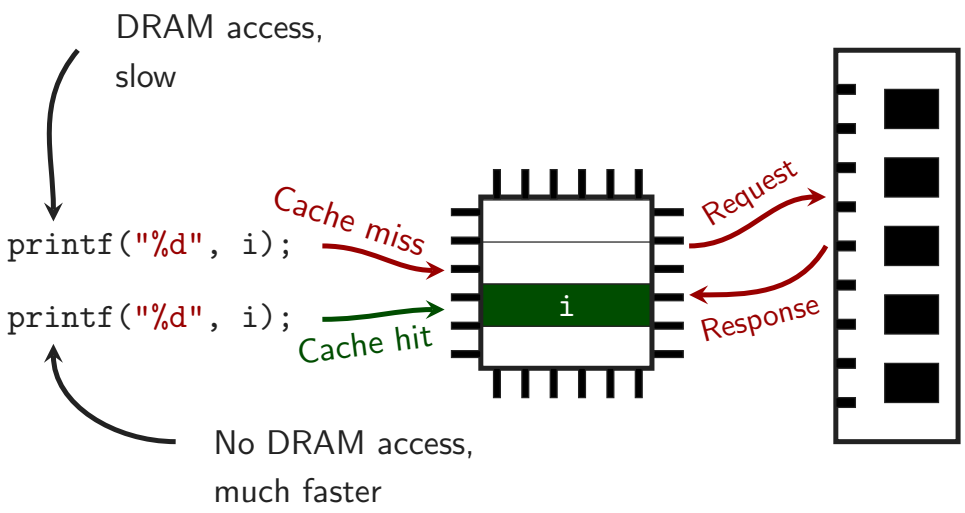


```
printf("%d", i);

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

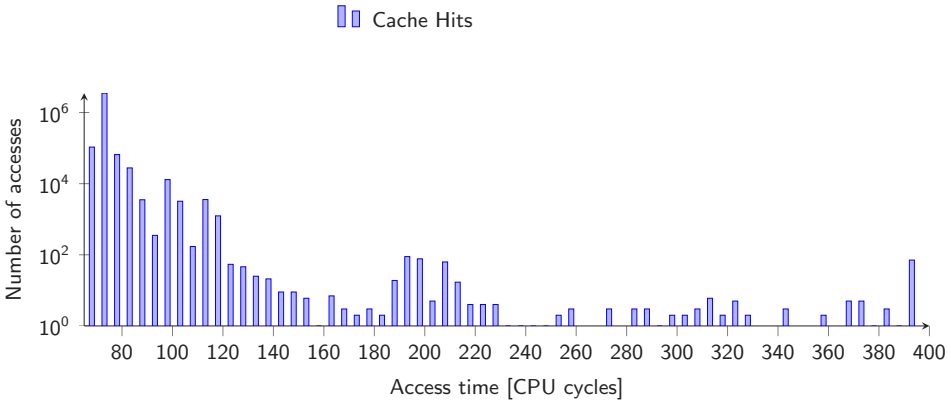



printf("%d", i);
printf("%d", i);
Cache miss
Cache hit
i
Request
Response

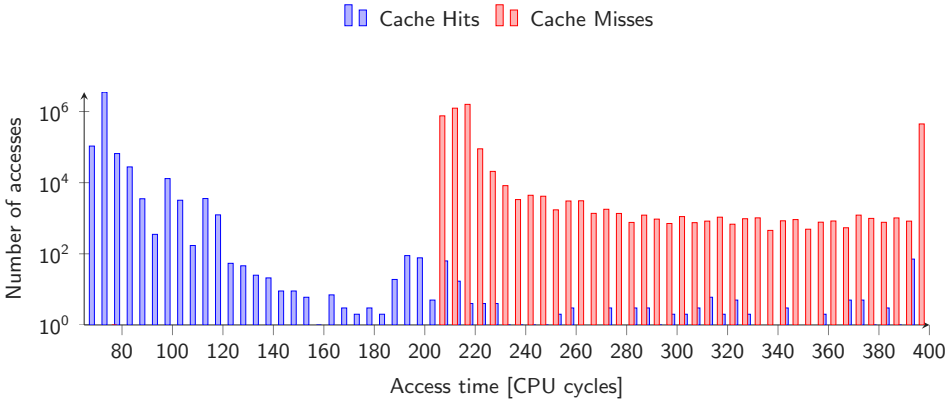

DRAM access,
slow
printf("%d", i);
printf("%d", i);
Cache miss
Cache hit
i
Request
Response

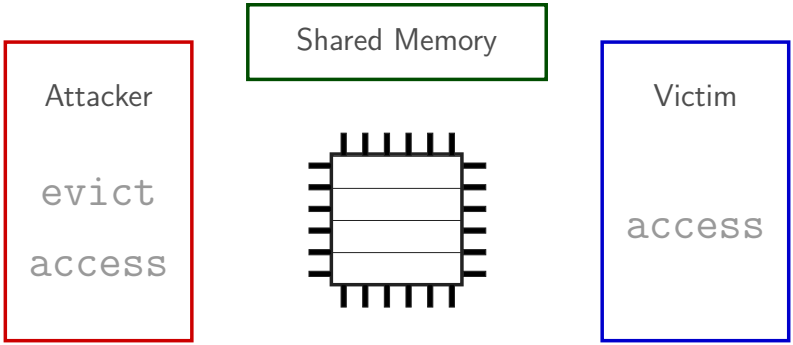

DRAM access,
slow
printf("%d", i);
printf("%d", i);
Cache miss
Cache hit
i
Request
Response
No DRAM access,
much faster

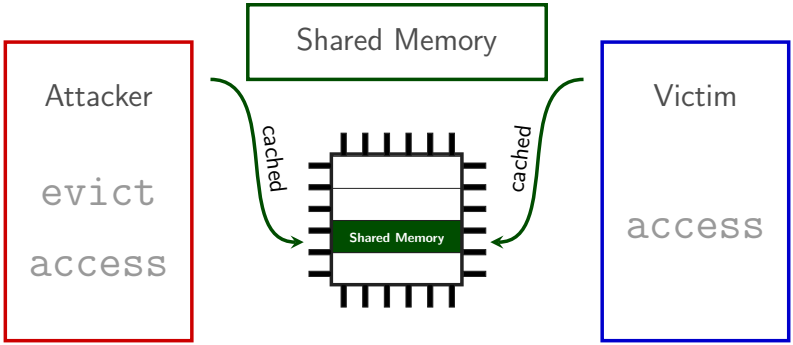
Caching speeds up Memory Accesses

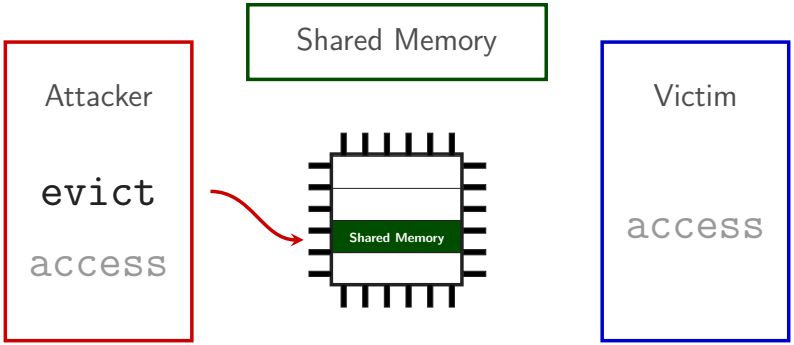


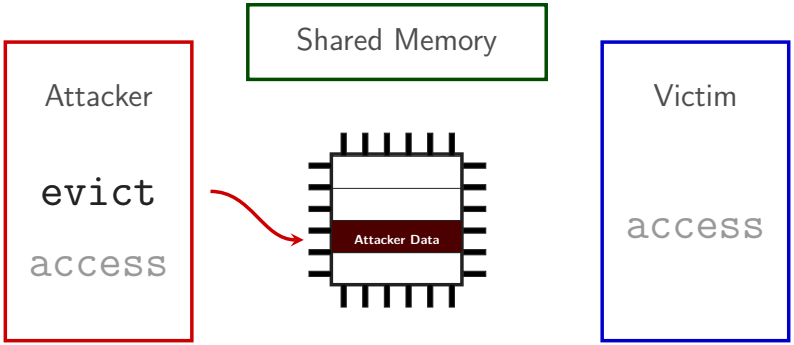
Caching speeds up Memory Accesses

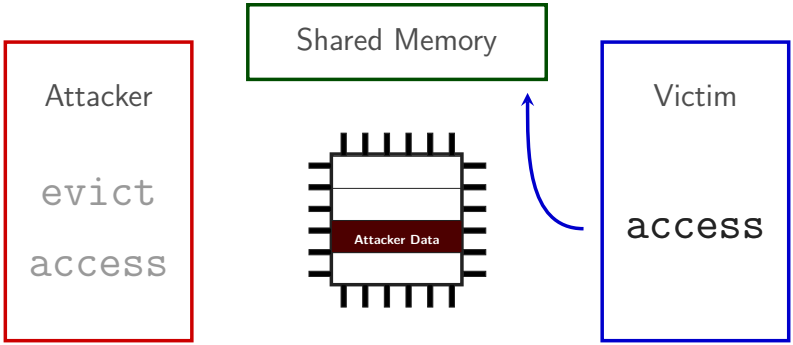


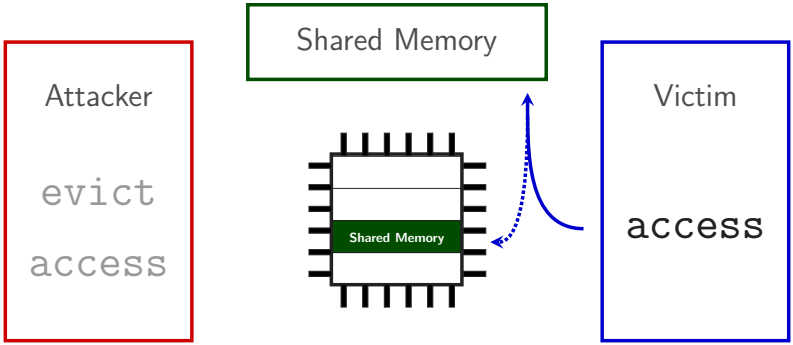

Shared Memory
Attacker
evict
access
Victim
access

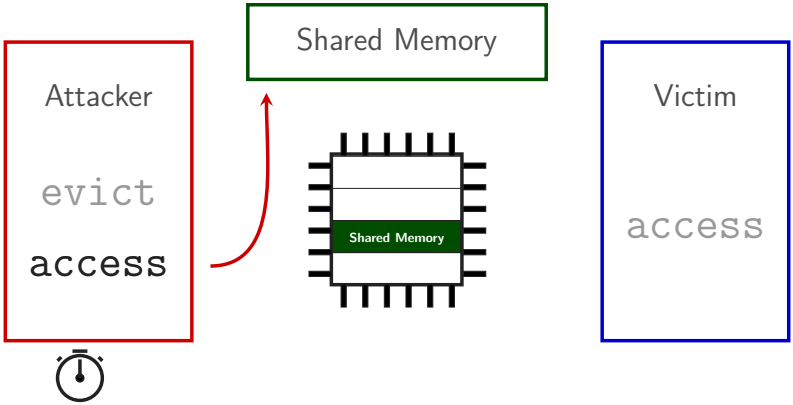

Shared Memory
Attacker
evict
access
cached
cached
Shared Memory
Victim
access

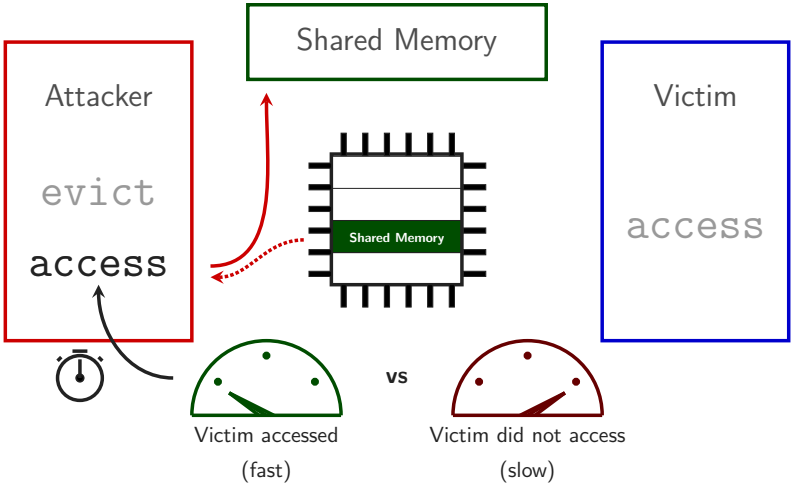

Shared Memory
Attacker
evict
access
Shared Memory
Victim
access


Shared Memory
Attacker
evict
access
Attacker Data
Victim
access


Shared Memory
Attacker
evict
access
Attacker Data
Victim
access


Shared Memory
Attacker
evict
access
Shared Memory
Victim
access


Shared Memory
Attacker
evict
access
Shared Memory
Victim
access


Shared Memory
Attacker
evict
access
Victim
access
Shared Memory
vs
Victim accessed
(fast)
Victim did not access
(slow)

- CPU tries to predict the future (branch predictor), ...

- CPU tries to predict the future (branch predictor), ...
  - ... based on events learned in the past

Speculative Execution

- CPU tries to predict the future (branch predictor), ...
  - ... based on events learned in the past
- Speculative execution of instructions

Speculative Execution

- CPU tries to predict the future (branch predictor), ...
  - ... based on events learned in the past
- Speculative execution of instructions
- If the prediction was correct, ...

Speculative Execution

- CPU tries to predict the future (branch predictor), ...
  - ... based on events learned in the past
- Speculative execution of instructions
- If the prediction was correct, ...
  - ... very fast

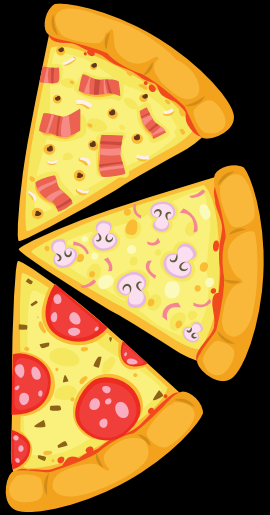
- CPU tries to predict the future (branch predictor), ...
  - ... based on events learned in the past
- Speculative execution of instructions
- If the prediction was correct, ...
  - ... very fast
  - otherwise: Discard results

PIZZA

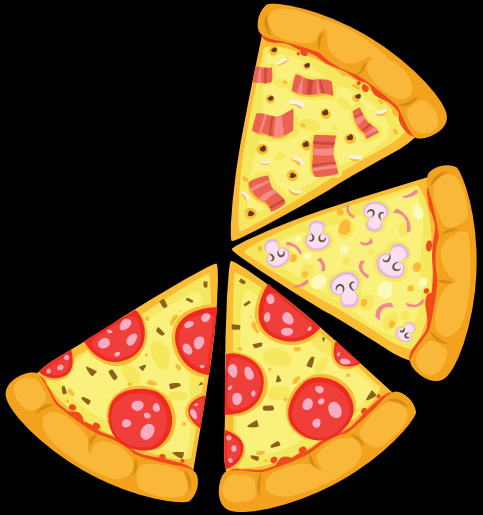
SPECIAL RECIPES

Prosciutto

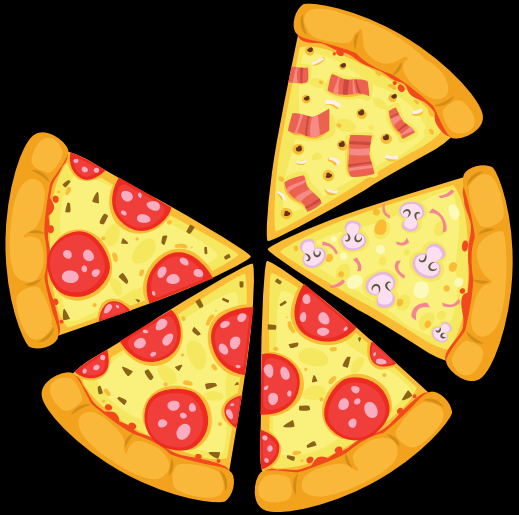
Funghi



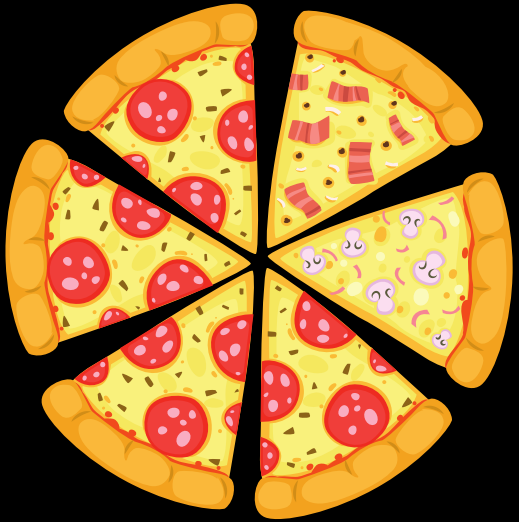
Diavolo



Diavolo



Diavolo



Diavolo


»A table for 6 please«

Speculative Cooking


»A table for 6 please«


PIZZA
SPECIAL RECIPES


PIZZA
SPECIAL RECIPES

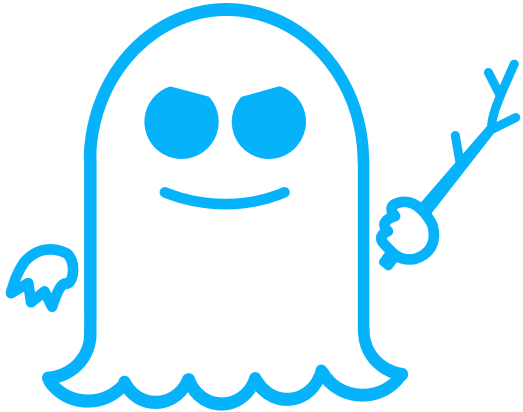

Pizza


PIZZA
SPECIAL RECIPES


Pizza


PIZZA
SPECIAL RECIPES


Pizza



SPECTRE

```
index = 0
```

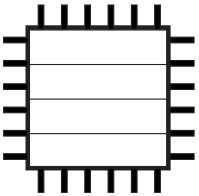
Shared Memory

	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

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

then → `glyph[data[index]]`

else → `{}`



Memory

D	data[0]
A	data[1]
T	data[2]
A	data[3]
K	
E	
Y	
...	

```
index = 0
```

Shared 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

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

then

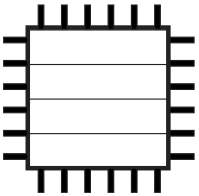
```
glyph[data[index]]
```

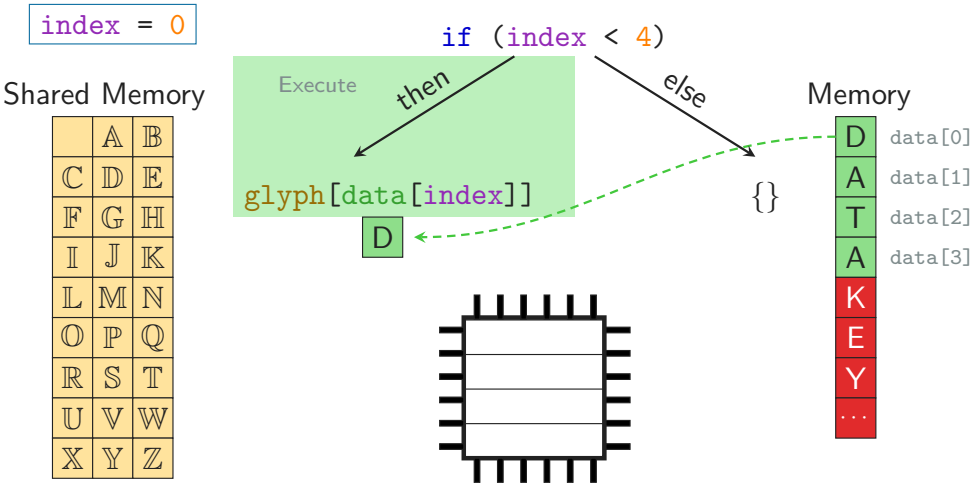
else

Speculate {}

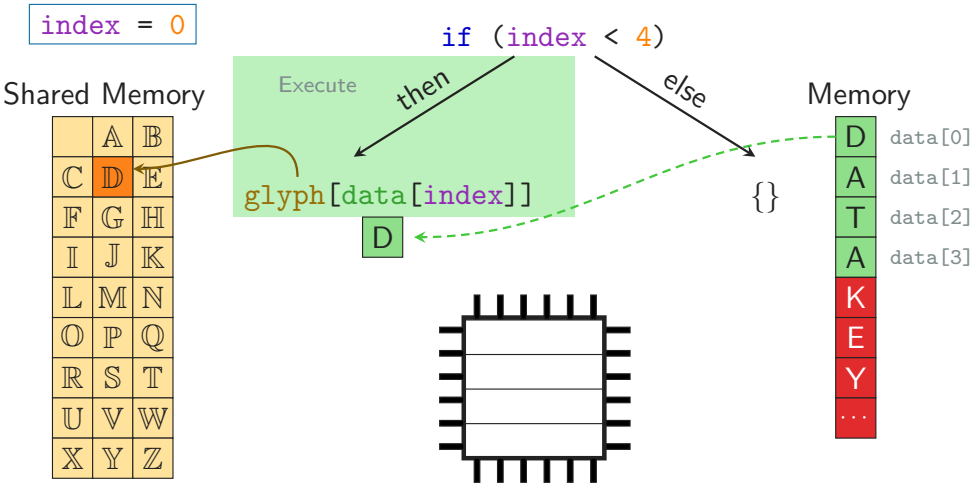
Memory

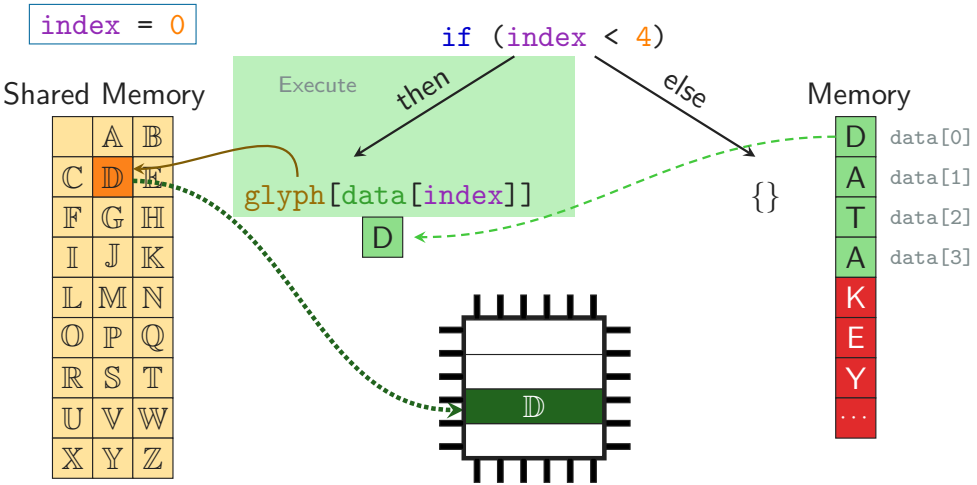
D	data[0]
A	data[1]
T	data[2]
A	data[3]
K	
E	
Y	
...	




index = 0
Shared Memory
if (index < 4)
Execute
then
else
glyph[data[index]]
D
{}
Memory
D
A
T
A
K
E
Y
...
data[0]
data[1]
data[2]
data[3]

Spectre-PHT (aka Spectre Variant 1)




index = 0
if (index < 4)
then
else
Execute
glyph[data[index]]
D
{}
Shared Memory
Memory
data[0]
data[1]
data[2]
data[3]

```
index = 1
```

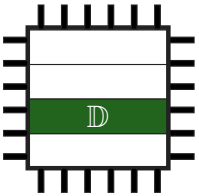
Shared Memory

	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

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

then → `glyph[data[index]]`

else → `{}`



Memory

D	data[0]
A	data[1]
T	data[2]
A	data[3]
K	
E	
Y	
...	

Spectre-PHT (aka Spectre Variant 1)

```
index = 1
```

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

Speculate

then

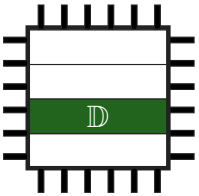
```
glyph[data[index]]
```

else

{}

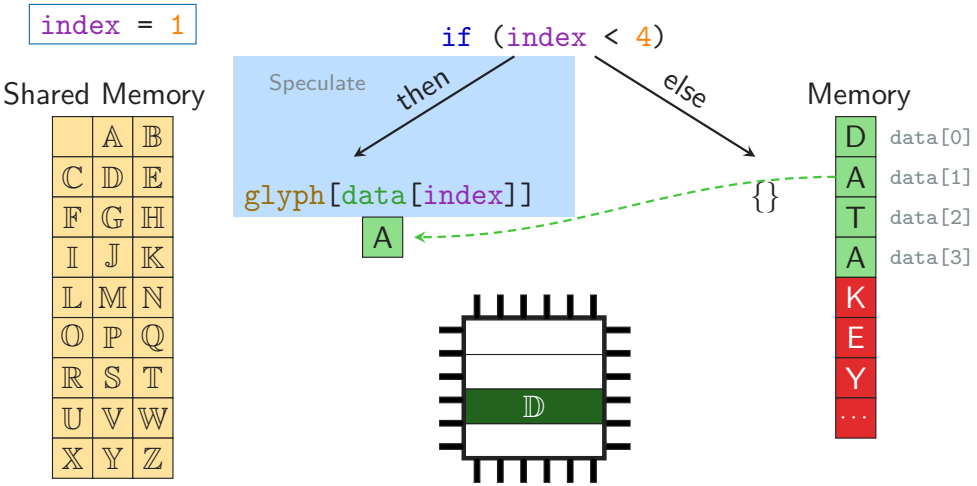
Shared Memory

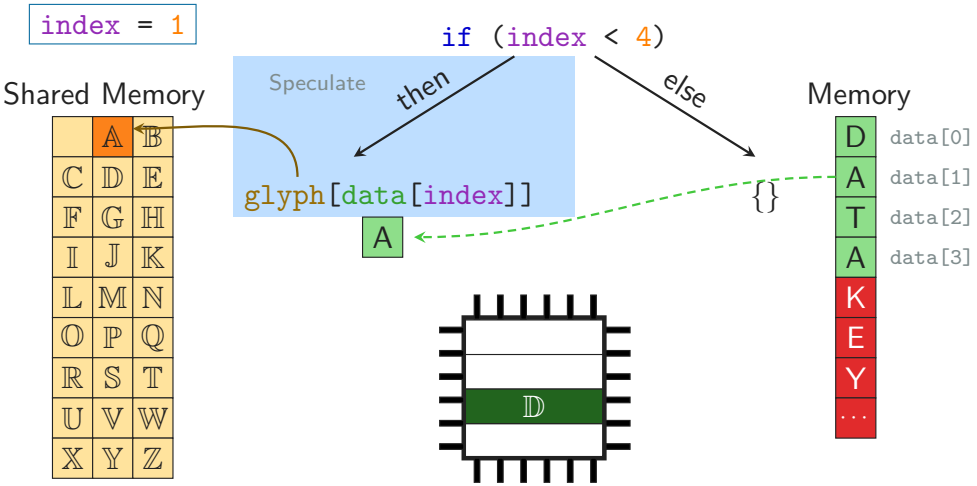
	A	B
C	D	E
F	G	H
I	J	K
L	M	N
O	P	Q
R	S	T
U	V	W
X	Y	Z

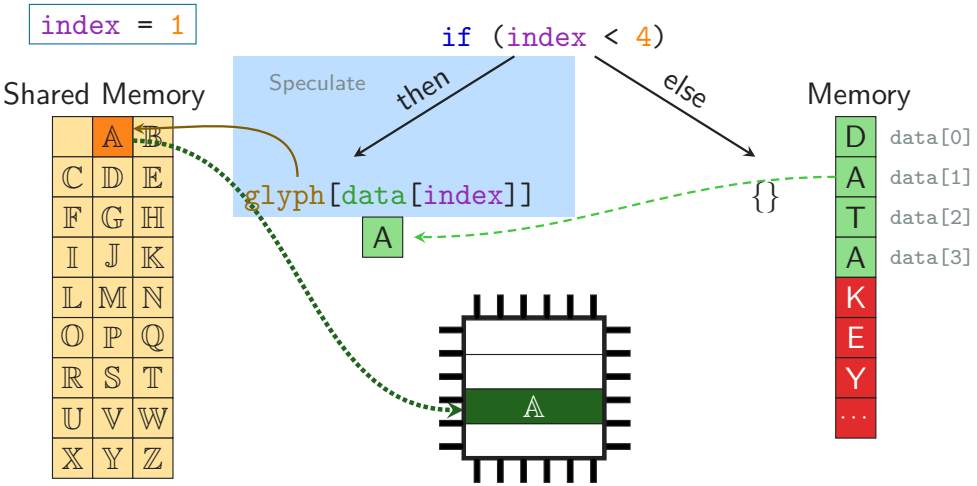


Memory

D	data[0]
A	data[1]
T	data[2]
A	data[3]
K	
E	
Y	
...	


index = 1
Shared Memory
if (index < 4)
Speculate
then
else
glyph[data[index]]
A
{}
Memory
D
A
T
A
K
E
Y
...
data[0]
data[1]
data[2]
data[3]
𝔻


index = 1
if (index < 4)
Speculate
then
else
Shared Memory
Memory
glyph[data[index]]
A
{}
data[0]
data[1]
data[2]
data[3]
D
A
T
A
K
E
Y
...
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
D


index = 1
if (index < 4)
Speculate
then
else
Shared Memory
Memory
glyph[data[index]]
A
{}
D
A
T
A
K
E
Y
...
data[0]
data[1]
data[2]
data[3]
A

```
index = 1
```

Shared 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

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

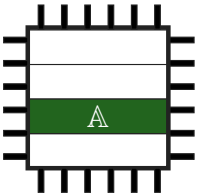
Execute

then

```
glyph[data[index]]
```

else

{}



Memory

D	data[0]
A	data[1]
T	data[2]
A	data[3]
K	
E	
Y	
...	

```
index = 2
```

Shared 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

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

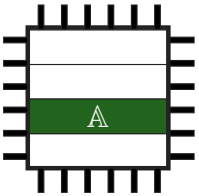
Speculate

then

```
glyph[data[index]]
```

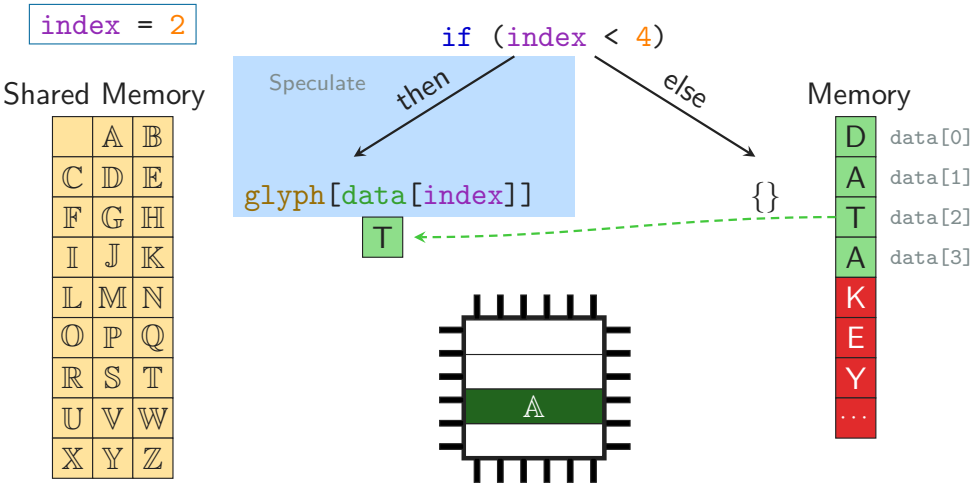
else

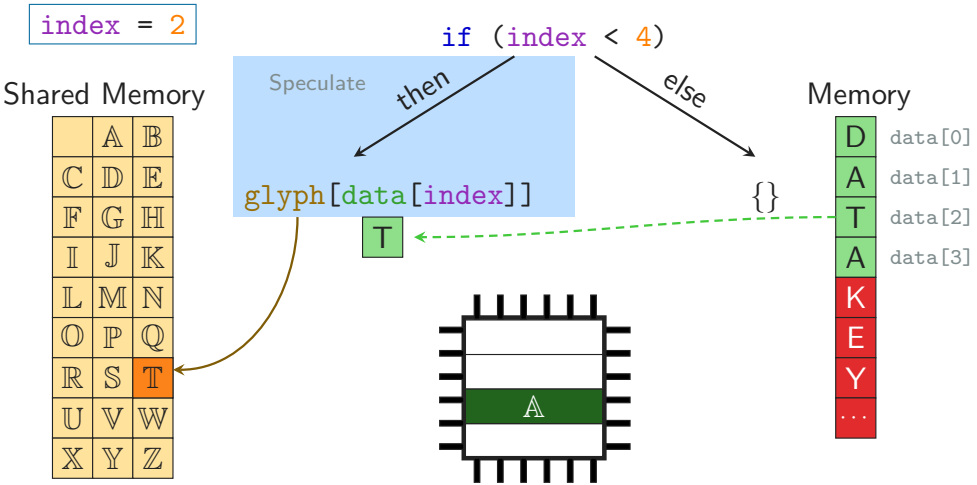
{}

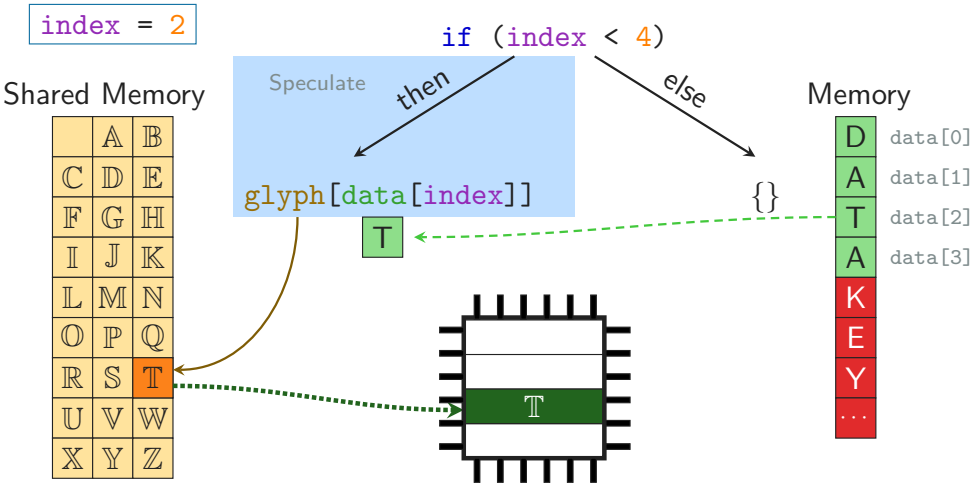


Memory

D	data[0]
A	data[1]
T	data[2]
A	data[3]
K	
E	
Y	
...	


index = 2
Shared Memory
if (index < 4)
Speculate
then
else
glyph[data[index]]
T
{}
Memory
D
A
T
A
K
E
Y
...
data[0]
data[1]
data[2]
data[3]
A


index = 2
if (index < 4)
Speculate
then
else
Shared Memory
glyph[data[index]]
T
{}
Memory
D
A
T
A
K
E
Y
...
data[0]
data[1]
data[2]
data[3]
A


index = 2
if (index < 4)
Speculate
then
else
Shared Memory
glyph[data[index]]
T
{}
Memory
D
A
T
A
K
E
Y
...
data[0]
data[1]
data[2]
data[3]
T

```
index = 2
```

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

then

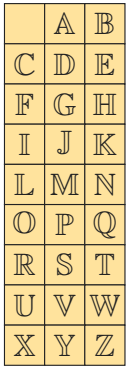
Execute

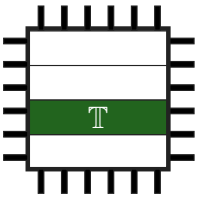
```
glyph[data[index]]
```

else

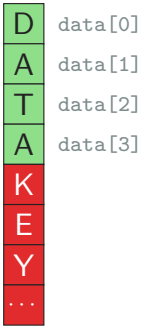
{}

Shared Memory





Memory



```
index = 3
```

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

Speculate

then

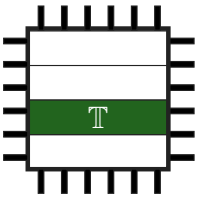
```
glyph[data[index]]
```

else

{}

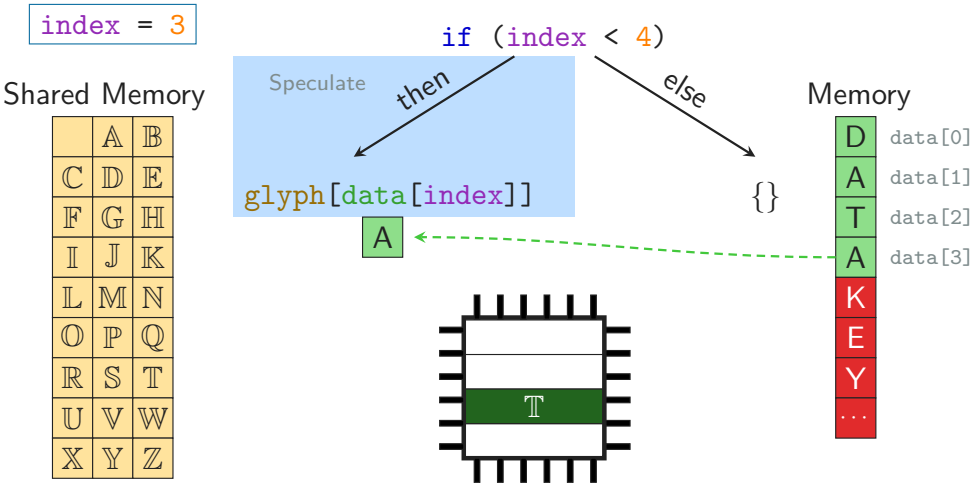
Shared Memory

	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

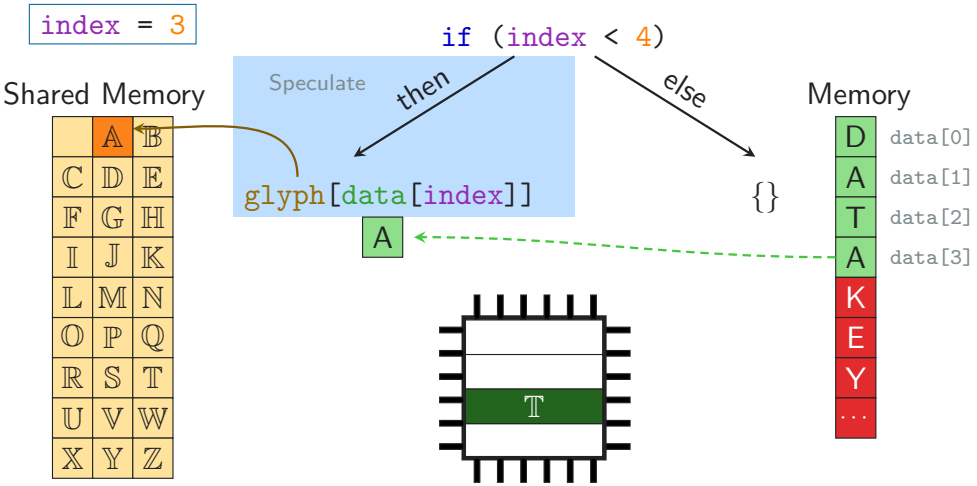


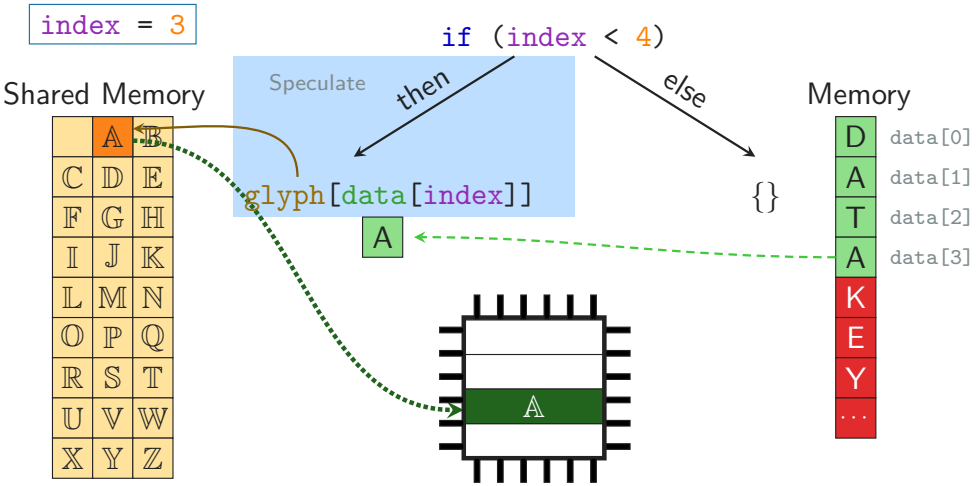
Memory

D	data[0]
A	data[1]
T	data[2]
A	data[3]
K	
E	
Y	
...	


index = 3
Shared Memory
if (index < 4)
Speculate
then
else
glyph[data[index]]
A
{}
Memory
D
A
T
A
K
E
Y
...
data[0]
data[1]
data[2]
data[3]

Spectre-PHT (aka Spectre Variant 1)




index = 3
if (index < 4)
Speculate
then
else
Shared Memory
Memory
glyph[data[index]]
{}
A
A
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
D
A
T
A
K
E
Y
...
data[0]
data[1]
data[2]
data[3]

```
index = 3
```

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

then

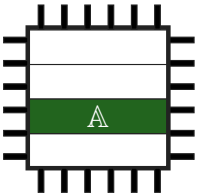
Execute

```
glyph[data[index]]
```

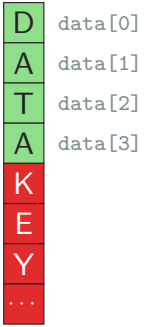
else

{}

Shared Memory



Memory



```
index = 4
```

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

Speculate

then

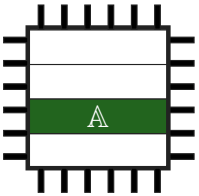
```
glyph[data[index]]
```

else

{}

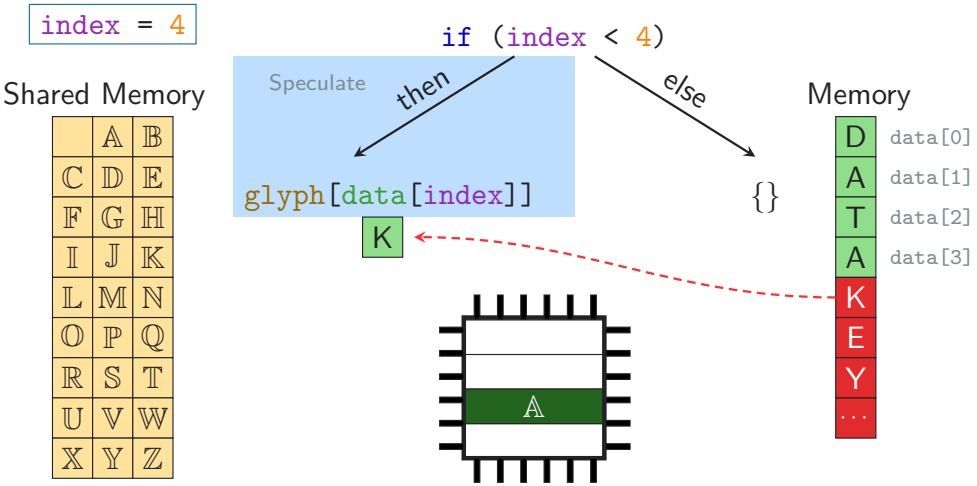
Shared Memory

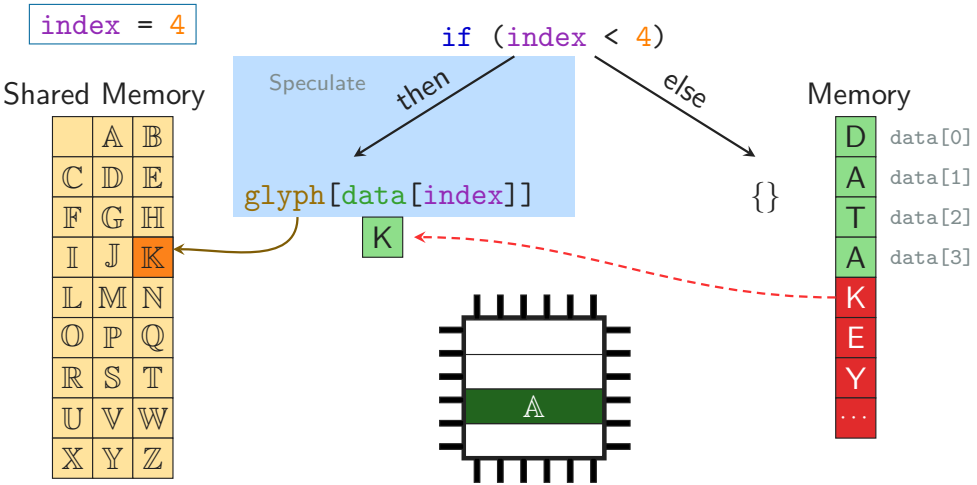
	A	B
C	D	E
F	G	H
I	J	K
L	M	N
O	P	Q
R	S	T
U	V	W
X	Y	Z

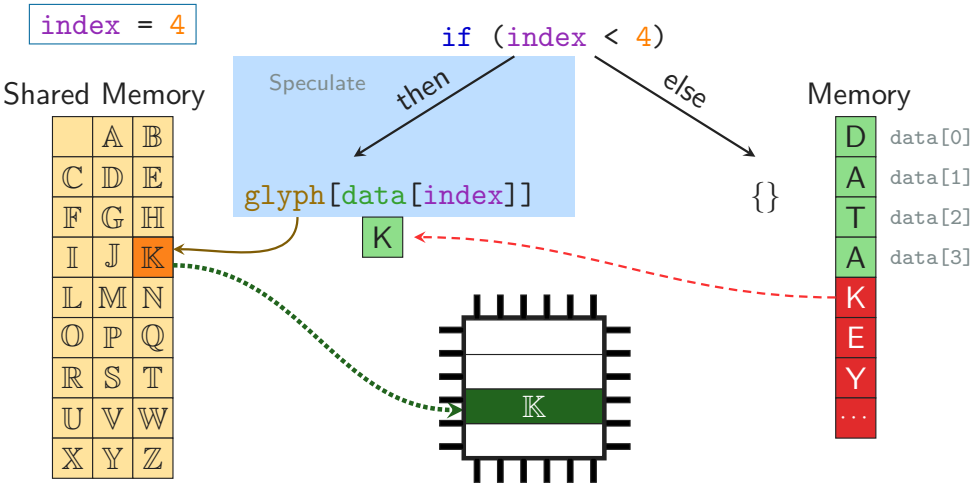


Memory

D	data[0]
A	data[1]
T	data[2]
A	data[3]
K	
E	
Y	
...	


index = 4
if (index < 4)
Speculate
then
else
Shared Memory
glyph[data[index]]
{}
Memory
K
D
A
T
A
K
E
Y
...
data[0]
data[1]
data[2]
data[3]
A


index = 4
if (index < 4)
Speculate
then
else
Shared Memory
glyph[data[index]]
K
{}
Memory
D
A
T
A
K
E
Y
...
data[0]
data[1]
data[2]
data[3]
A


index = 4
if (index < 4)
Speculate
then
else
Shared Memory
Memory
glyph[data[index]]
K
{}
data[0]
data[1]
data[2]
data[3]
D
A
T
A
K
E
Y
...

```
index = 4
```

Shared 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

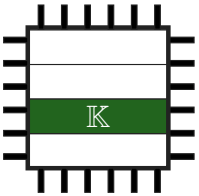
```
if (index < 4)
```

then

```
glyph[data[index]]
```

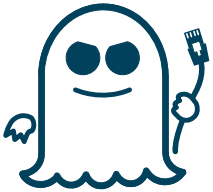
else

Execute {}

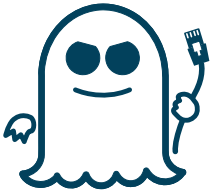


Memory

D	data[0]
A	data[1]
T	data[2]
A	data[3]
K	
E	
Y	
...	



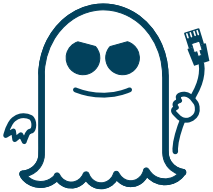
We want to build a Spectre attack which...



We want to build a Spectre attack which...

- is capable of leaking secrets from a remote system

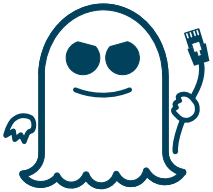
The goal



We want to build a Spectre attack which...

- is capable of leaking secrets from a remote system
- has neither physical access nor code execution on system

The goal



We want to build a Spectre attack which...

- is capable of leaking secrets from a remote system
- has neither physical access nor code execution on system
- does not rely on software vulnerabilities

CVSS v3 for CVE-2017-5753 (Spectre)

Attack Vector

Network	Adjacent Network	**Local**	Physical

CVSS v3 for CVE-2017-5753 (Spectre)

Attack Vector

Network	Adjacent Network	**Local**	Physical

Attack Complexity

Low	**High**

CVSS v3 for CVE-2017-5753 (Spectre)

Attack Vector

Network	Adjacent Network	**Local**	Physical

Attack Complexity

Low	**High**

Privilege Required

None	**Low**	High

CVSS v3 for CVE-2017-5753 (Spectre)

Attack Vector

Network	Adjacent Network	**Local**	Physical

Attack Complexity

Low	**High**

Privilege Required

None	**Low**	High

User Interaction

None	Required

Spectre without code execution is complicated

Spectre without code execution is complicated

- Which branch can be exploited

Spectre without code execution is complicated

- Which branch can be exploited
- Cannot observe the cache state

Spectre without code execution is complicated

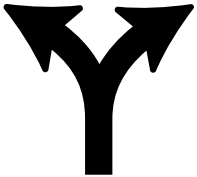
- Which branch can be exploited
- Cannot observe the cache state
- Spectre gadgets will be different

Spectre without code execution is complicated

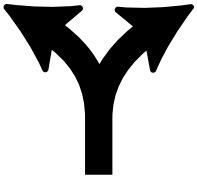
- Which branch can be exploited
- Cannot observe the cache state
- Spectre gadgets will be different
- No timing measurement on the attacked system

Spectre without code execution is complicated

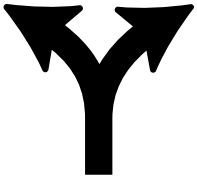
- Which branch can be exploited
- Cannot observe the cache state
- Spectre gadgets will be different
- No timing measurement on the attacked system
- How to select the data to leak



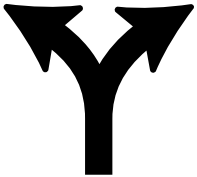
- No code can be injected



- No code can be injected
- Public interface (API) accessing data



- No code can be injected
- Public interface (API) accessing data
- Branches in API can be mistrained remotely



- No code can be injected
- Public interface (API) accessing data
- Branches in API can be mistrained remotely
- Attacker only calls the API via network requests

```
def check_user_privileges(user_id):
  [...]
  if user_id < len(users):
    if test_bit(privileges, user_id) == True:
      admin = True

  return SUCCESS
```

```
def check_user_privileges(user_id):
  [...]
  if user_id < len(users):
    if test_bit(privileges, user_id) == True:
      admin = True

  return SUCCESS
```

Bounds check

API Example

```
def check_user_privileges(user_id):
  [...]
  if user_id < len(users):
    if test_bit(privileges, user_id) == True:
      admin = True

  return SUCCESS
```

Bounds check

Speculative out-of-bounds read

API Example

```
def is_admin():
  return admin
```

API Example

```
def is_admin():
  return admin
```

```
def is_admin():
  return admin
```

- If bit in array was set $\rightarrow$ `admin` is cached

API Example

```
def is_admin():
  return admin
```

- If bit in array was set → `admin` is cached
- If bit was not set → `admin` is not cached

API Example

```
def is_admin():
  return admin
```

- If bit in array was set → `admin` is cached
- If bit was not set → `admin` is not cached
- Observe cache state via function execution time

- Cannot measure time directly on the attacked system

Timing Measurement

- Cannot measure time directly on the attacked system
- Network latency depends on API execution time

Timing Measurement

- Cannot measure time directly on the attacked system
- Network latency depends on API execution time

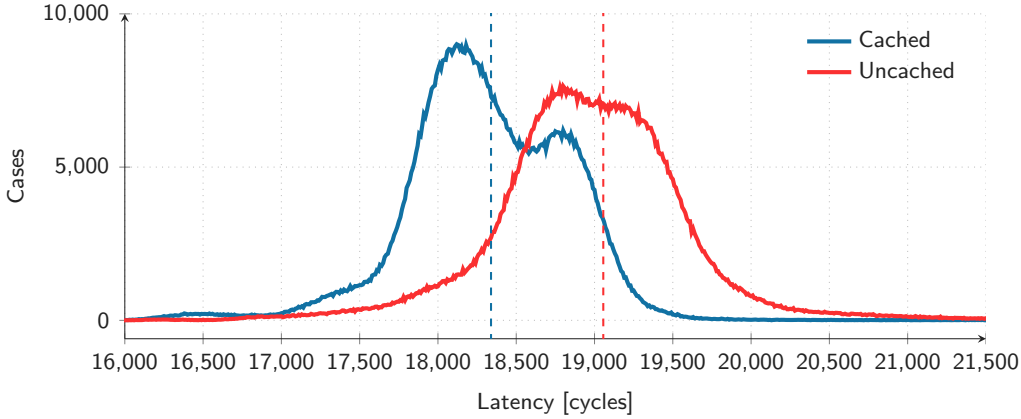
→ Measure the network roundtrip time

Timing Measurement

- Cannot measure time directly on the attacked system
- Network latency depends on API execution time

→ Measure the network roundtrip time

- Reveals whether the variable is cached


10,000
5,000
0
Cases
Cached
Uncached
16,000
16,500
17,000
17,500
18,000
18,500
19,000
19,500
20,000
20,500
21,000
21,500
Latency [cycles]

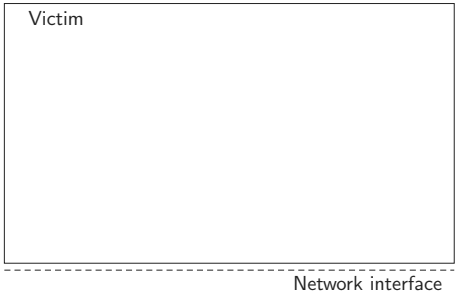
- After measuring variable is always cached

- After measuring variable is always cached
- How do we evict the variable?

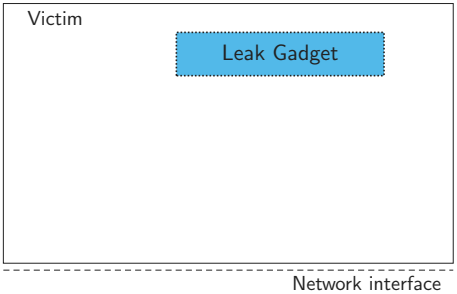
- After measuring variable is always cached
- How do we evict the variable?
- Constantly evict the cache via a file download

- After measuring variable is always cached
- How do we evict the variable?
- Constantly evict the cache via a file download
- Thrash+Reload → crude form of Evict+Reload

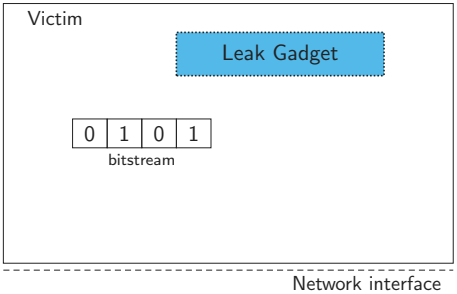
NetSpectre - The Big Picture



NetSpectre - The Big Picture

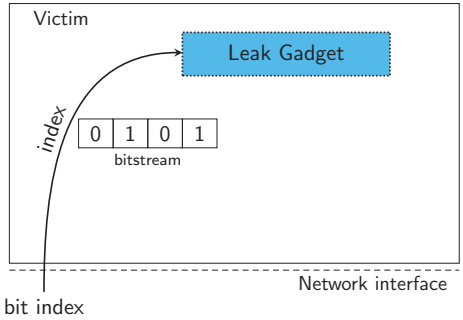



```
if (x < bitstream_length)
    if(bitstream[x])
        flag = true
```



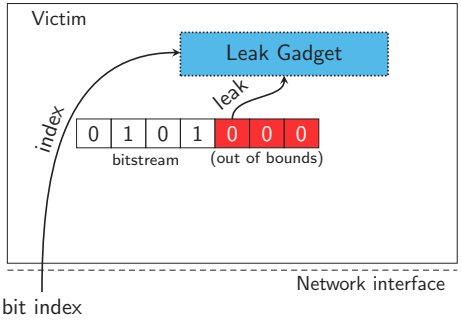

```
if (x < bitstream_length)
   if(bitstream[x])
      flag = true
```

NetSpectre - The Big Picture



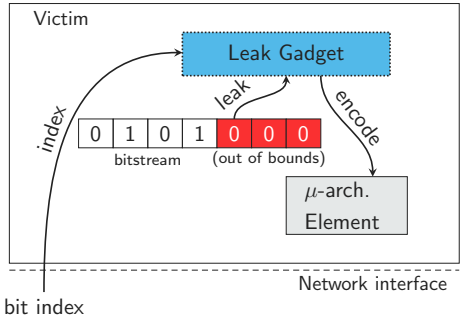

```
if (x < bitstream_length)
    if(bitstream[x])
        flag = true
```

NetSpectre - The Big Picture



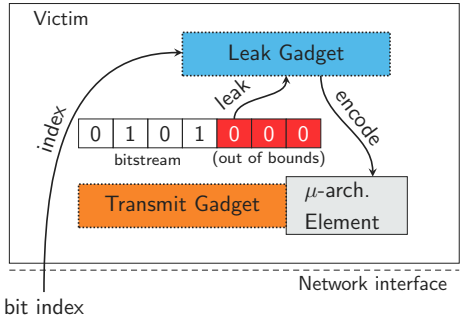

```
if (x < bitstream_length)
    if(bitstream[x])
        flag = true
```

NetSpectre - The Big Picture



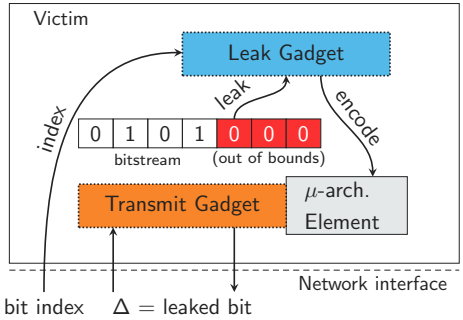

```
if (x < bitstream_length)
    if(bitstream[x])
        flag = true
```

NetSpectre - The Big Picture



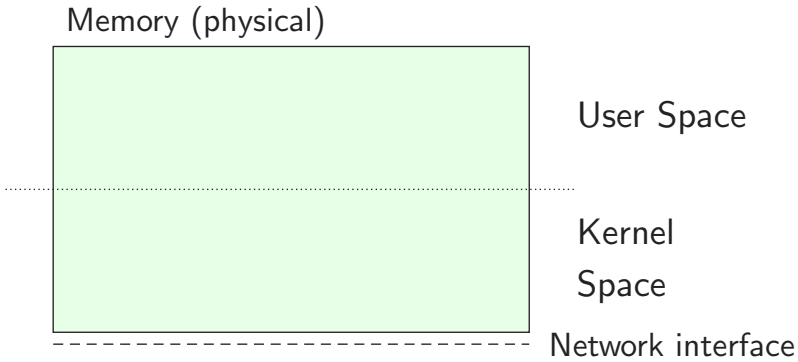

```
if (x < bitstream_length)
    if(bitstream[x])
        flag = true
```

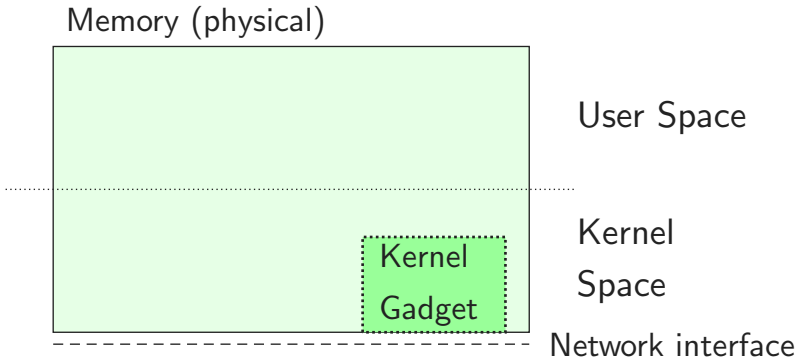
```
send(flag)
```

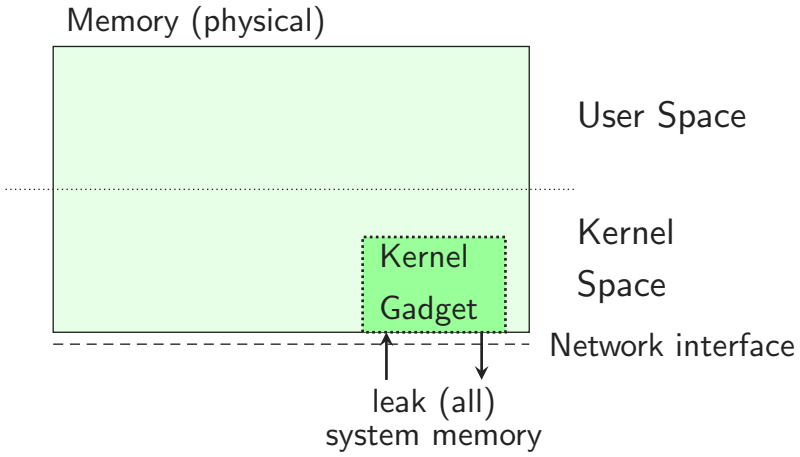


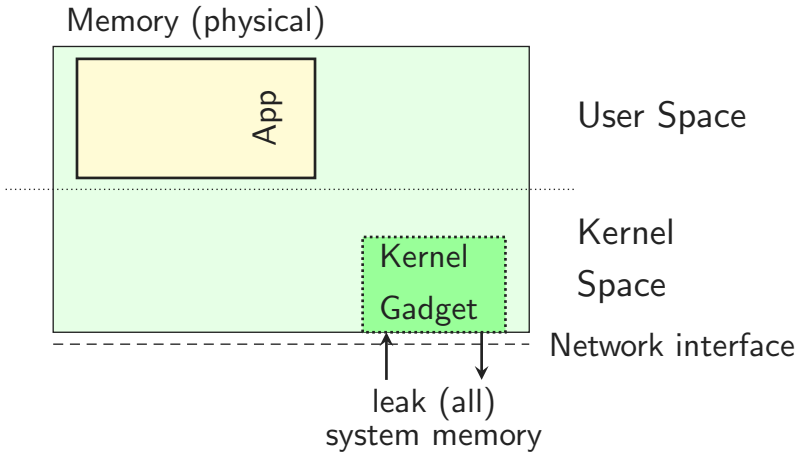

```
if (x < bitstream_length)
    if(bitstream[x])
        flag = true
```

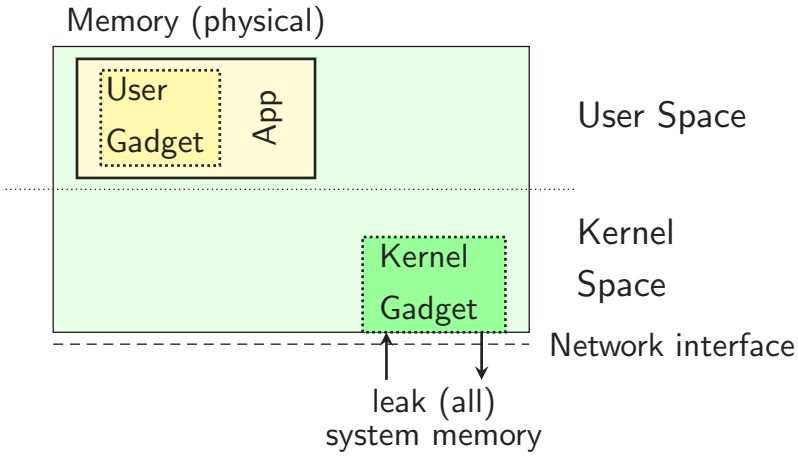
```
send(flag)
```

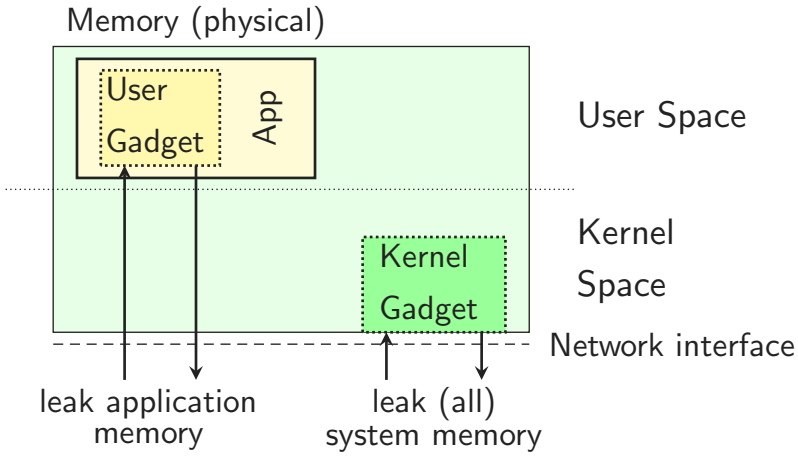

Memory (physical)
User Space
Kernel
Space
Network interface

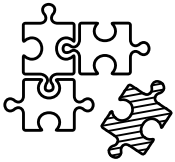

Memory (physical)
User Space
Kernel
Space
Kernel
Gadget
Network interface


Memory (physical)
User Space
Kernel
Space
Kernel
Gadget
Network interface
leak (all)
system memory


Memory (physical)
App
User Space
Kernel
Space
Kernel
Gadget
Network interface
leak (all)
system memory

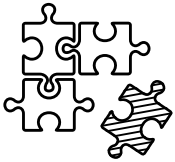

Memory (physical)
User
Gadget
App
User Space
Kernel
Space
Kernel
Gadget
Network interface
leak (all)
system memory


Memory (physical)
User
Gadget
App
User Space
Kernel
Gadget
Kernel
Space
Network interface
leak application
memory
leak (all)
system memory

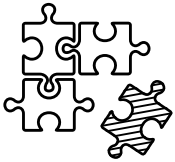


- Mistrain branch predictor with in-bounds requests

Combining Everything

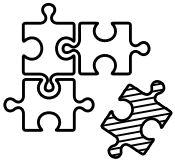


- Mistrain branch predictor with in-bounds requests
- Evict everything from cache via file download



- Mistrain branch predictor with in-bounds requests
- Evict everything from cache via file download
- Leak a bit: do nothing (‘0’) or cache a memory location (‘1’)

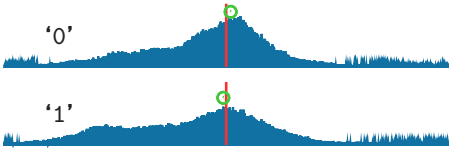
Combining Everything



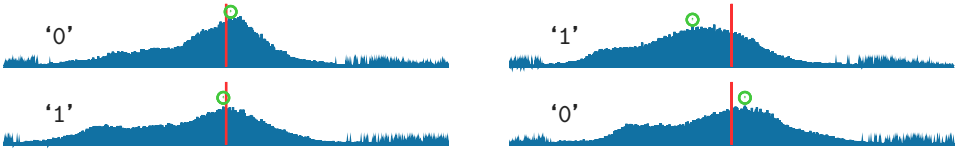
- Mistrain branch predictor with in-bounds requests
- Evict everything from cache via file download
- Leak a bit: do nothing (‘0’) or cache a memory location (‘1’)
- Measure function latency which uses the memory location

Leaking byte ’d’ (0)

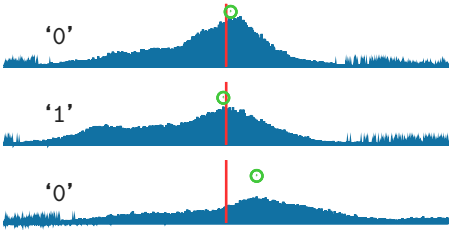
Leaking byte ’d’ (01)

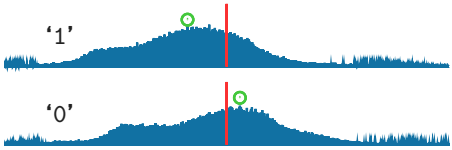


Leaking byte ’d’ (011)

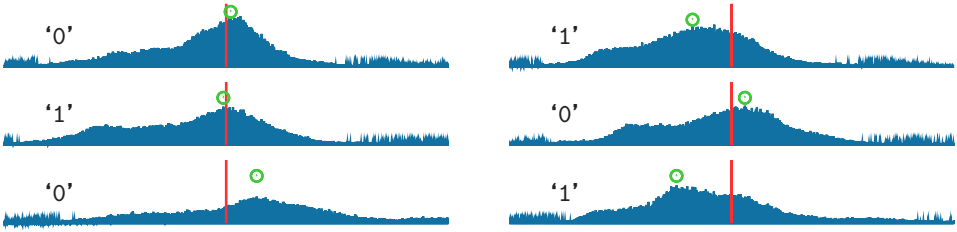


Leaking byte ’d’ (0110)

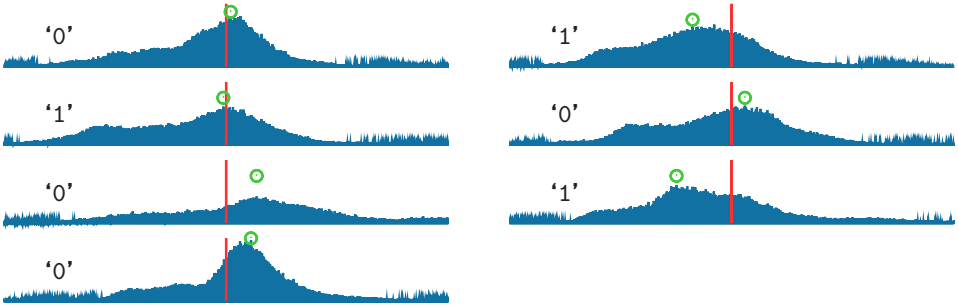




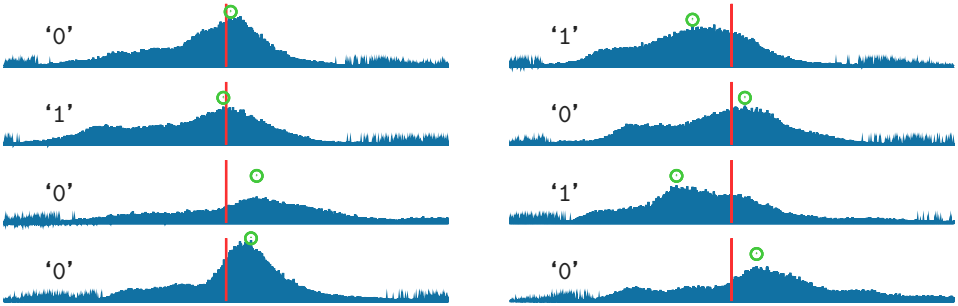
Leaking byte 'd' (01100)



Leaking byte 'd' (011001)



Leaking byte 'd' (0110010)



Leaking byte 'd' (01100100)

Attack Targets

- Several possible attack targets

Attack Targets

- Several possible attack targets
- Different impacts depending on target

Attack Targets

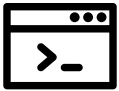
- Several possible attack targets
- Different impacts depending on target

Web/FTP Servers
(user gadget)

Attack Targets

- Several possible attack targets
- Different impacts depending on target

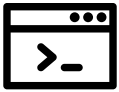
Web/FTP Servers
(user gadget)



SSH Daemons
(user gadget)

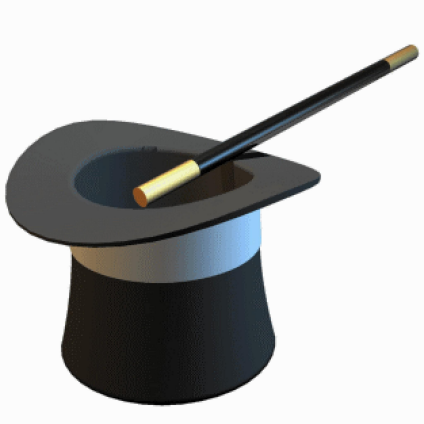
- Several possible attack targets
- Different impacts depending on target

Web/FTP Servers
(user gadget)



SSH Daemons
(user gadget)

Network Drivers
(kernel gadget)

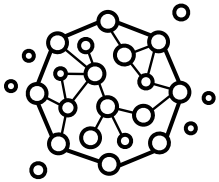


- Finding Spectre gadgets is still an open problem

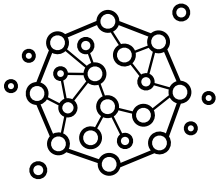
- Finding Spectre gadgets is still an open problem
- Out of all papers, only 4 show real-world gadgets

- Finding Spectre gadgets is still an open problem
- Out of all papers, only 4 show real-world gadgets
- Among them, only 2 Spectre-PHT (v1) gadgets

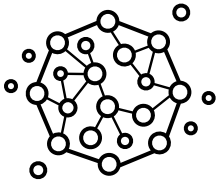
- Finding Spectre gadgets is still an open problem
- Out of all papers, only 4 show real-world gadgets
- Among them, only 2 Spectre-PHT (v1) gadgets
- Still no fully automated approach



- Linux kernel uses static code analysis

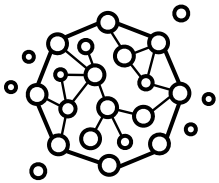


- Linux kernel uses static code analysis
- High false positive rate



- Linux kernel uses static code analysis
- High false positive rate

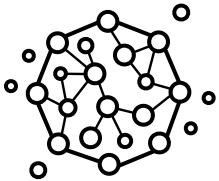
→ Out of 736 reports only 15 real gadgets



- Linux kernel uses static code analysis
- High false positive rate

→ Out of 736 reports only 15 real gadgets

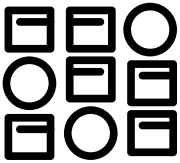
- Ongoing effort, $>$ 100 patches applied to Linux kernel



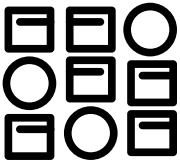
- Linux kernel uses static code analysis
- High false positive rate

→ Out of 736 reports only 15 real gadgets

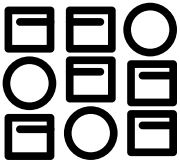
- Ongoing effort, $>$ 100 patches applied to Linux kernel
- $>$ 930 Spectre patches in open-source projects



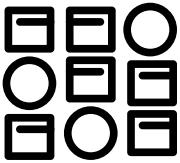
- Built 21 toy examples, 18 containing Spectre gadgets



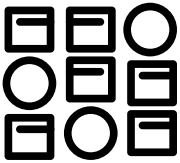
- Built 21 toy examples, 18 containing Spectre gadgets
- We created two static approaches on detecting (Net)Spectre gadgets



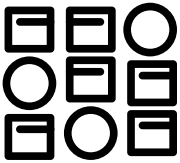
- Built 21 toy examples, 18 containing Spectre gadgets
- We created two static approaches on detecting (Net)Spectre gadgets
  - Coccinelle (Matching the code pattern)



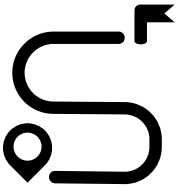
- Built 21 toy examples, 18 containing Spectre gadgets
- We created two static approaches on detecting (Net)Spectre gadgets
  - Coccinelle (Matching the code pattern)
  - Python Capstone (Matching the binary pattern)



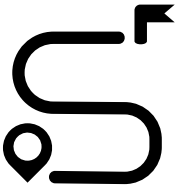
- Built 21 toy examples, 18 containing Spectre gadgets
- We created two static approaches on detecting (Net)Spectre gadgets
  - Coccinelle (Matching the code pattern)
  - Python Capstone (Matching the binary pattern)
- All gadgets were detected, only 3 false positives



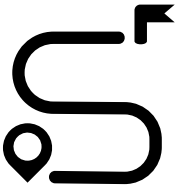
- Built 21 toy examples, 18 containing Spectre gadgets
- We created two static approaches on detecting (Net)Spectre gadgets
  - Coccinelle (Matching the code pattern)
  - Python Capstone (Matching the binary pattern)
- All gadgets were detected, only 3 false positives
- Adapted oo7 approach to masscan open-source software



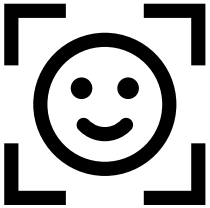
- Taint Tracking $\leftrightarrow$ mark all input as evil



- Taint Tracking $\leftrightarrow$ mark all input as evil
- If input x flows into branch $x < size$, the branch is marked as tainted

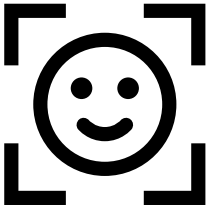


- Taint Tracking $\leftrightarrow$ mark all input as evil
- If input x flows into branch $x < size$, the branch is marked as tainted
- $\exists$ a memory access relative within an array in a time window, report it as susceptible



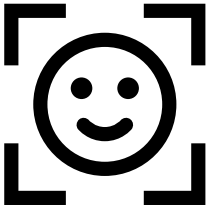
- Not clear how a Spectre gadget can look like

Challenges in Identifying Gadgets



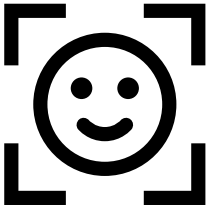
- Not clear how a Spectre gadget can look like
- Potentially many different forms

Challenges in Identifying Gadgets



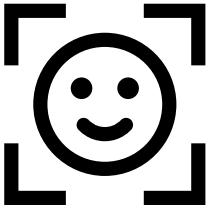
- Not clear how a Spectre gadget can look like
- Potentially many different forms
- Can be scattered over many instructions

Challenges in Identifying Gadgets



- Not clear how a Spectre gadget can look like
- Potentially many different forms
- Can be scattered over many instructions
- Similar to finding ROP chains

Challenges in Identifying Gadgets

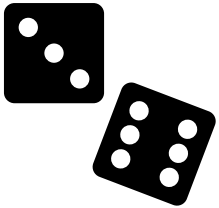


- Not clear how a Spectre gadget can look like
- Potentially many different forms
- Can be scattered over many instructions
- Similar to finding ROP chains
- While searching, discovered novel type of gadget

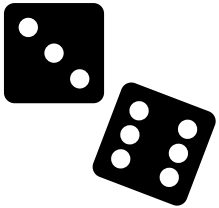
- No indirection, simple array access

- No indirection, simple array access

```
if (x < array_length)
  y = array[x];
```

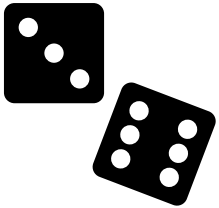


- What to do with weaker gadgets?



- What to do with weaker gadgets?

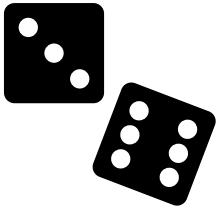
→ Break ASLR



- What to do with weaker gadgets?

→ Break ASLR

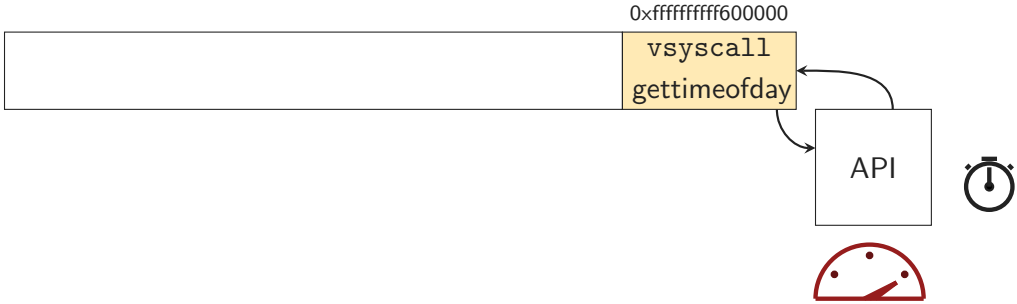
- Not relevant for local Spectre attacks



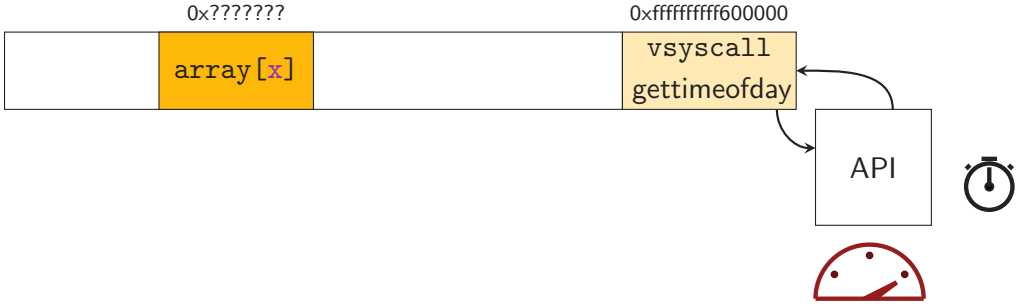
- What to do with weaker gadgets?

→ Break ASLR

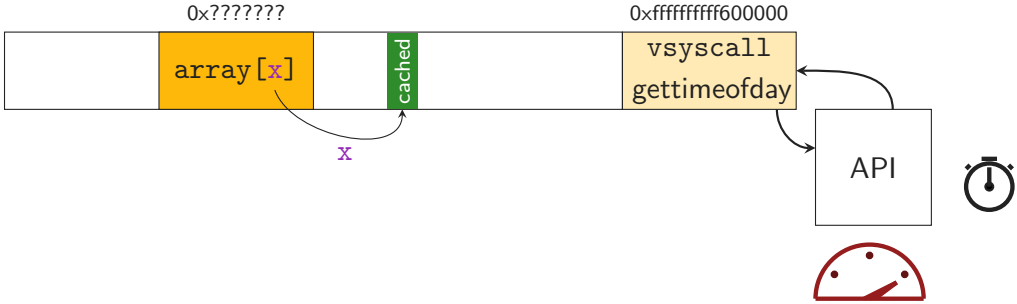
- Not relevant for local Spectre attacks
- Valuable in a remote scenario


0xffffffffff600000
vsyscall
gettimeofday
API

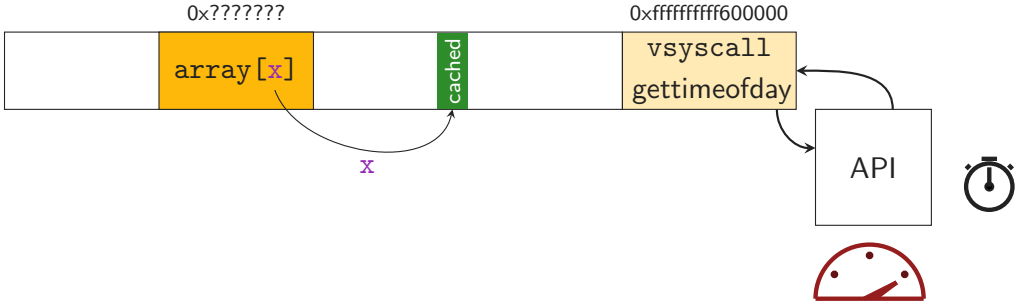
ASLR Break Idea



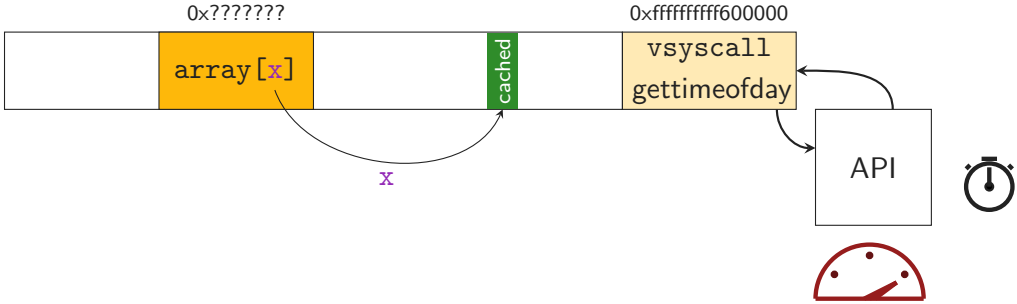
ASLR Break Idea



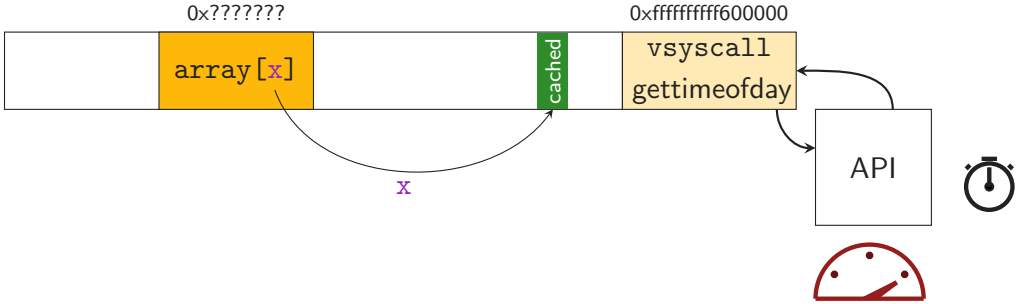
ASLR Break Idea

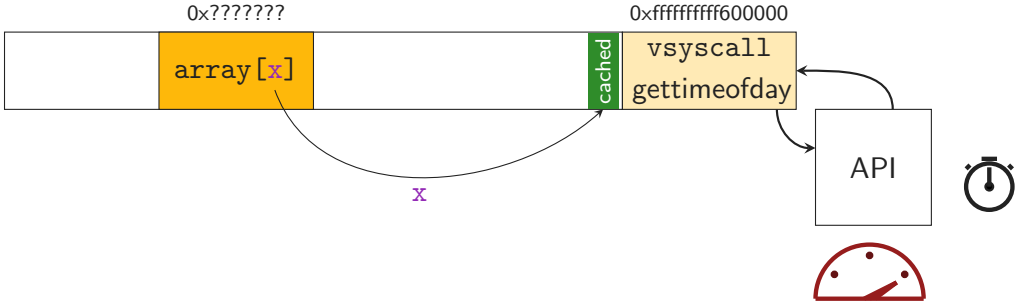


ASLR Break Idea

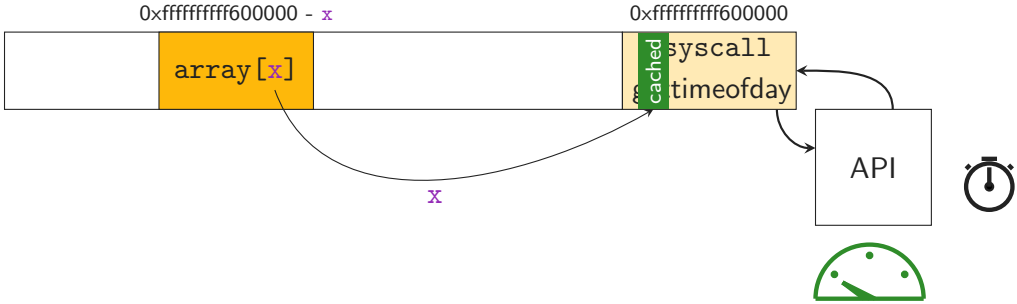


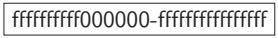
ASLR Break Idea

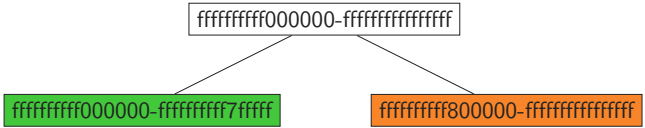


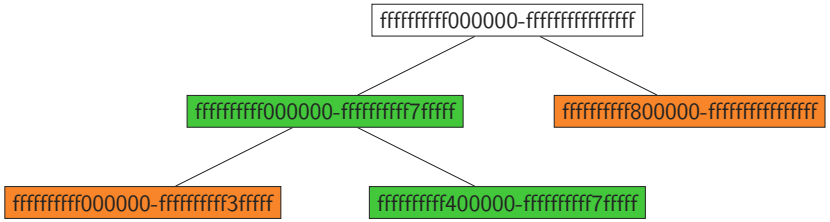

0x???????
0xffffffffff600000
array[x]
cached
vsyscall
gettimeofday
API
x

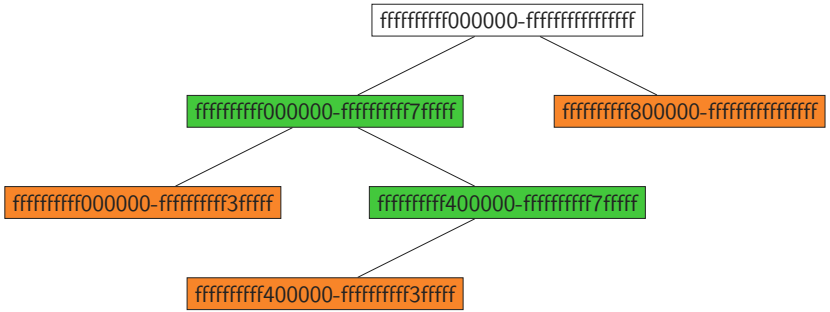
ASLR Break Idea

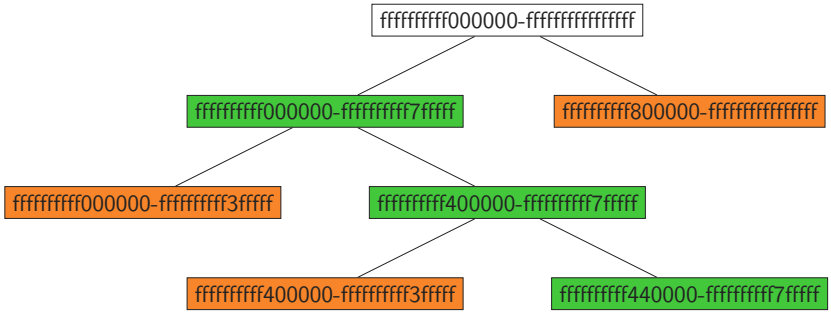


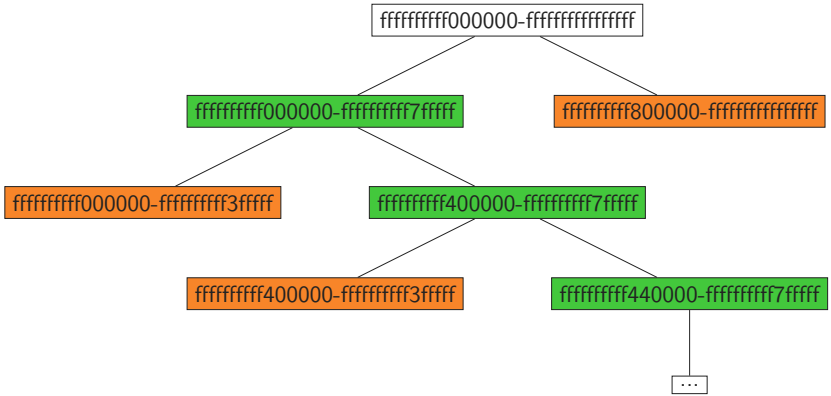

ffffffffff000000-ffffffffffffffff

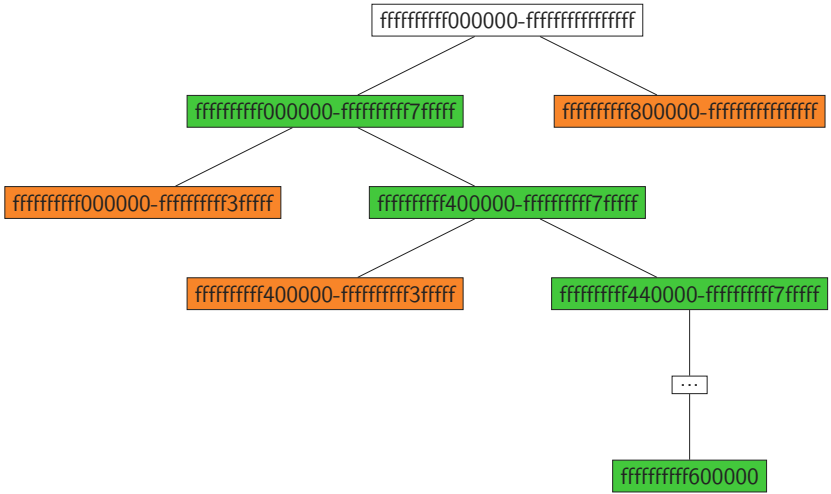

ffffffffff000000-ffffffffffffffff
ffffffffff000000-ffffffffff7fffff
ffffffffff800000-ffffffffffffffff


ffffffffff000000-ffffffffffffffff
ffffffffff000000-fffffffff7fffff
ffffffffff800000-ffffffffffffffff
ffffffffff000000-fffffffff3fffff
ffffffffff400000-fffffffff7fffff

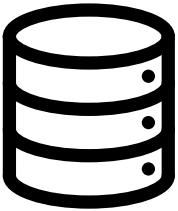

ffffffffff000000-ffffffffffffffff
ffffffffff000000-ffffffffff7fffff
ffffffffff800000-ffffffffffffffff
ffffffffff000000-ffffffffff3fffff
ffffffffff400000-ffffffffff7fffff
ffffffffff400000-ffffffffff3fffff


ffffffffff000000-ffffffffffffffff
ffffffffff000000-fffffffff7fffff
ffffffffff800000-ffffffffffffffff
ffffffffff000000-fffffffff3fffff
ffffffffff400000-fffffffff7fffff
ffffffffff400000-fffffffff3fffff
ffffffffff440000-fffffffff7fffff


ffffffffff000000-ffffffffffffffff
ffffffffff000000-fffffffff7fffff
ffffffffff800000-ffffffffffffffff
ffffffffff000000-fffffffff3fffff
ffffffffff400000-fffffffff7fffff
ffffffffff400000-fffffffff3fffff
ffffffffff440000-fffffffff7fffff
...

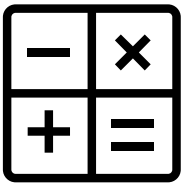

ffffffffff000000-ffffffffffffffff
ffffffffff000000-fffffffff7fffff
ffffffffff800000-ffffffffffffffff
ffffffffff000000-fffffffff3fffff
ffffffffff400000-fffffffff7fffff
ffffffffff400000-fffffffff3fffff
ffffffffff440000-fffffffff7fffff
...
ffffffffff600000

- All Spectre variants so far use the cache

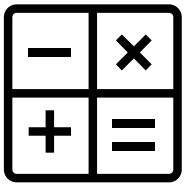


- All Spectre variants so far use the cache
- Is this a requirement?

- All Spectre variants so far use the cache
- Is this a requirement?
- Can we encode the data somewhere else?

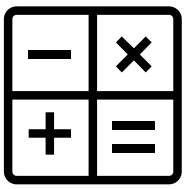


- Allow performing an operation in parallel on multiple data

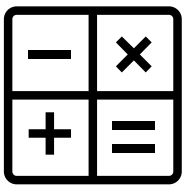


- Allow performing an operation in parallel on multiple data
- Commonly used in gaming and cryptography

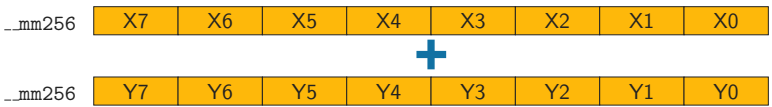
Advanced Vector Instructions (AVX)

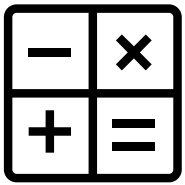


- Allow performing an operation in parallel on multiple data
- Commonly used in gaming and cryptography

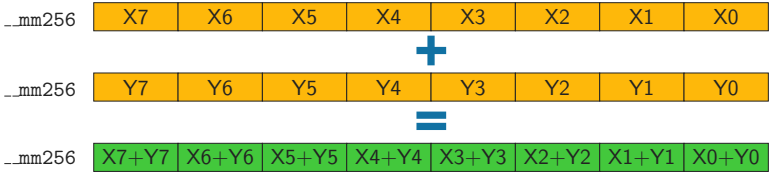


- Allow performing an operation in parallel on multiple data
- Commonly used in gaming and cryptography





- Allow performing an operation in parallel on multiple data
- Commonly used in gaming and cryptography



- 256-bit instructions need a lot of power

- 256-bit instructions need a lot of power
  - → On Intel, disabled by default, enabled on first use

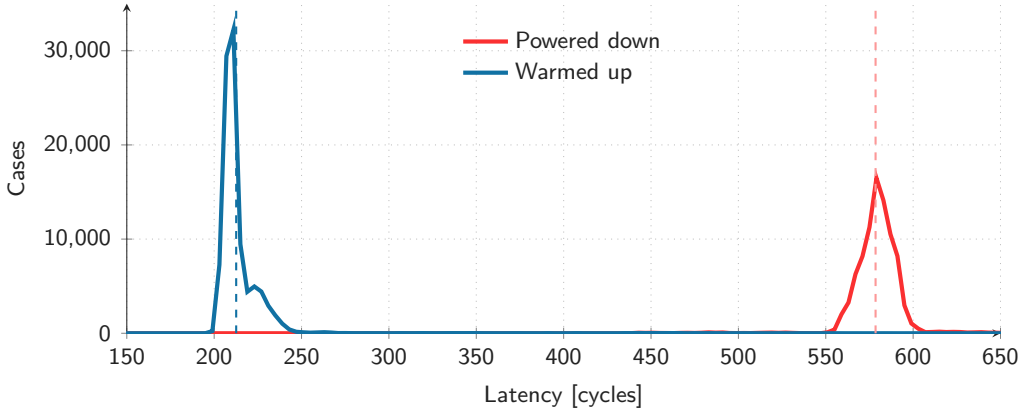
- 256-bit instructions need a lot of power
  - → On Intel, disabled by default, enabled on first use
- Requires some time to power up

- 256-bit instructions need a lot of power
  - → On Intel, disabled by default, enabled on first use
- Requires some time to power up
- Measure execution time of AVX instruction

- 256-bit instructions need a lot of power
  - → On Intel, disabled by default, enabled on first use
- Requires some time to power up
- Measure execution time of AVX instruction

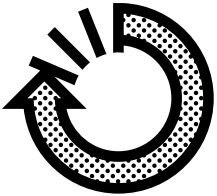
→ Leak timing information

AVX Latency

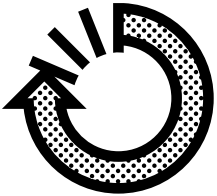

Powered down
Warmed up
Cases
30,000
20,000
10,000
0
150
200
250
300
350
400
450
500
550
600
650
Latency [cycles]

AVX Leak Gadget

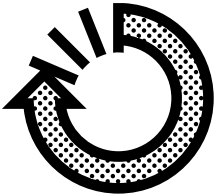
```
if (x < bitstream_length)
    if(bitstream[x])
        _mm256_instruction();
```



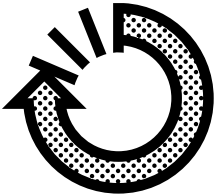
- We had to thrash cache to reset state



- We had to thrash cache to reset state
- Wait $\approx$1 ms $\rightarrow$ AVX unit powers off

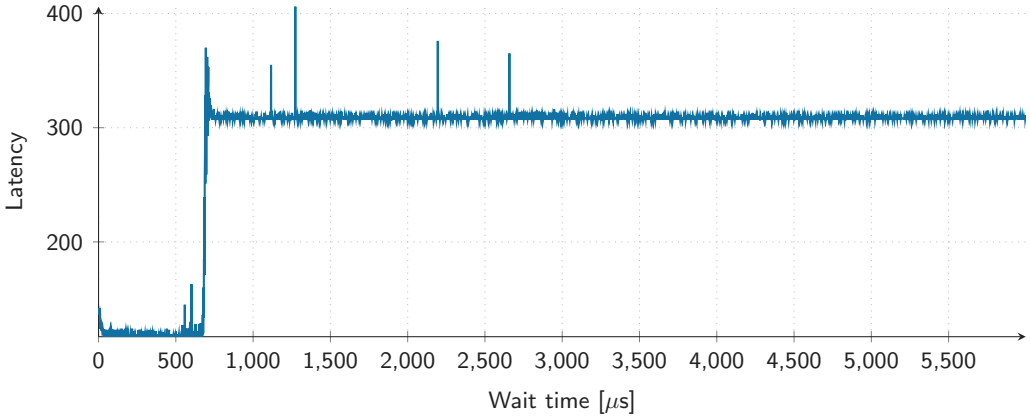


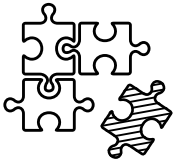
- We had to thrash cache to reset state
- Wait $\approx$1 ms $\rightarrow$ AVX unit powers off
- More efficient and stealthier than constantly downloading a file



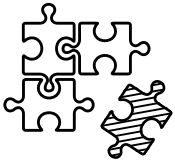
- We had to thrash cache to reset state
- Wait $\approx$1 ms $\rightarrow$ AVX unit powers off
- More efficient and stealthier than constantly downloading a file
- $\rightarrow$ higher performance than cache covert channel

AVX Cooldown

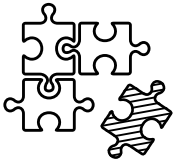




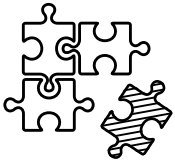
1. Mistrain branch predictor with in-bounds requests



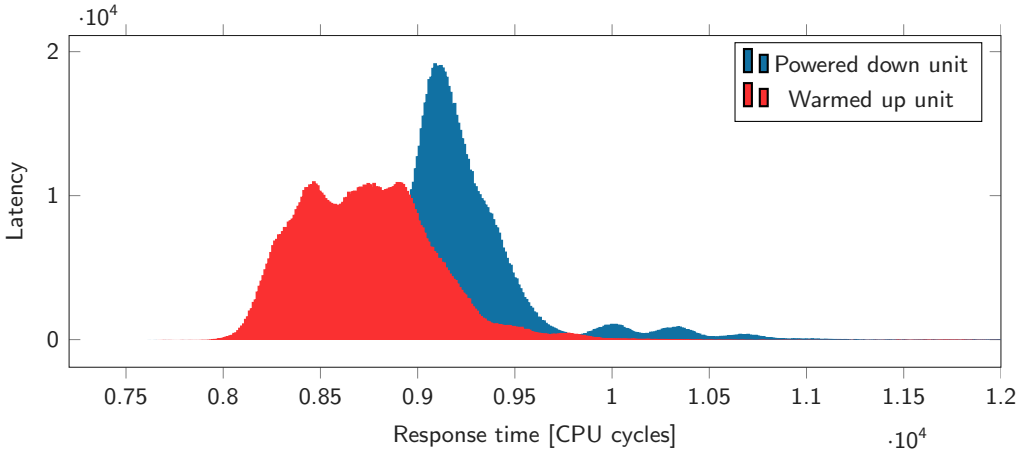
1. Mistrain branch predictor with in-bounds requests
2. Wait for AVX unit to power off (1ms)



1. Mistrain branch predictor with in-bounds requests
2. Wait for AVX unit to power off (1ms)
3. Leak a bit: do nothing (‘0’) or power AVX unit (‘1’)

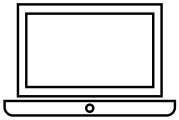


1. Mistrain branch predictor with in-bounds requests
2. Wait for AVX unit to power off (1ms)
3. Leak a bit: do nothing (‘0’) or power AVX unit (‘1’)
4. Measure function latency which uses AVX instruction


·10^4
2
1
0
Latency
Powered down unit
Warmed up unit
0.75
0.8
0.85
0.9
0.95
1
1.05
1.1
1.15
1.2
Response time [CPU cycles]
·10^4

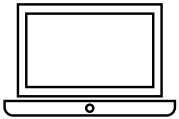
- NetSpectre tested in various environments

- NetSpectre tested in various environments



i5-6200U, i7-8550U

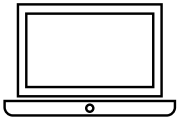
- NetSpectre tested in various environments



i5-6200U, i7-8550U

i7-6700K, i7-8700K

- NetSpectre tested in various environments

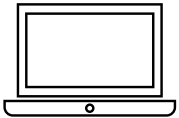


i5-6200U, i7-8550U

i7-6700K, i7-8700K

Skylake Xeon

- NetSpectre tested in various environments



i5-6200U, i7-8550U

i7-6700K, i7-8700K

Skylake Xeon

ARM Cortex A75

- Local Network (1 000 000 measurements/bit)

- Local Network (1 000 000 measurements/bit)

30 min/byte

- Local Network (1 000 000 measurements/bit)

30 min/byte

8 min/byte

- Local Network (1 000 000 measurements/bit)

30 min/byte

8 min/byte

- Cloud (20 000 000 measurements/bit)

- Local Network (1 000 000 measurements/bit)

30 min/byte

8 min/byte

- Cloud (20 000 000 measurements/bit)

1 h/bit

- Mitigating NetSpectre

- Mitigating NetSpectre

Network side

- Mitigating NetSpectre

Network side

Fix Spectre

- Prevent NetSpectre on the network side

- Prevent NetSpectre on the network side

Firewalls and DDoS protections

- Prevent NetSpectre on the network side

Firewalls and DDoS protections

Add random noise to packets

- Prevent NetSpectre on the network side

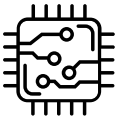
Firewalls and DDoS protections

Add random noise to packets

Network segmentation

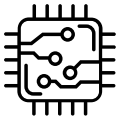
- Prevent (Net)Spectre on the system side

- Prevent (Net)Spectre on the system side



Hardware Fixes

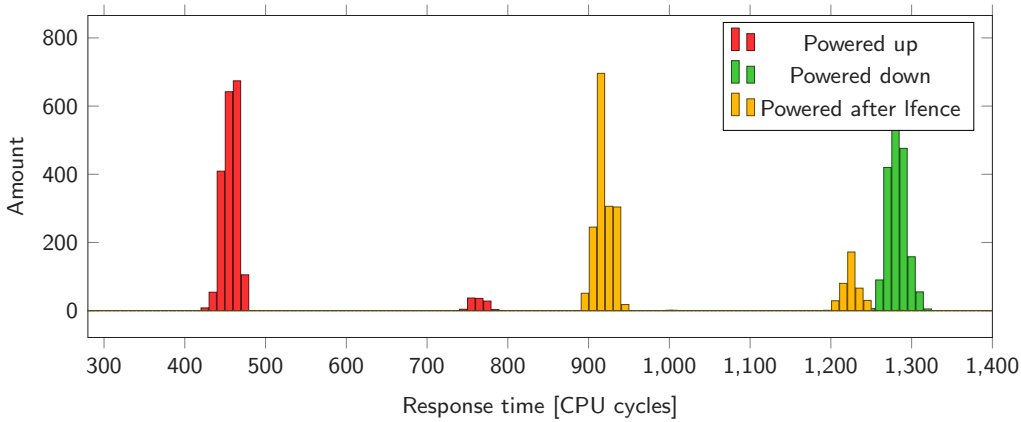
- Prevent (Net)Spectre on the system side



Hardware Fixes

Software Changes

AVX vs single lfence



- NetSpectre requires a fast and stable network connection

- NetSpectre requires a fast and stable network connection
  - Local networks

- NetSpectre requires a fast and stable network connection
  - Local networks
  - Data centers (VM to VM attack)

- NetSpectre requires a fast and stable network connection
  - Local networks
  - Data centers (VM to VM attack)
- Internet speeds improve (e.g., fiber, 5G)

- NetSpectre requires a fast and stable network connection
  - Local networks
  - Data centers (VM to VM attack)
- Internet speeds improve (e.g., fiber, 5G)

$\rightarrow$ possible in the near future?

- NetSpectre requires a fast and stable network connection
  - Local networks
  - Data centers (VM to VM attack)
- Internet speeds improve (e.g., fiber, 5G)

$\rightarrow$ possible in the near future?

- Attack speeds can be drastically improved

- NetSpectre requires a fast and stable network connection
  - Local networks
  - Data centers (VM to VM attack)
- Internet speeds improve (e.g., fiber, 5G)

$\rightarrow$ possible in the near future?

- Attack speeds can be drastically improved
  - Better signal processing/filtering

- NetSpectre requires a fast and stable network connection
  - Local networks
  - Data centers (VM to VM attack)
- Internet speeds improve (e.g., fiber, 5G)

$\rightarrow$ possible in the near future?

- Attack speeds can be drastically improved
  - Better signal processing/filtering
  - Dedicated measuring hardware

- Gadgets are more versatile than expected

- Gadgets are more versatile than expected
- Finding gadgets is even harder than expected

What do we learn from it?

- Gadgets are more versatile than expected
- Finding gadgets is even harder than expected
- Proposed security mechanisms are incomplete

What do we learn from it?

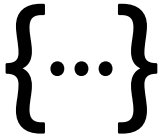
- Gadgets are more versatile than expected
- Finding gadgets is even harder than expected
- Proposed security mechanisms are incomplete
  - focus only on the cache

What do we learn from it?

- Gadgets are more versatile than expected
- Finding gadgets is even harder than expected
- Proposed security mechanisms are incomplete
  - focus only on the cache
  - often assume (local) code execution

What do we learn from it?

- Gadgets are more versatile than expected
- Finding gadgets is even harder than expected
- Proposed security mechanisms are incomplete
  - focus only on the cache
  - often assume (local) code execution
- Root problem has to be solved → more research required



- Speculative execution leaks secrets without exploiting bugs
- Spectre attacks are not limited to local attackers
- Spectre attacks have a larger impact than assumed

black hat ASIA 2019

NetSpectre

A Truly Remote Spectre Variant

Michael Schwarz @misc0110

Martin Schwarzl @marv0x90