black hat ASIA 2019

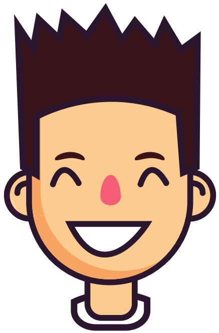
NetSpectre

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

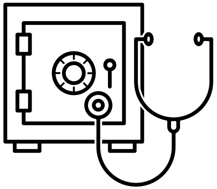
Master student @ Graz University of Technology

@marv0x90

m.schwarzl@student.tugraz.at

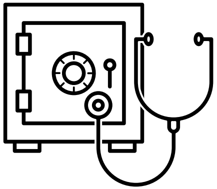
Side-Channel Attacks

- Bug-free software does not mean safe execution

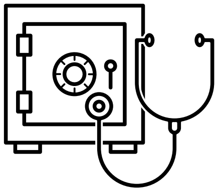


Side-Channel Attacks

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



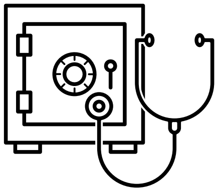
Side-Channel Attacks



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

Side-Channel Attacks

- 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

●●●

Architecture and Microarchitecture

- Instruction Set Architecture (ISA) is an abstract model of a computer (x86, ARMv8, SPARC, ...)

Architecture and Microarchitecture

- Instruction Set Architecture (ISA) is an abstract model of a computer (x86, ARMv8, SPARC, ...)
- Interface between hardware and software

Architecture and Microarchitecture

- Instruction Set Architecture (ISA) is an abstract model of a computer (x86, ARMv8, SPARC, ...)
- Interface between hardware and software
- Microarchitecture is an ISA implementation

Architecture and Microarchitecture

- Instruction Set Architecture (ISA) is an abstract model of a computer (x86, ARMv8, SPARC, ...)
- Interface between hardware and software
- Microarchitecture is an ISA implementation

Microarchitectural Components

- Modern CPUs contain multiple microarchitectural elements

Microarchitectural Components

- Modern CPUs contain multiple microarchitectural elements

Caches and buffers

Predictors

●●●

Microarchitectural Components

- Modern CPUs contain multiple microarchitectural elements

Caches and buffers

Predictors

- Transparent for the programmer

Microarchitectural Components

- Modern CPUs contain multiple microarchitectural elements

Caches and buffers

Predictors

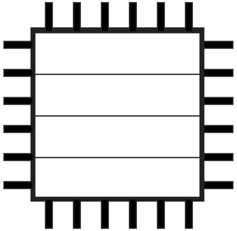
●●●

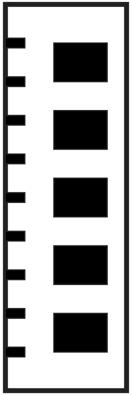
- Transparent for the programmer
- Timing optimizations → side-channel leakage

Let’s have a deeper look at the cache

CPU Cache

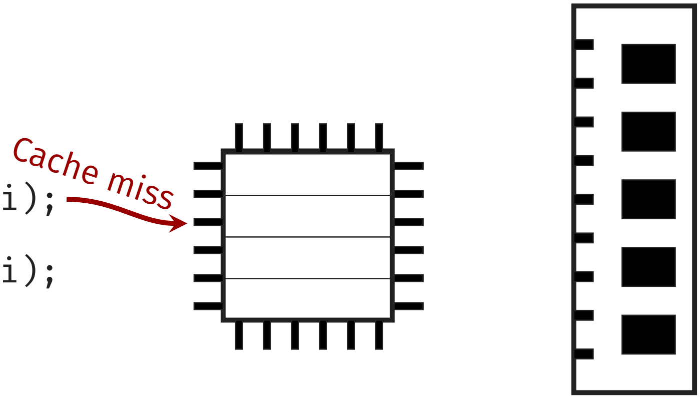
```
printf("%d", i);
printf("%d", i);
```





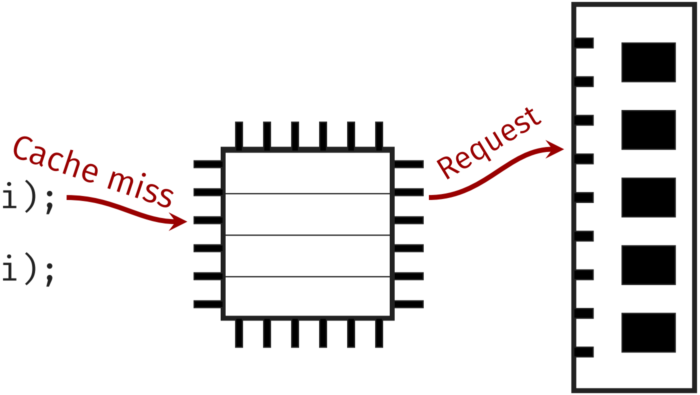
CPU Cache

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



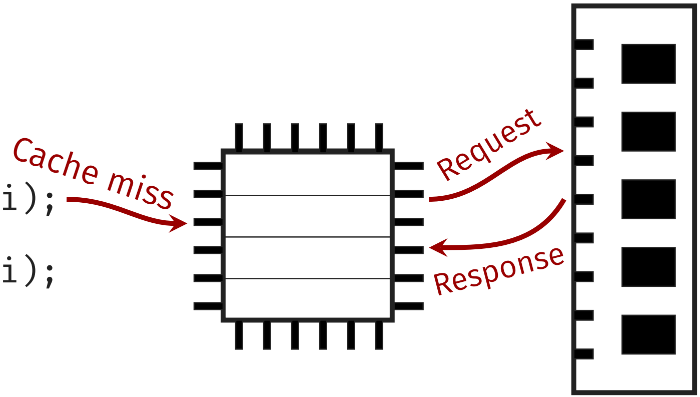
CPU Cache

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

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



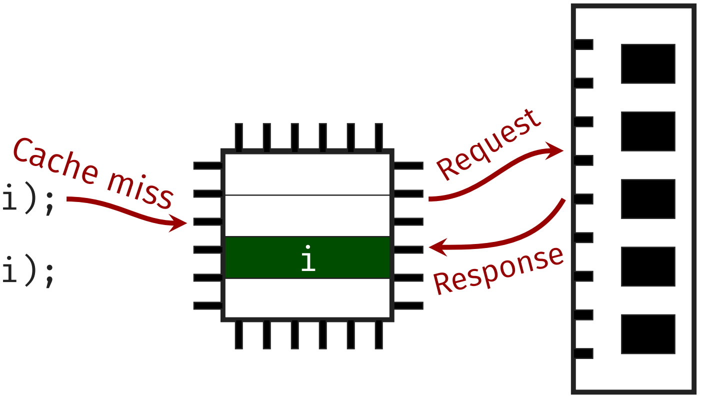
CPU Cache

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

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

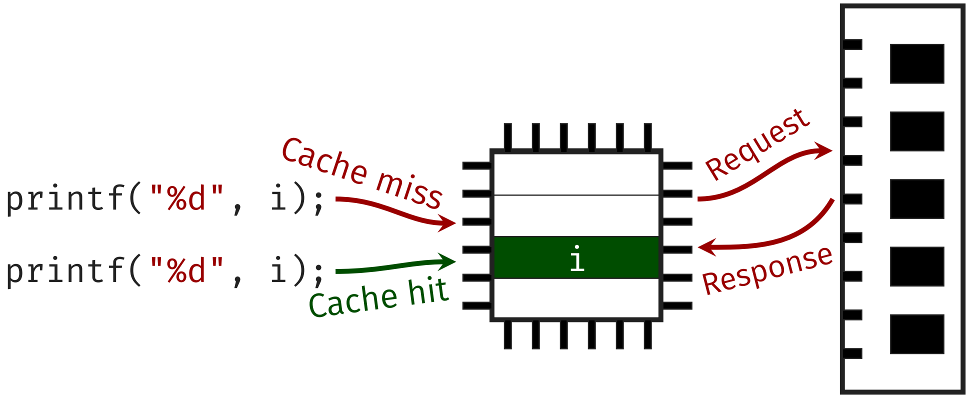


CPU Cache

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

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



CPU Cache

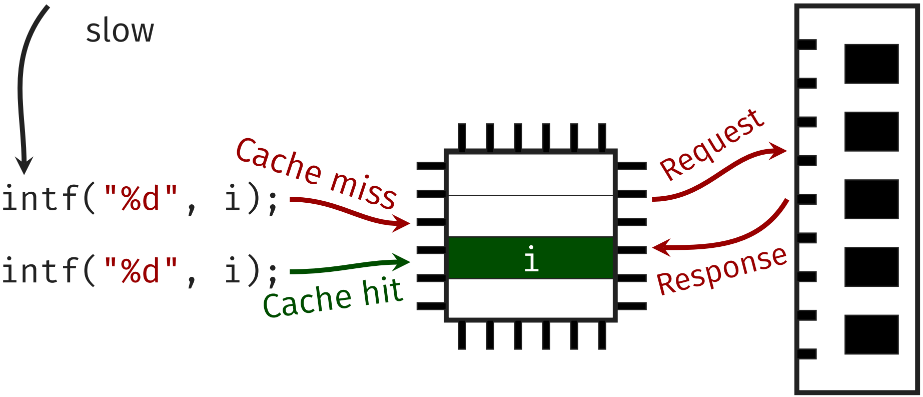


```
printf("%d", i);
printf("%d", i);
```

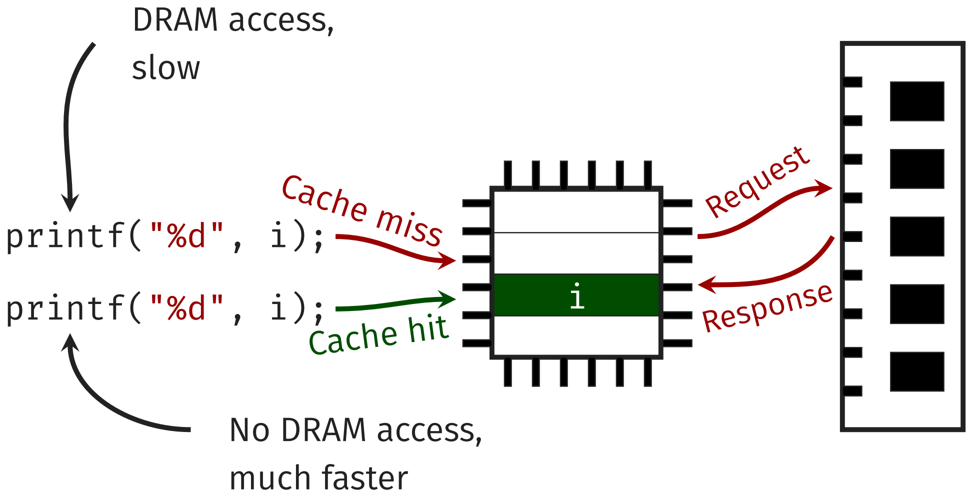
CPU Cache

DRAM access,
slow

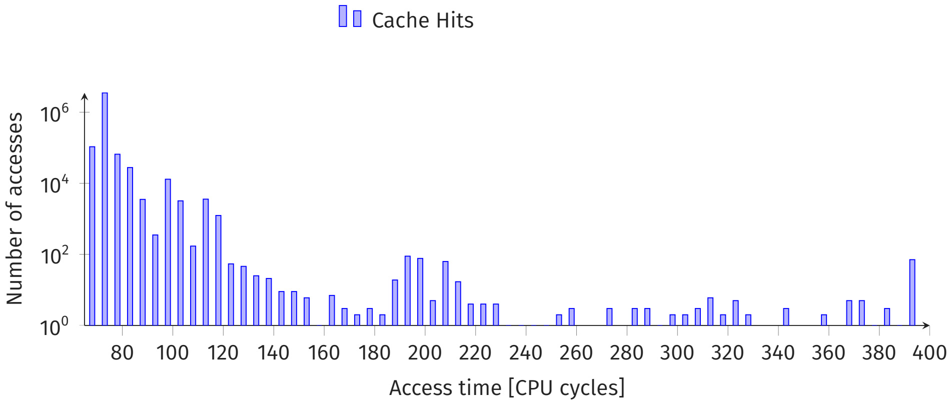
```
printf("%d", i);
printf("%d", i);
```



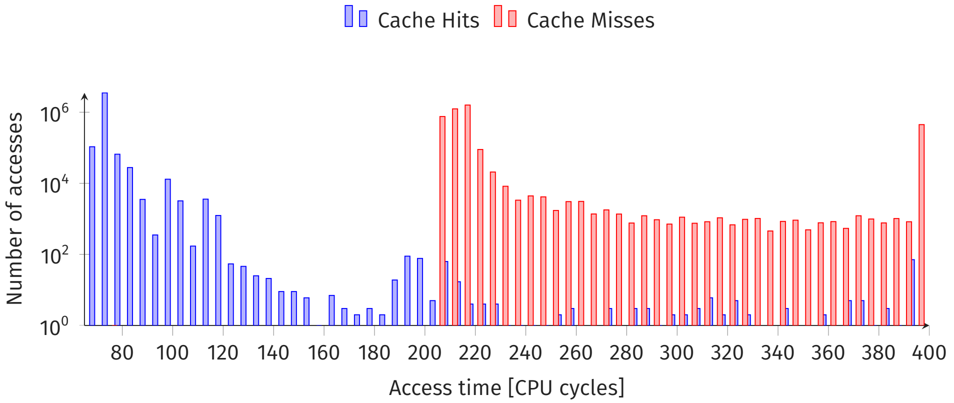
CPU Cache


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

Caching speeds up Memory Accesses



Caching speeds up Memory Accesses



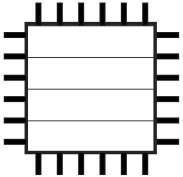
Evict+Reload

Shared Memory

Attacker

evict

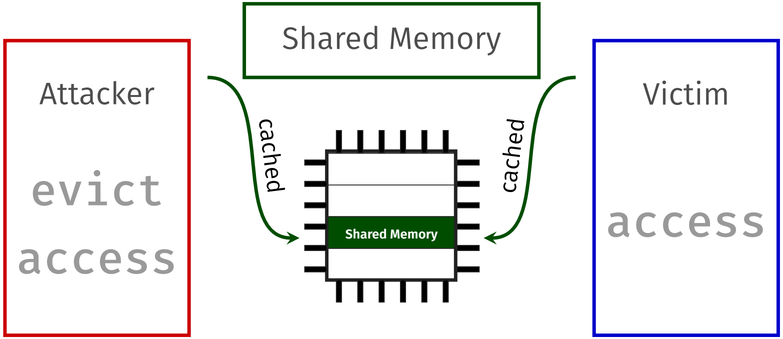
access



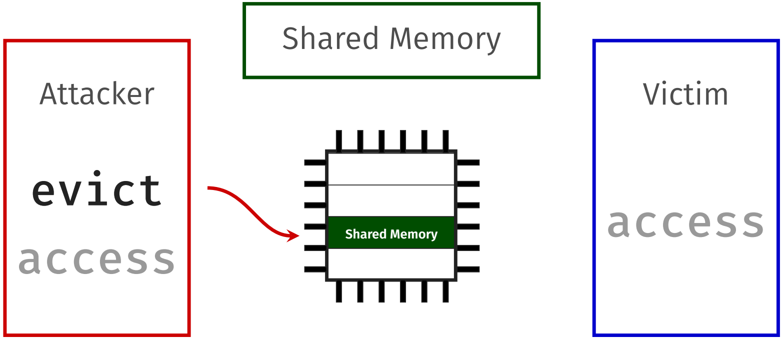
Victim

access

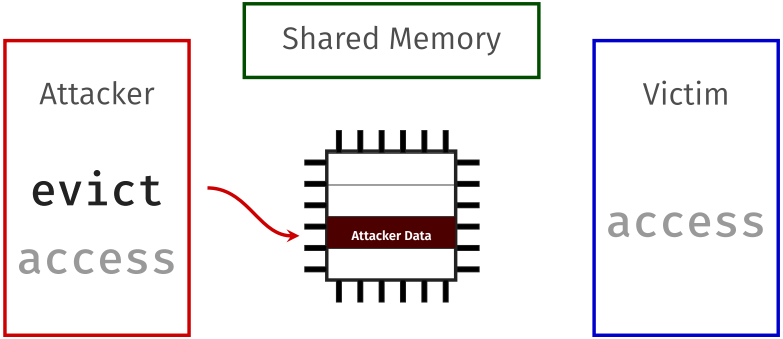
Evict+Reload



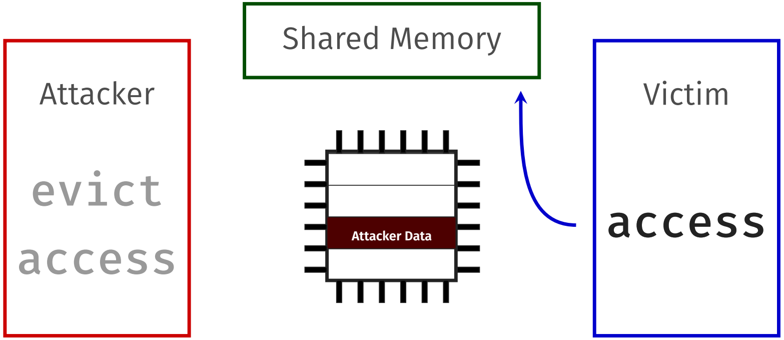
Evict+Reload



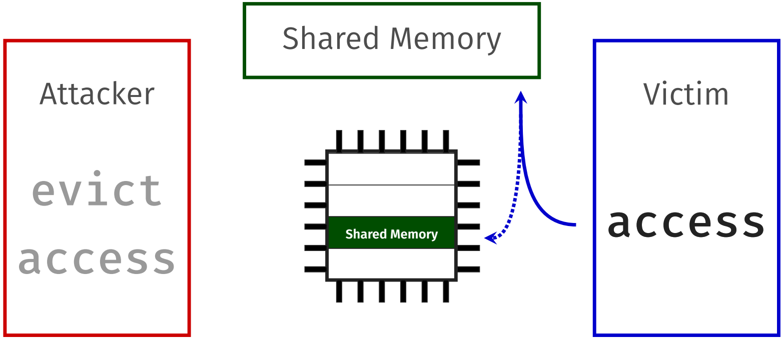
Evict+Reload



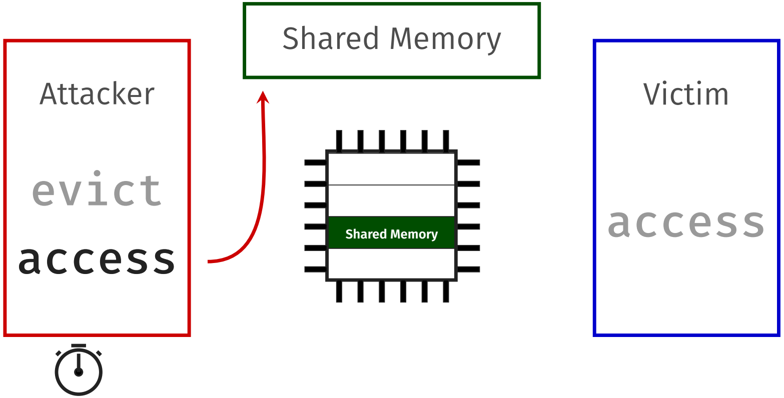
Evict+Reload



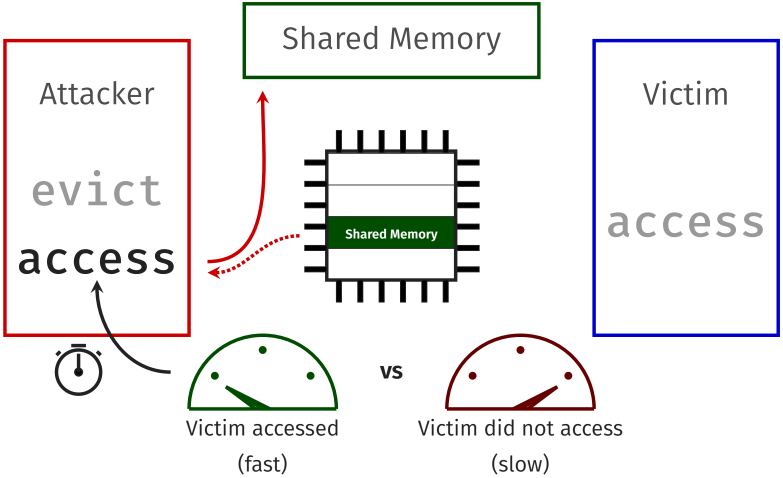
Evict+Reload



Evict+Reload



Evict+Reload



Speculative execution

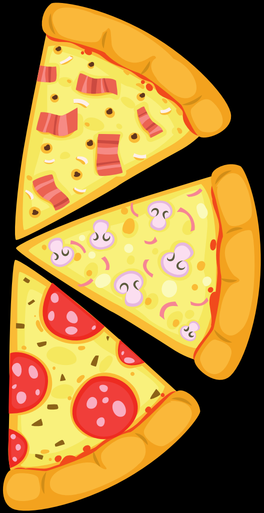
Speculative Execution

- 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
  - otherwise: Discard results
- Measurable side-effects

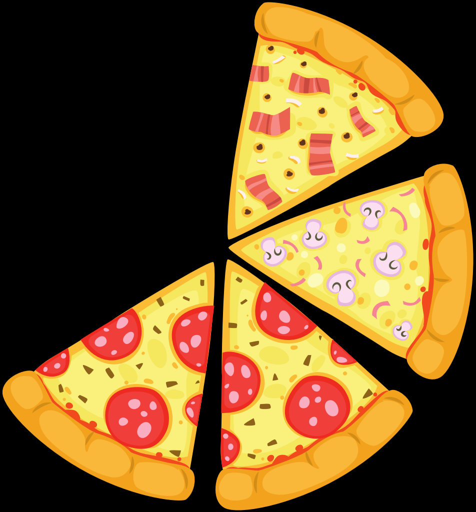
PIZZA
SPECIAL RECIPES

Prosciutto

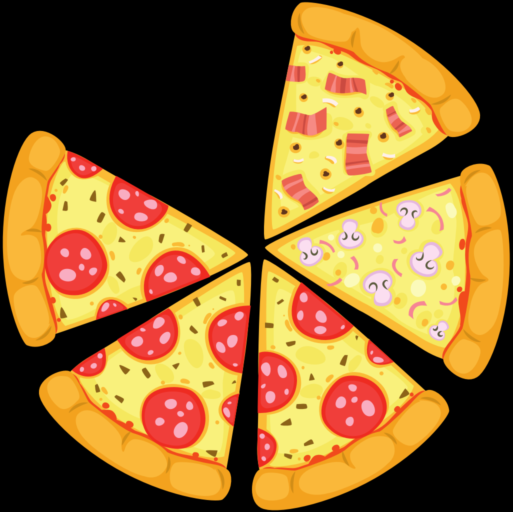
Funghi



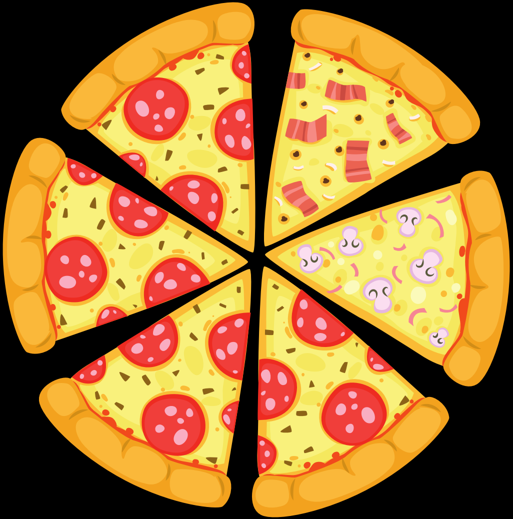
Diavolo



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

Spectre Requirements

- On Intel and AMD CPUs

Spectre Requirements

- On Intel and AMD CPUs
- Some ARMs (Cortex R and Cortex A) are also affected

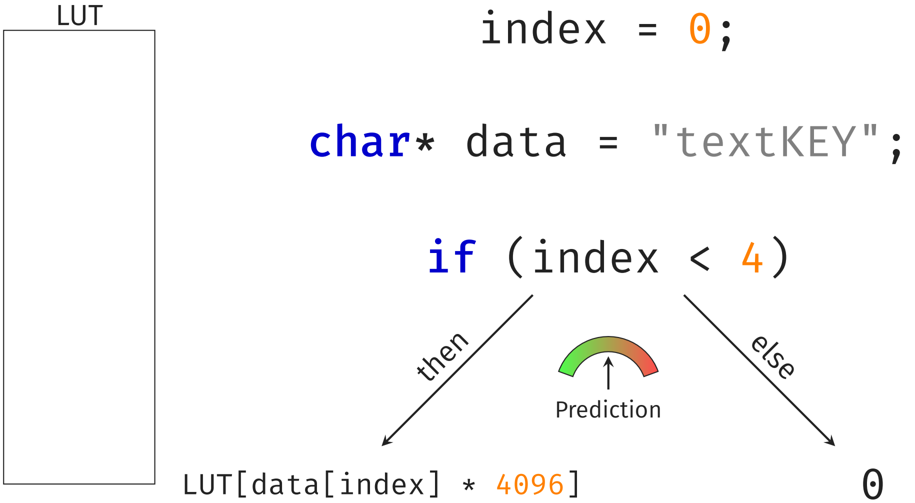
Spectre Requirements

- On Intel and AMD CPUs
- Some ARMs (Cortex R and Cortex A) are also affected
- Common cause: speculative execution of branches

Spectre Requirements

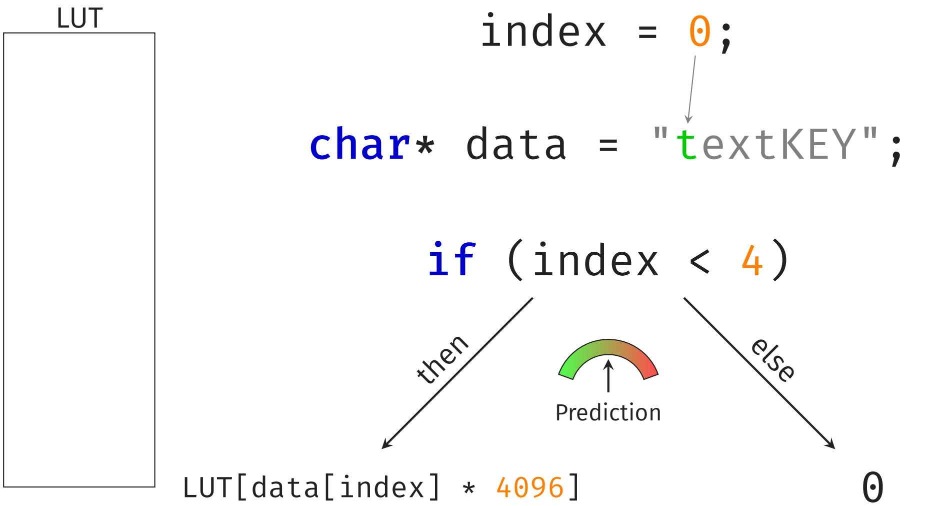
- On Intel and AMD CPUs
- Some ARMs (Cortex R and Cortex A) are also affected
- Common cause: speculative execution of branches
- Speculative execution leaves microarchitectural traces which leak secret

Spectre-PHT (aka Spectre Variant 1)

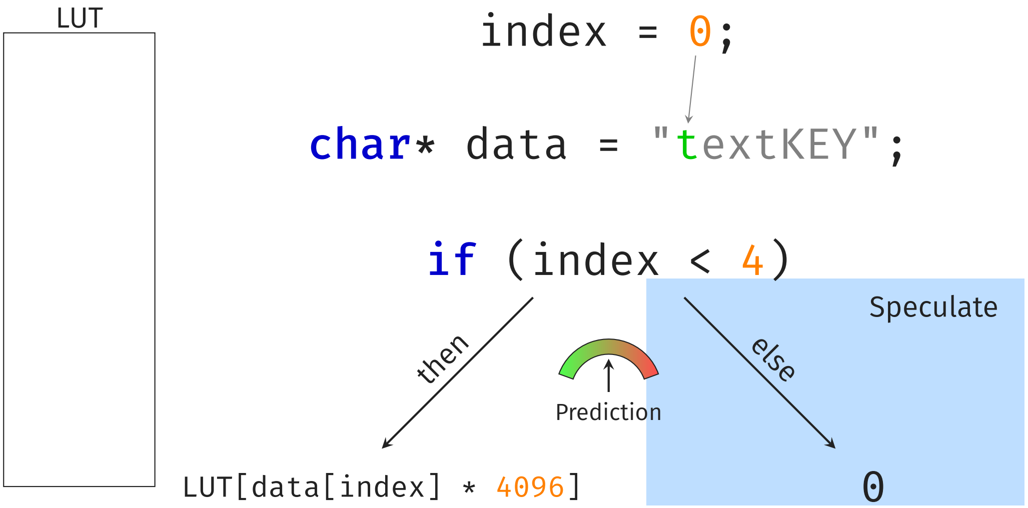


```
index = 0;

char* data = "textKEY";

if (index < 4)

LUT[data[index] * 4096]                    0
```

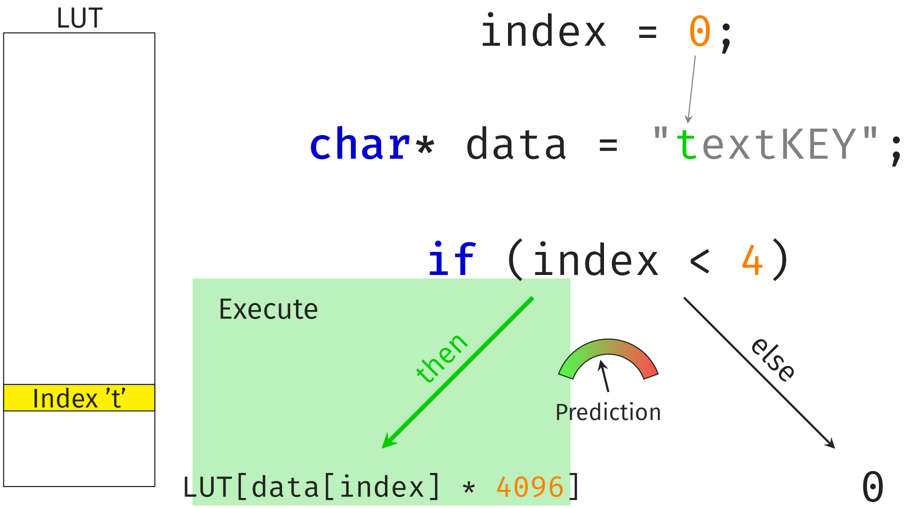
Spectre-PHT (aka Spectre Variant 1)



Spectre-PHT (aka Spectre Variant 1)

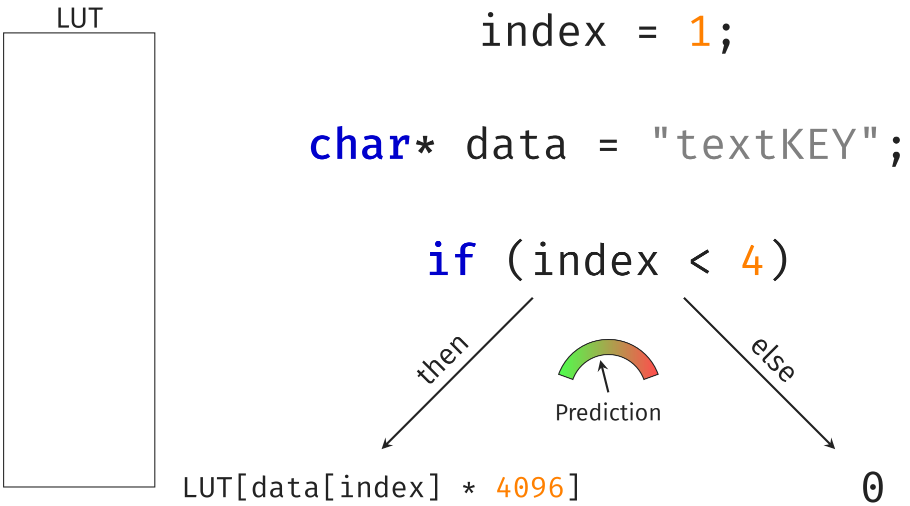


Spectre-PHT (aka Spectre Variant 1)



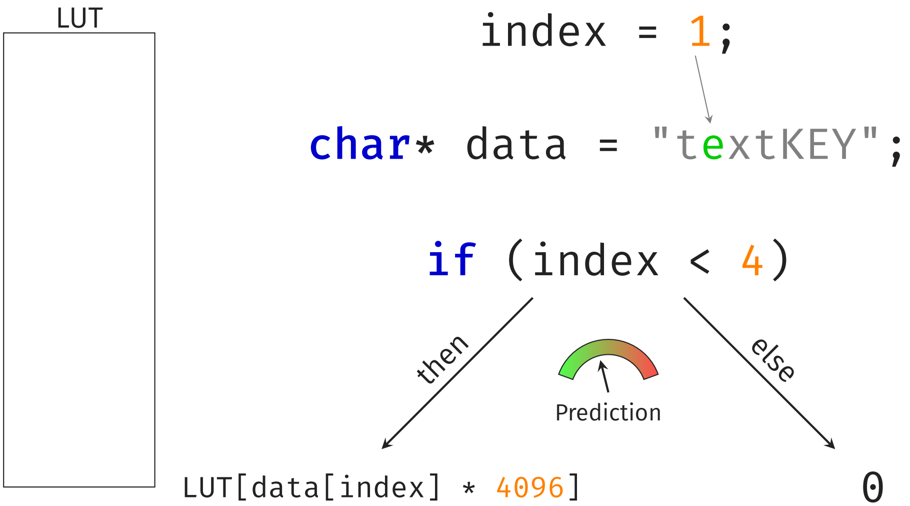
```
index = 0;

char* data = "textKEY";

if (index < 4)

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

Spectre-PHT (aka Spectre Variant 1)

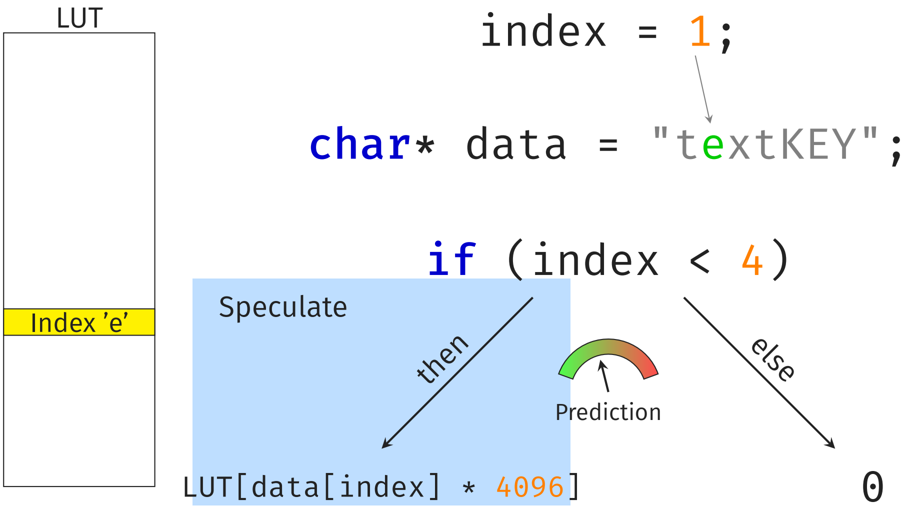


```
index = 1;

char* data = "textKEY";

if (index < 4)

LUT[data[index] * 4096]          0
```

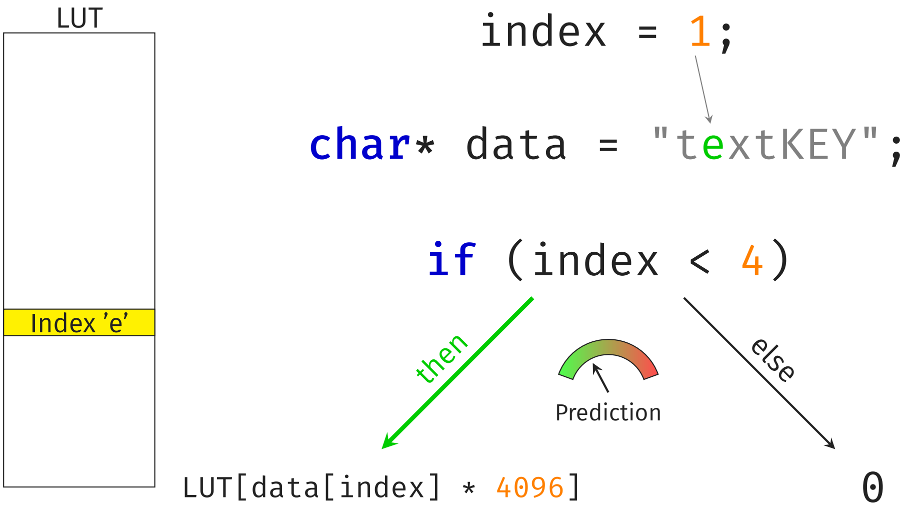
Spectre-PHT (aka Spectre Variant 1)



Spectre-PHT (aka Spectre Variant 1)

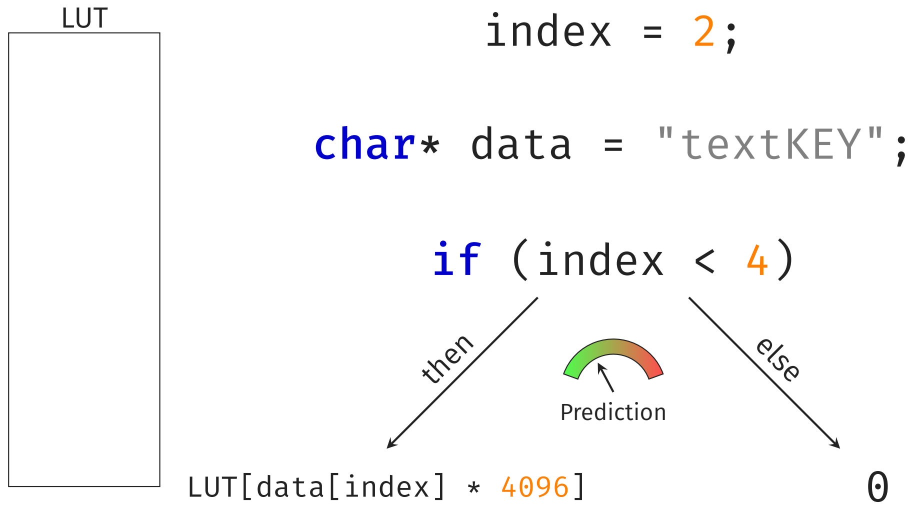


Spectre-PHT (aka Spectre Variant 1)



```
index = 1;

char* data = "textKEY";

if (index < 4)

LUT[data[index] * 4096]            0
```

Spectre-PHT (aka Spectre Variant 1)

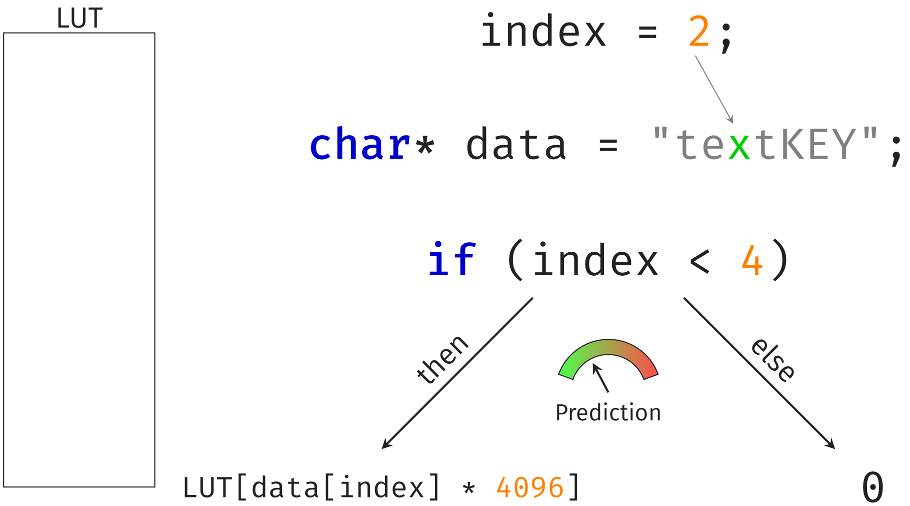


```
index = 2;

char* data = "textKEY";

if (index < 4)
```

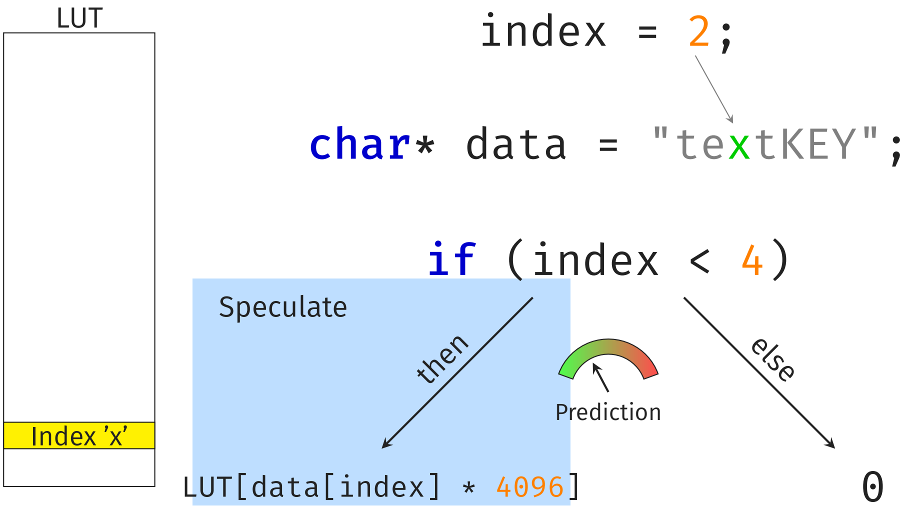
```
LUT[data[index] * 4096]
```

```
0
```

Spectre-PHT (aka Spectre Variant 1)

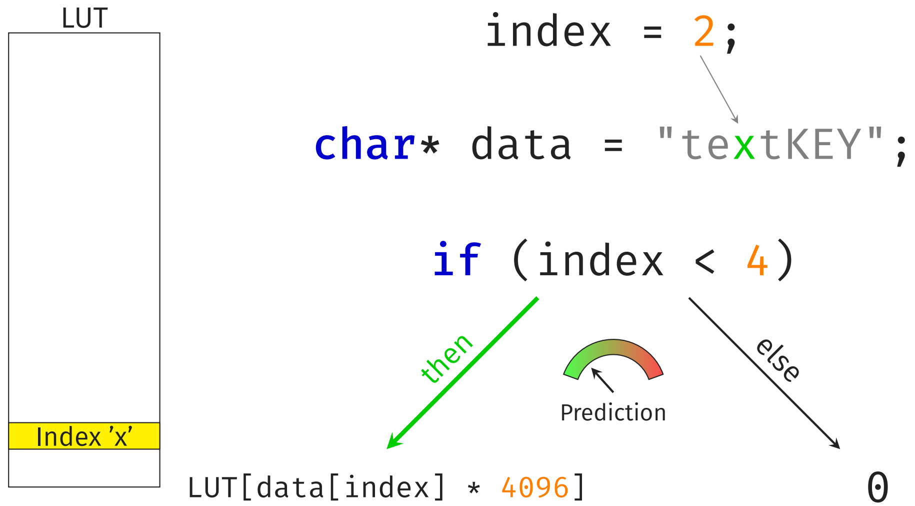


Spectre-PHT (aka Spectre Variant 1)

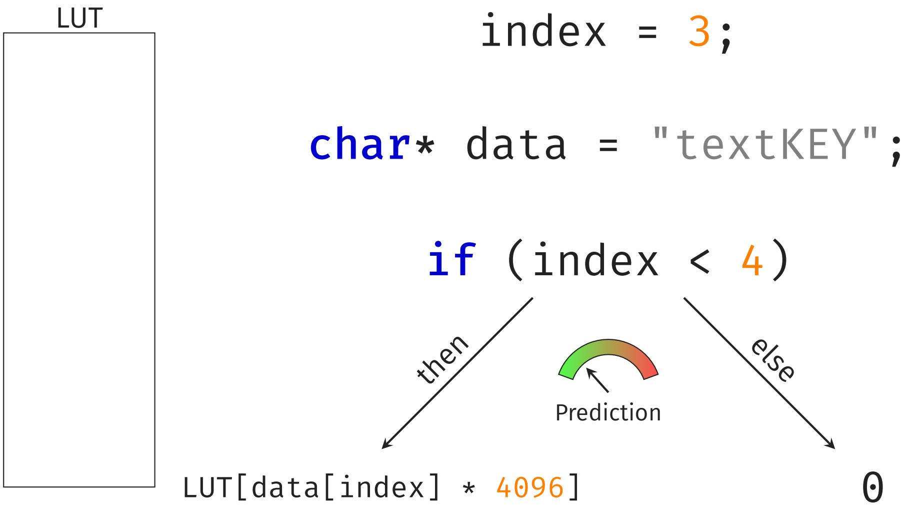


```
index = 2;

char* data = "textKEY";

if (index < 4)

LUT[data[index] * 4096]        0
```

Spectre-PHT (aka Spectre Variant 1)

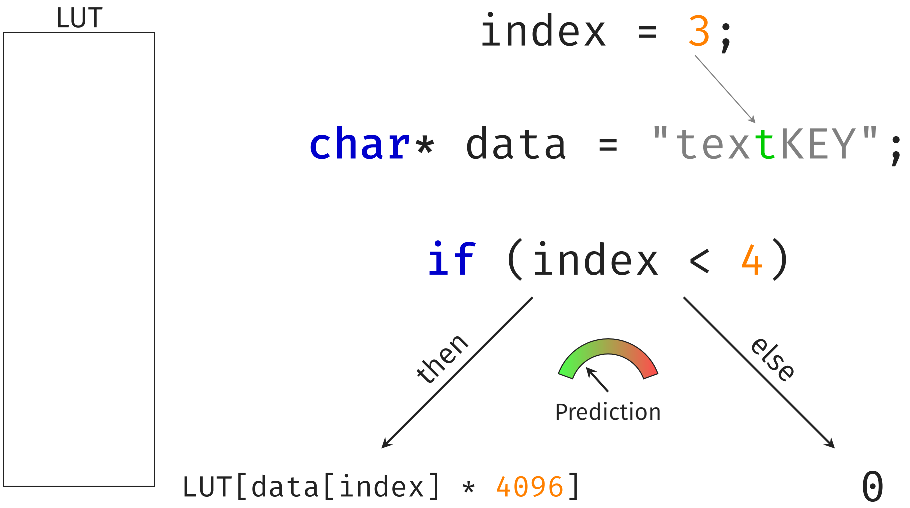


Spectre-PHT (aka Spectre Variant 1)

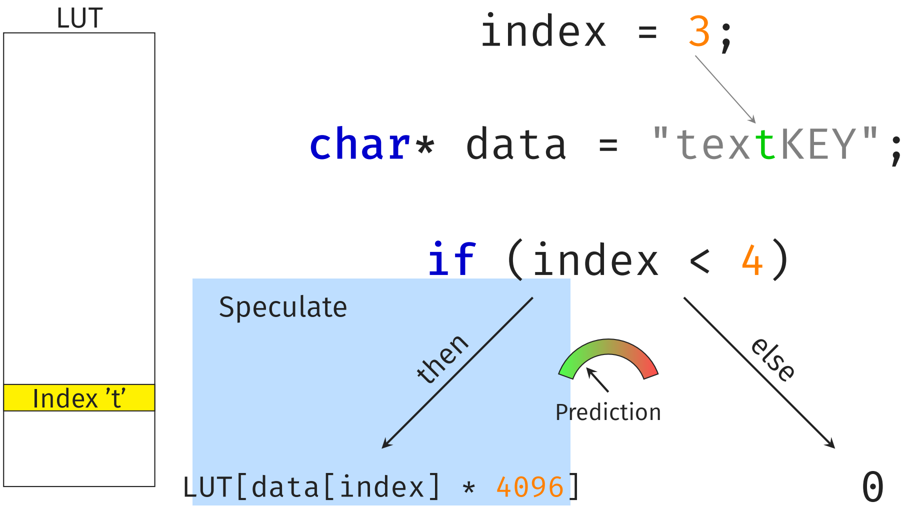


```
index = 3;

char* data = "textKEY";

if (index < 4)

LUT[data[index] * 4096]                    0
```

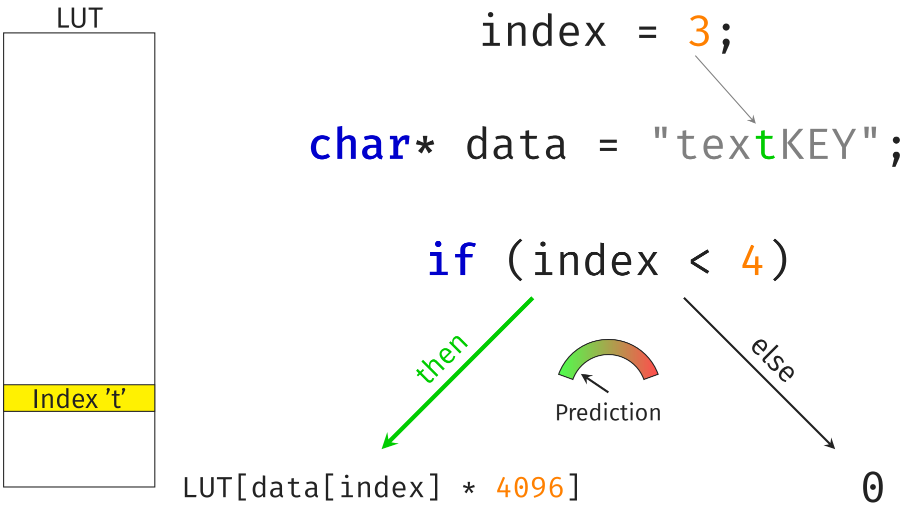
Spectre-PHT (aka Spectre Variant 1)



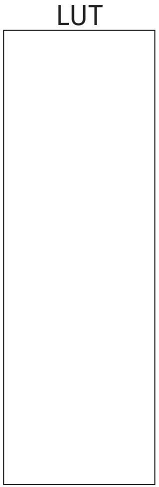
Spectre-PHT (aka Spectre Variant 1)



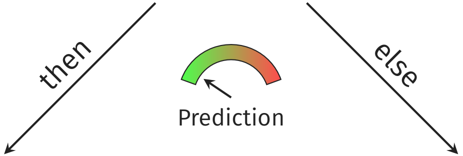
Spectre-PHT (aka Spectre Variant 1)



Spectre-PHT (aka Spectre Variant 1)

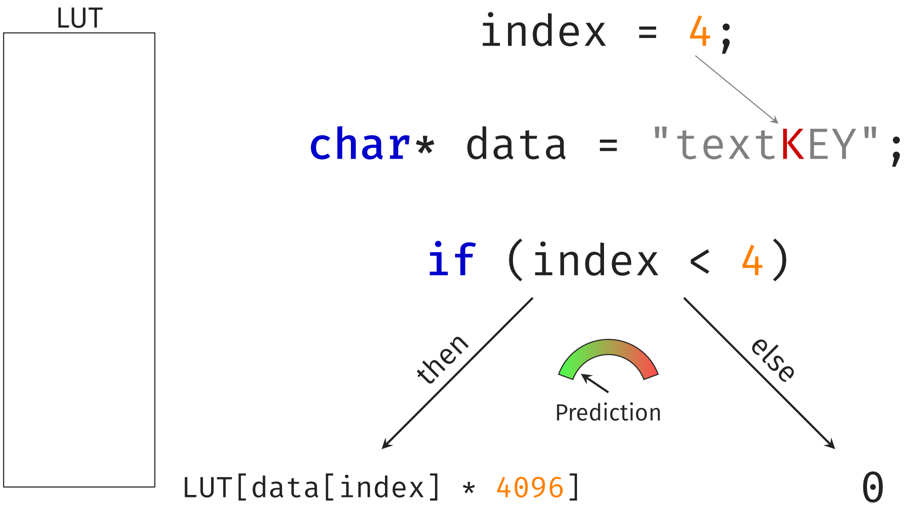


```
index = 4;

char* data = "textKEY";

if (index < 4)
```

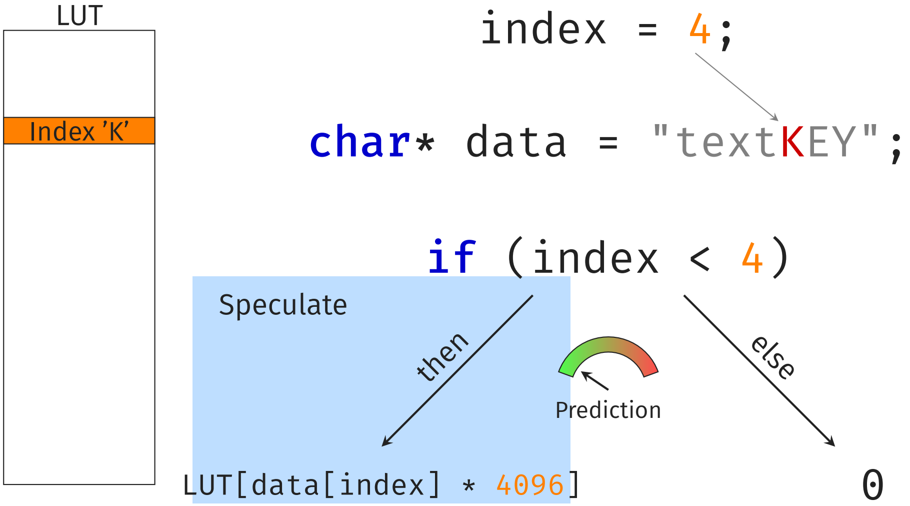


```
LUT[data[index] * 4096]                 0
```

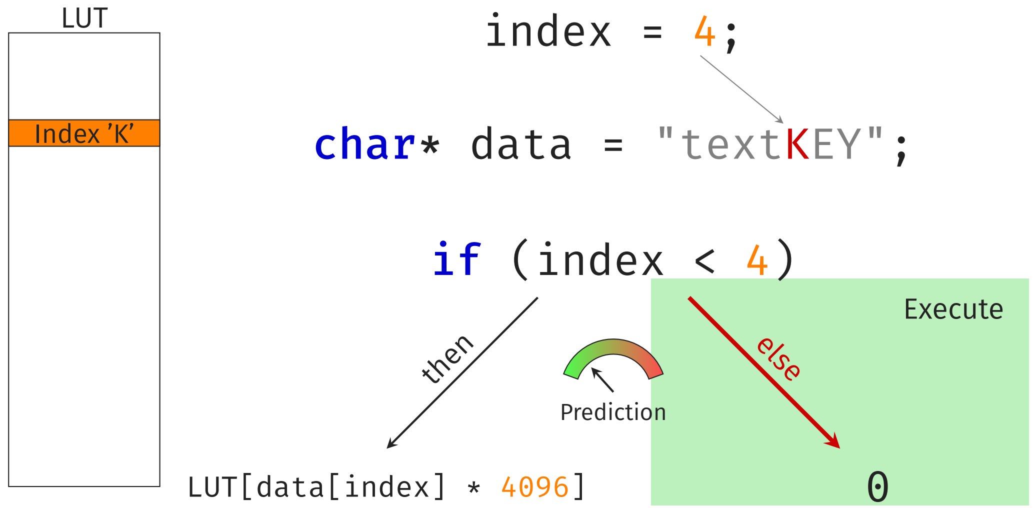
Spectre-PHT (aka Spectre Variant 1)



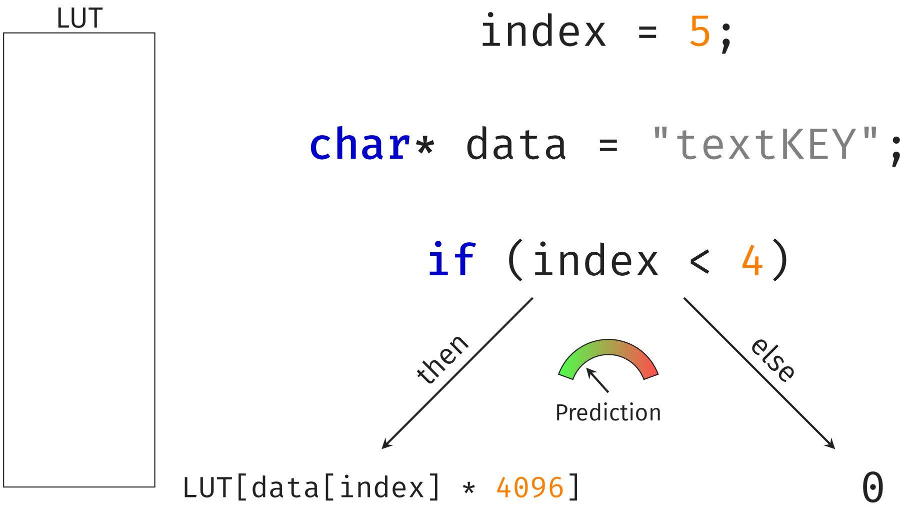
Spectre-PHT (aka Spectre Variant 1)



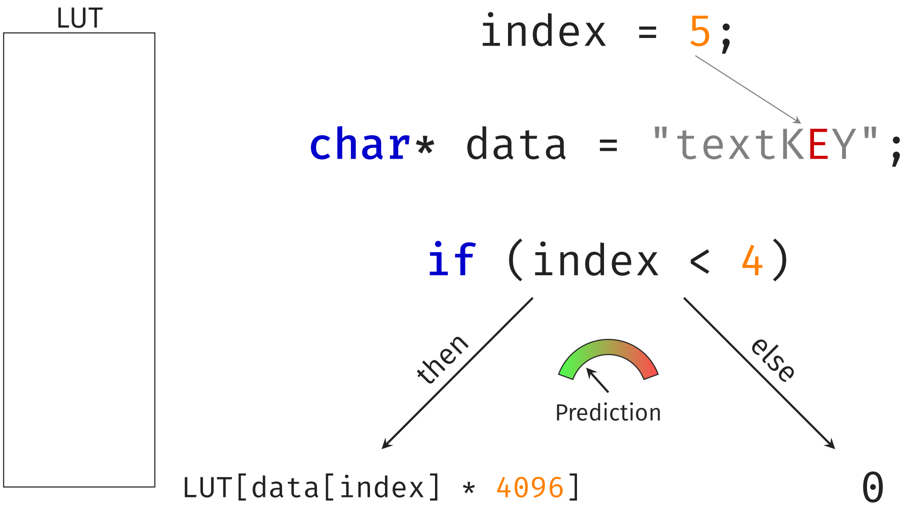
Spectre-PHT (aka Spectre Variant 1)



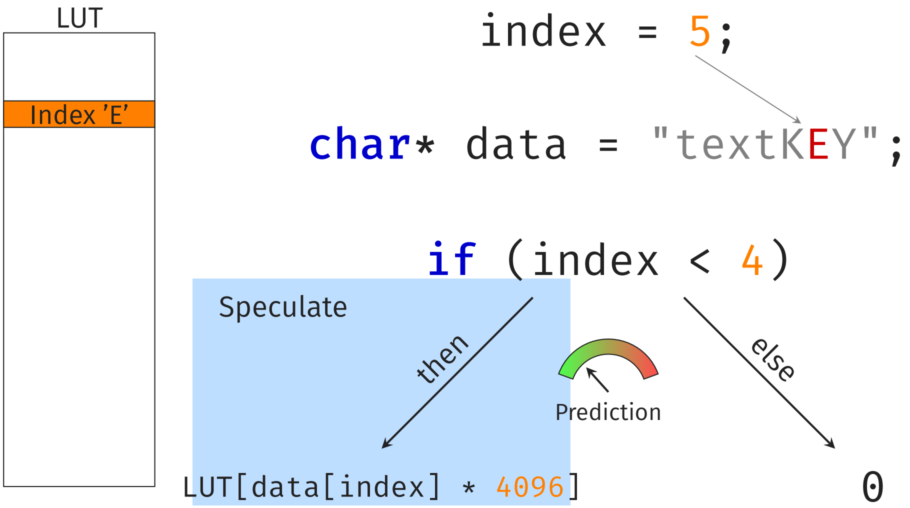
Spectre-PHT (aka Spectre Variant 1)



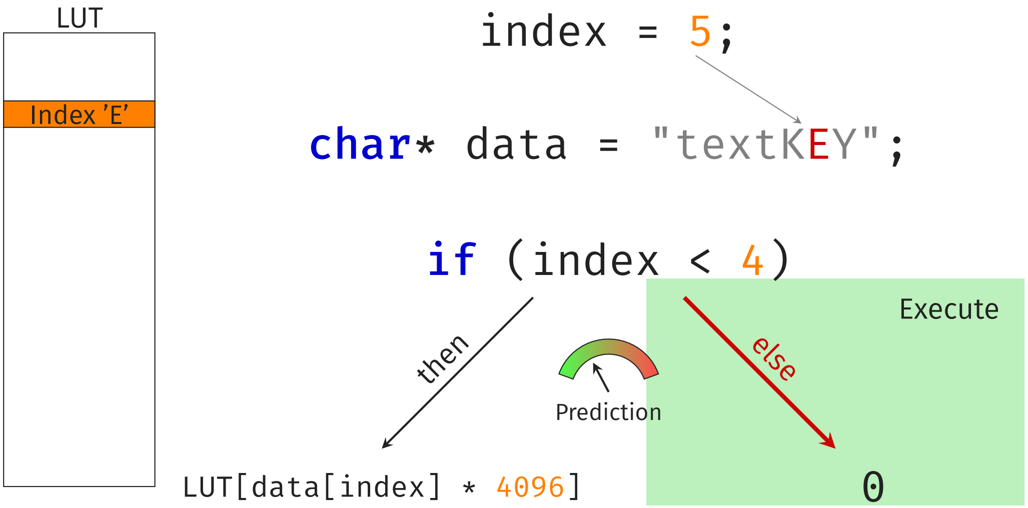
Spectre-PHT (aka Spectre Variant 1)



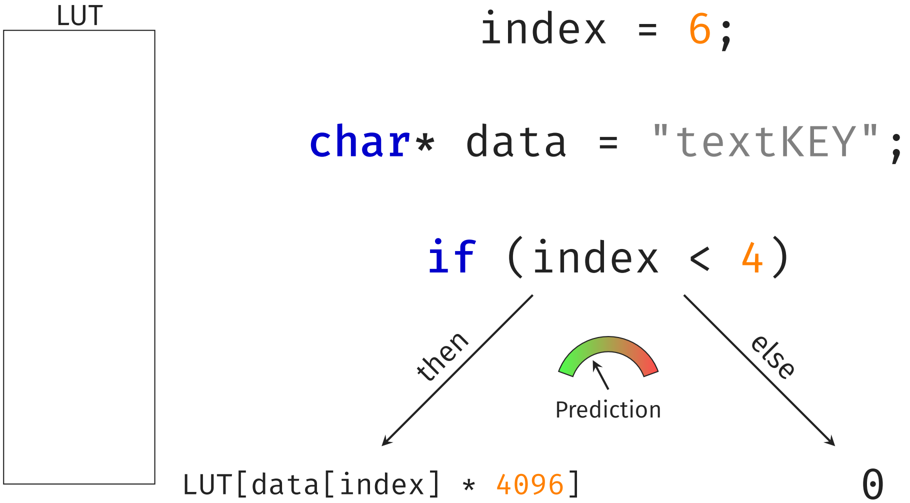
Spectre-PHT (aka Spectre Variant 1)



Spectre-PHT (aka Spectre Variant 1)

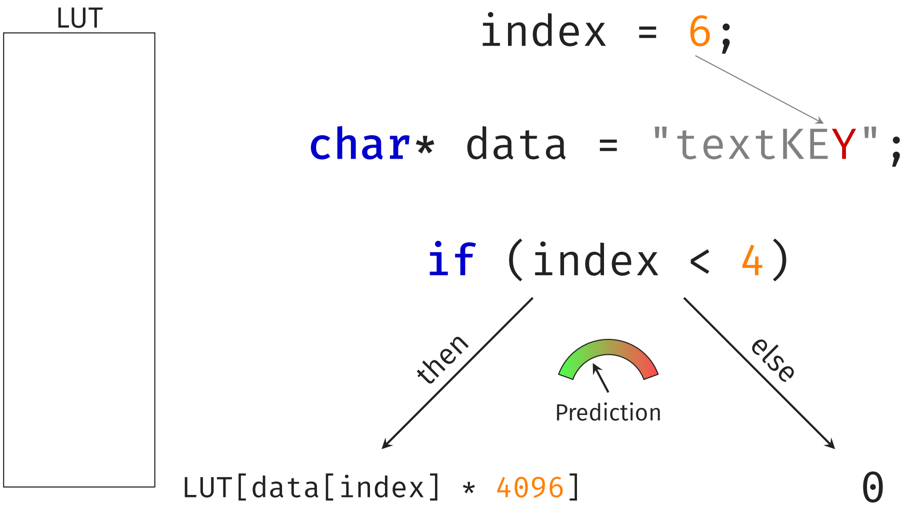


Spectre-PHT (aka Spectre Variant 1)

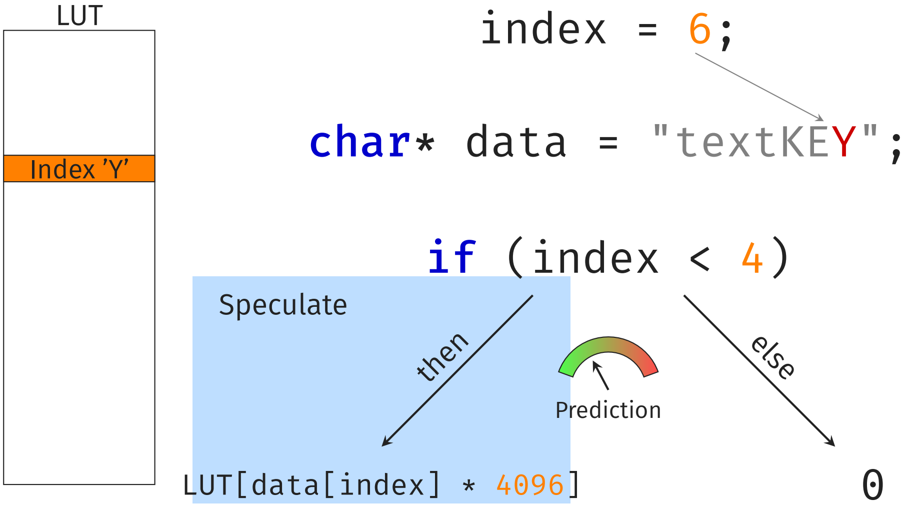


```
index = 6;

char* data = "textKEY";

if (index < 4)

LUT[data[index] * 4096]                 0
```

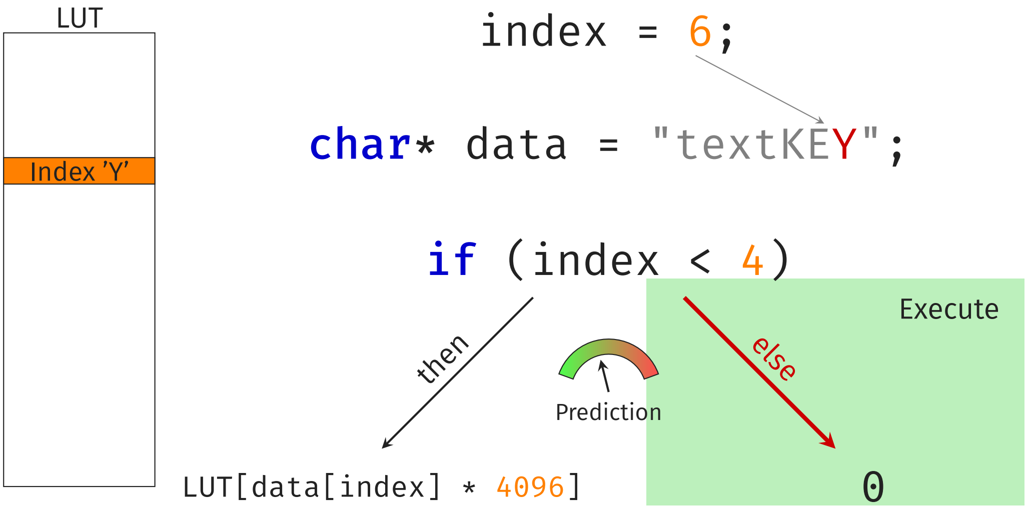
Spectre-PHT (aka Spectre Variant 1)



Spectre-PHT (aka Spectre Variant 1)



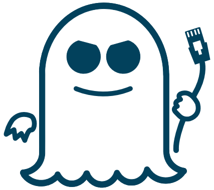
Spectre-PHT (aka Spectre Variant 1)



```
index = 6;

char* data = "textKEY";

if (index < 4)

LUT[data[index] * 4096]

0
```

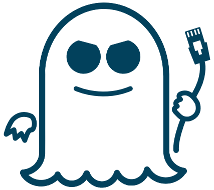
NetSpectre: A Remote Spectre Variant

The goal



We want to build a Spectre attack which...

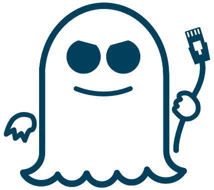
The goal



We want to build a Spectre attack which...

- is capable of leaking secrets from a remote system

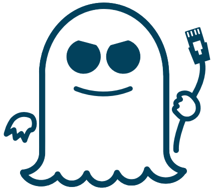
The goal



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

Wait a minute...

CVSS v3 for CVE-2017-5753 (Spectre)

Attack Vector

Network	Adjacent Network	[✓ Local]	Physical

Wait a minute...

CVSS v3 for CVE-2017-5753 (Spectre)

Attack Vector

Network	Adjacent Network	[✓ Local]	Physical

Attack Complexity

Low	[✓ High]

Wait a minute...

CVSS v3 for CVE-2017-5753 (Spectre)

Attack Vector

Network	Adjacent Network	[✓ Local]	Physical

Attack Complexity

Low	[✓ High]

Privilege Required

None	[✓ Low]	High

Wait a minute...

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

Problems

Spectre without code execution is complicated

Problems

Spectre without code execution is complicated

- Which branch can be exploited

Problems

Spectre without code execution is complicated

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

Problems

Spectre without code execution is complicated

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

Problems

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

Problems

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

Exploiting Branches

- No code can be injected

Exploiting Branches

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

Exploiting Branches

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

Exploiting Branches

- 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

API Example

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

  return SUCCESS
```

API Example

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

  return SUCCESS
```

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
```

API Example

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

- If bit in array was set → `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

Timing Measurement

- 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

$\rightarrow$ Measure the network roundtrip time

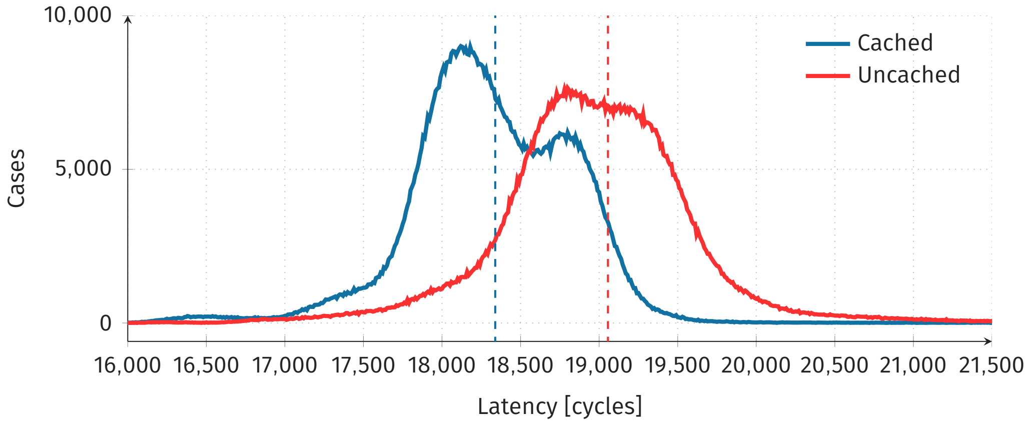
Timing Measurement

- 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

Network Measurement



Resetting Cache State

- After measuring variable is always cached

Resetting Cache State

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

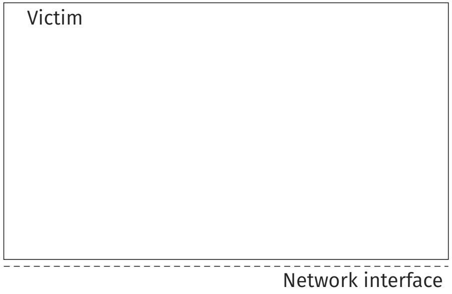
Resetting Cache State

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

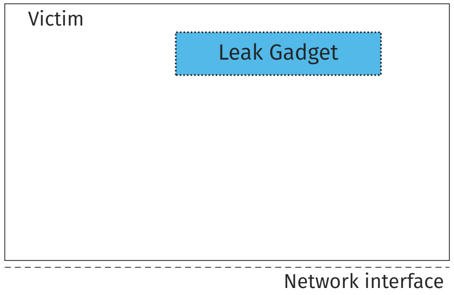
Resetting Cache State

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

NetSpectre - The Big Picture

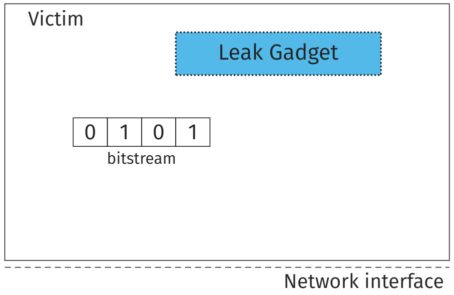


NetSpectre - The Big Picture



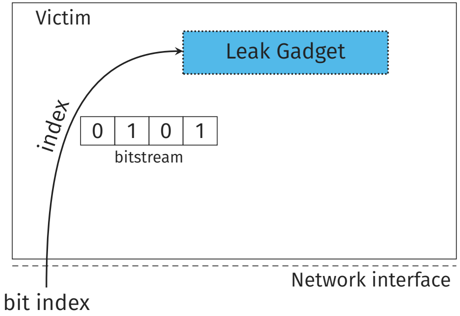
```
if (x < bitstream_length)
   if(bitstream[x])
      flag = true
```

NetSpectre - The Big Picture



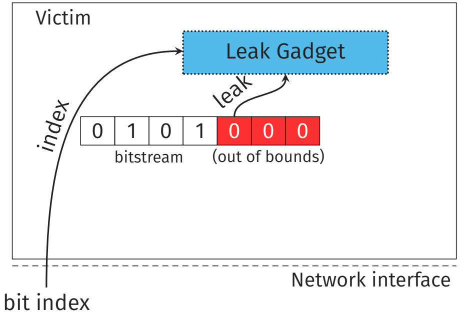
```
if (x < bitstream_length)
   if(bitstream[x])
      flag = true
```

NetSpectre - The Big Picture



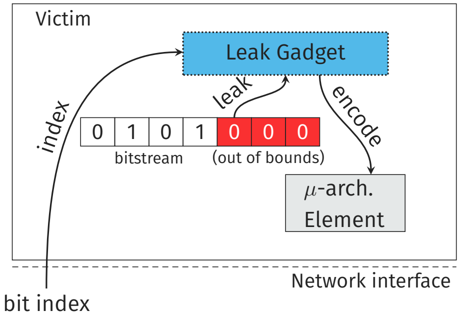
```
if (x < bitstream_length)
   if(bitstream[x])
      flag = true
```

NetSpectre - The Big Picture



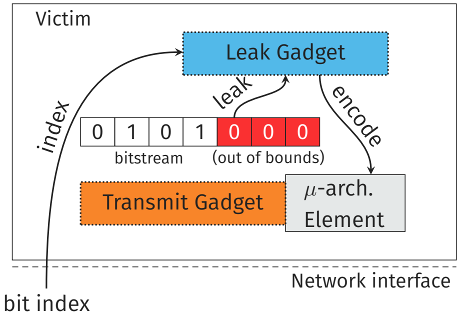
```
if (x < bitstream_length)
   if(bitstream[x])
      flag = true
```

NetSpectre - The Big Picture



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

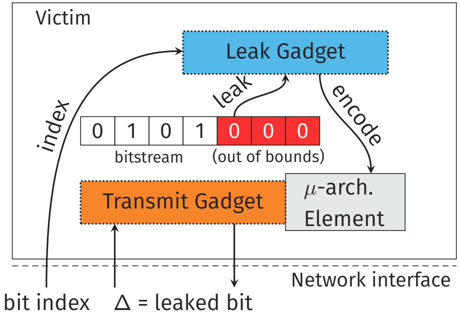
NetSpectre - The Big Picture



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

```
send(flag)
```

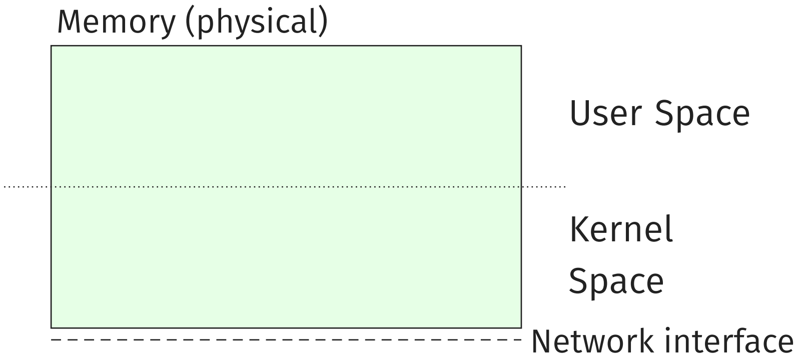
NetSpectre - The Big Picture



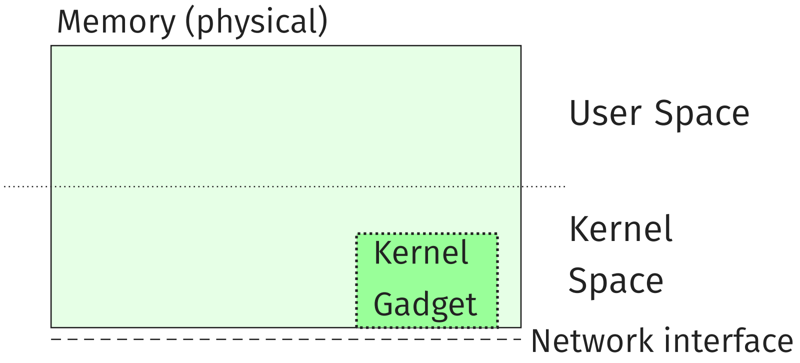
```
if (x < bitstream_length)
   if(bitstream[x])
      flag = true
```

```
send(flag)
```

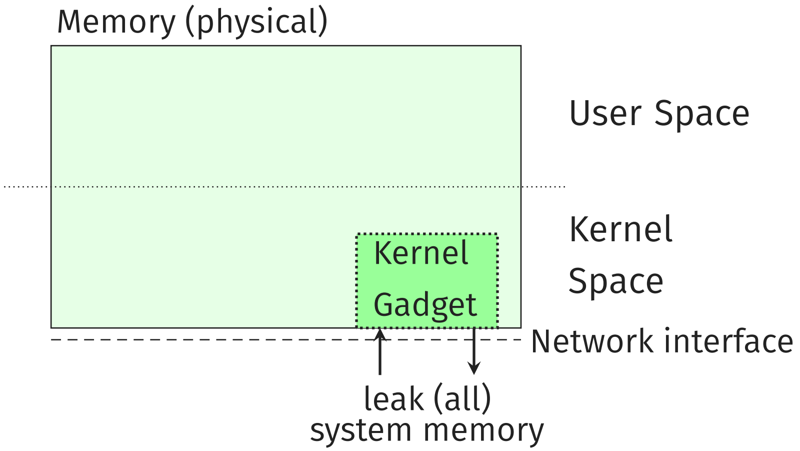
Gadgets


Memory (physical)
User Space
Kernel Space
Network interface

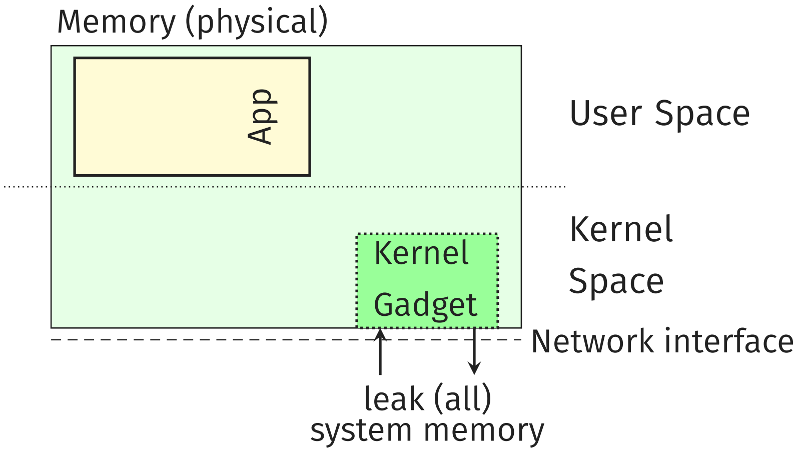
Gadgets


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

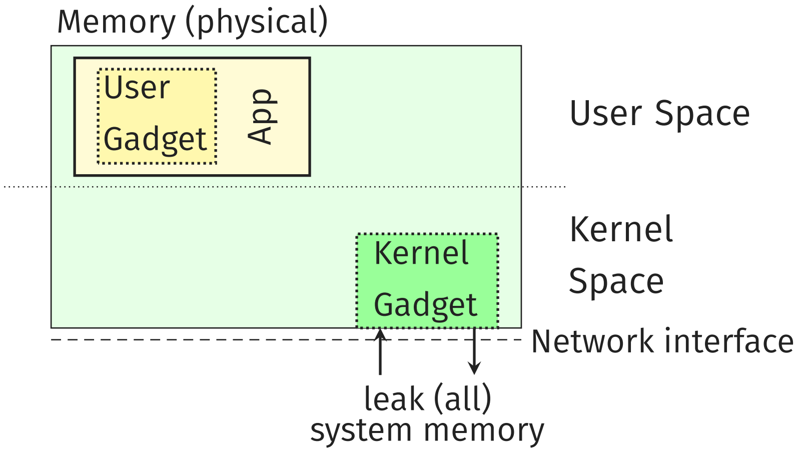
Gadgets


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

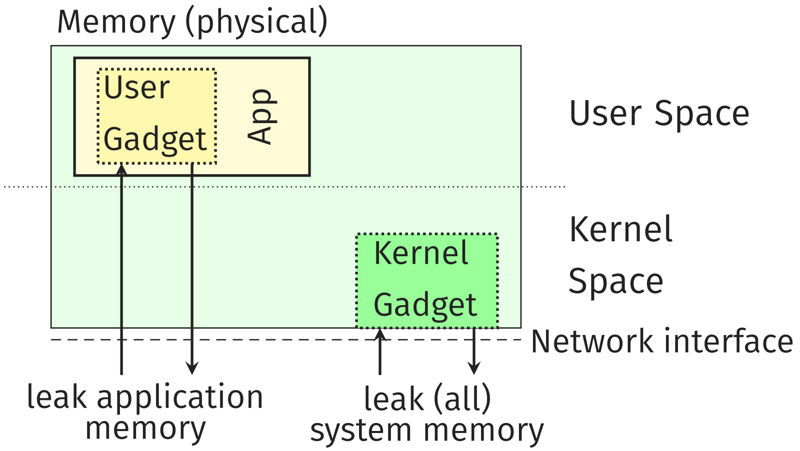
Gadgets


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

Gadgets

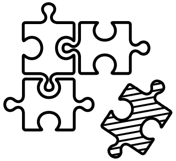

Memory (physical)
User Gadget
App
User Space
Kernel Space
Kernel Gadget
Network interface
leak (all) system memory

Gadgets



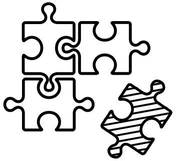
Memory (physical)
User Gadget
App
User Space
Kernel Space
Kernel Gadget
Network interface
leak application memory
leak (all) system memory

Combining Everything



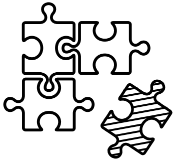
- Mistrain branch predictor with in-bounds requests

Combining Everything



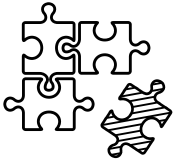
- Mistrain branch predictor with in-bounds requests
- Evict everything from cache via file download

Combining Everything



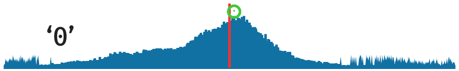
- Mistrain branch predictor with in-bounds requests
- Evict everything from cache via file download
- Leak a bit: do nothing (‘0’) or cache a memory location (‘1’)

Combining Everything



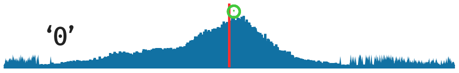
- Mistrain branch predictor with in-bounds requests
- Evict everything from cache via file download
- Leak a bit: do nothing (‘0’) or cache a memory location (‘1’)
- Measure function latency which uses the memory location

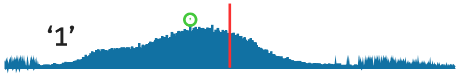
Leaking



Leaking byte ’d’ (0   )

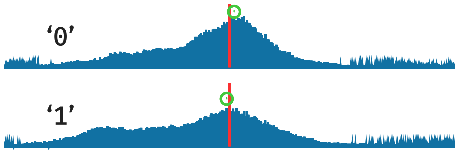
Leaking

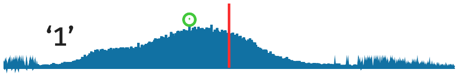




Leaking byte ’d’ (`01`)

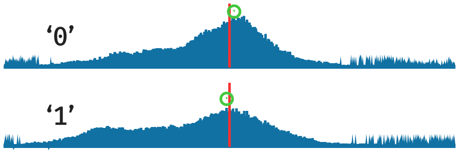
Leaking

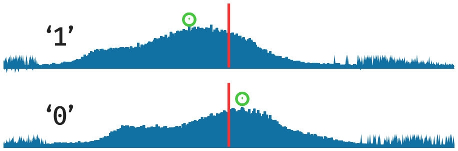




Leaking byte ’d’ (011)

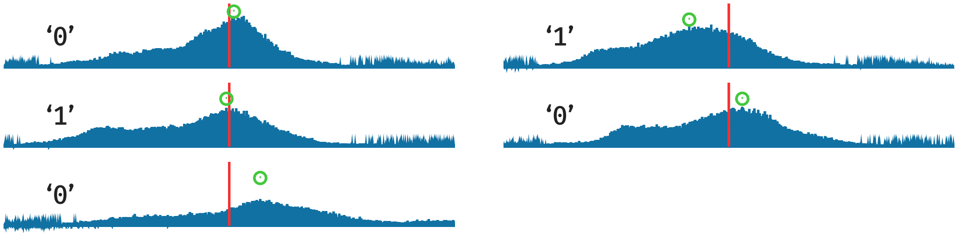
Leaking





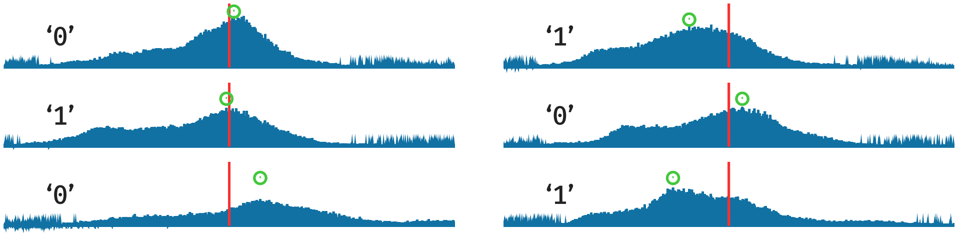
Leaking byte ’d’ (`0110`)

Leaking



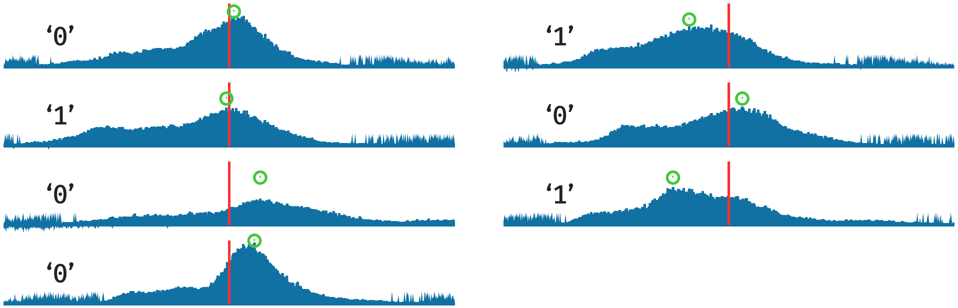
Leaking byte ’d’ (01100)

Leaking



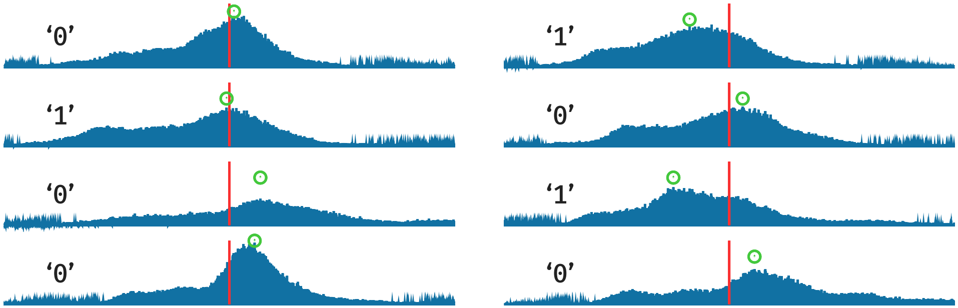
Leaking byte ’d’ (011001)

Leaking



Leaking byte 'd' (`0110010  `)

Leaking



Leaking byte ’d’ (`01100100`)

What can we exploit with them?

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)

Attack Targets

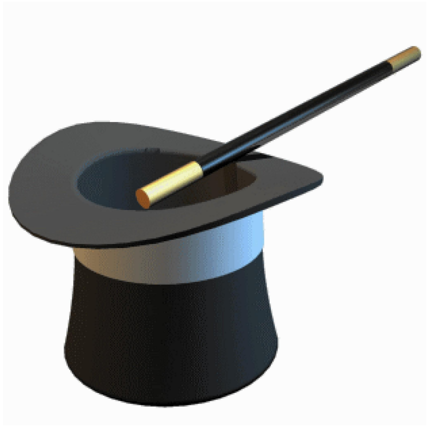
- Several possible attack targets
- Different impacts depending on target

Web/FTP Servers
(user gadget)

SSH Daemons
(user gadget)

Network Drivers
(kernel gadget)

Live Demo



That’s nice but how do we find the gadgets?

How to Find a Gadget

- Finding Spectre gadgets is still an open problem

How to Find a Gadget

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

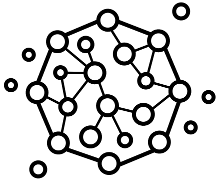
How to Find a Gadget

- 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

How to Find a Gadget

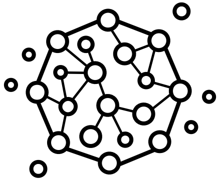
- Finding Spectre gadgets is still an open problem
- Out of all papers, only 4 show real-world gadgets
- Among them, only 2 Spectre-PHT (v1) gadgets
- Still no fully automated approach

Automated Gadget Detection



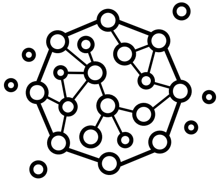
- Linux kernel uses static code analysis

Automated Gadget Detection



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

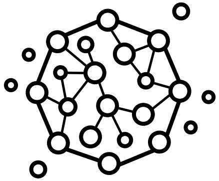
Automated Gadget Detection



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

→ Out of 736 reports only 15 real gadgets

Automated Gadget Detection

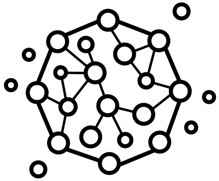


- Linux kernel uses static code analysis
- High false positive rate

$\rightarrow$ Out of 736 reports only 15 real gadgets

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

Automated Gadget Detection



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

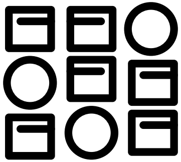
$\rightarrow$ Out of 736 reports only 15 real gadgets

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

Automated Gadget Detection

- Built 21 toy examples, 18 containing Spectre gadgets

Automated Gadget Detection



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

Automated Gadget Detection

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

Automated Gadget Detection

- 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)

Automated Gadget Detection

- 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

Automated Gadget Detection

- 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

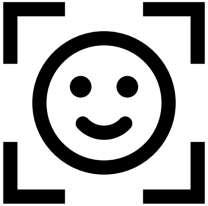
- Taint Tracking $\leftrightarrow$ mark all input as evil
- If input x flows into branch $x < size$, the branch is marked as tainted

- 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

Challenges in Identifying Gadgets

- Not clear how a Spectre gadget can look like

Challenges in Identifying Gadgets

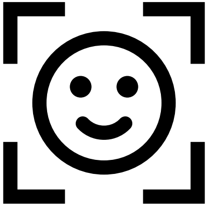


- Not clear how a Spectre gadget can look like
- Potentially many different forms

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

Previously Ignored Spectre Gadgets

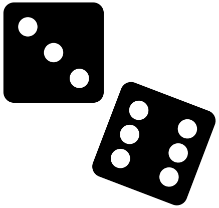
- No indirection, simple array access

Previously Ignored Spectre Gadgets

- No indirection, simple array access

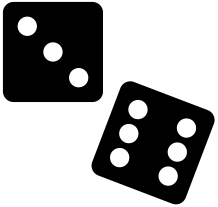
```
if (x < array_length)
  y = array[x];
```

Weaker Gadgets



- What to do with weaker gadgets?

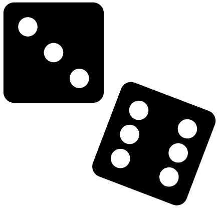
Weaker Gadgets



- What to do with weaker gadgets?

→ Break ASLR

Weaker Gadgets

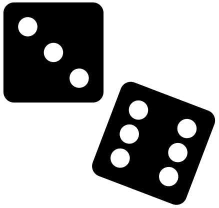


- What to do with weaker gadgets?

→ Break ASLR

- Not relevant for local Spectre attacks

Weaker Gadgets

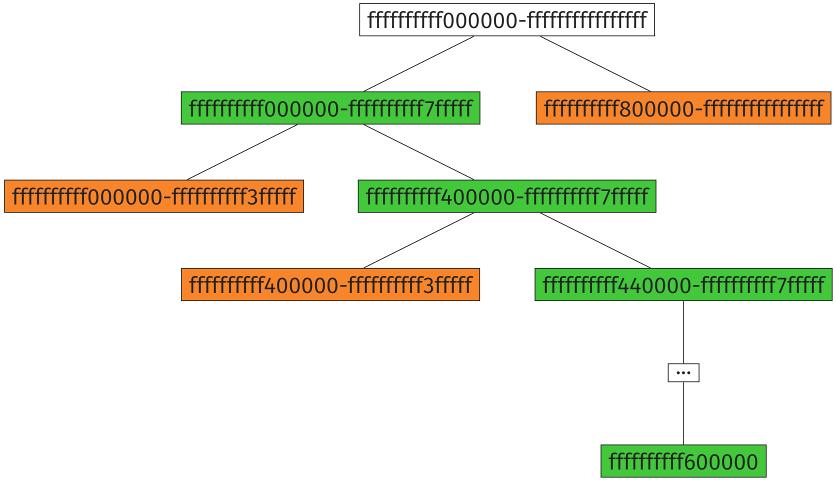


- What to do with weaker gadgets?

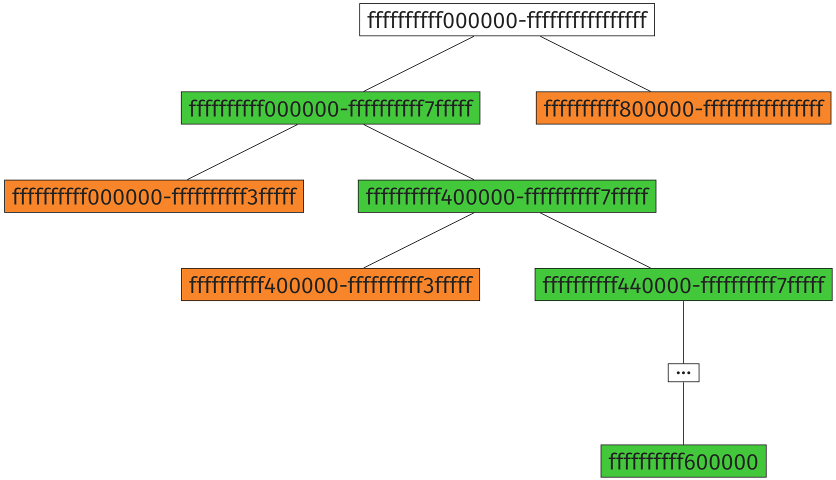
→ Break ASLR

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

Break ASLR

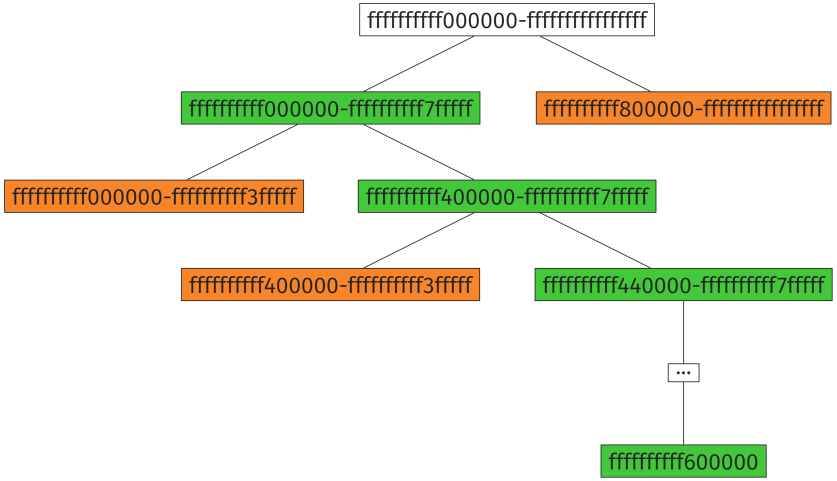

ffffffffff000000-ffffffffffffffff
ffffffffff000000-ffffffffff7fffff
ffffffffff800000-ffffffffffffffff
ffffffffff000000-ffffffffff3fffff
ffffffffff400000-ffffffffff7fffff
ffffffffff400000-ffffffffff3fffff
ffffffffff440000-ffffffffff7fffff
...
ffffffffff600000

Break ASLR

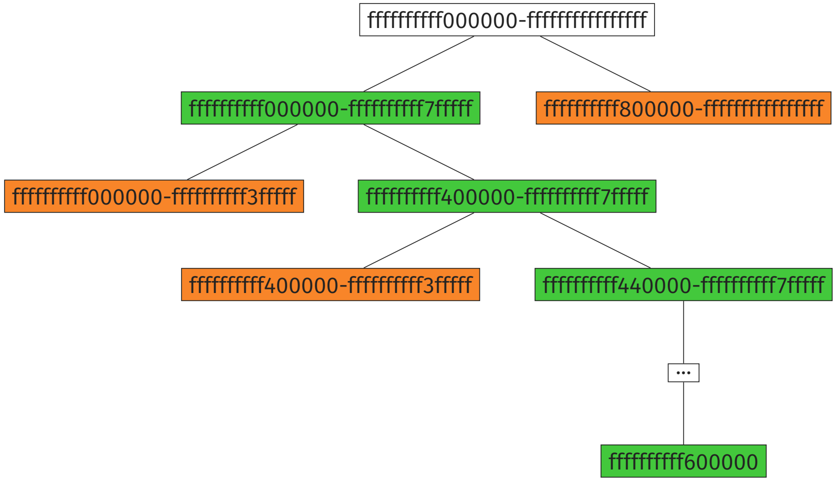


ffffffffff000000-ffffffffffffffff
ffffffffff000000-ffffffffff7fffff
ffffffffff800000-ffffffffffffffff
ffffffffff000000-ffffffffff3fffff
ffffffffff400000-ffffffffff7fffff
ffffffffff400000-ffffffffff3fffff
ffffffffff440000-ffffffffff7fffff
...
ffffffffff600000

Break ASLR

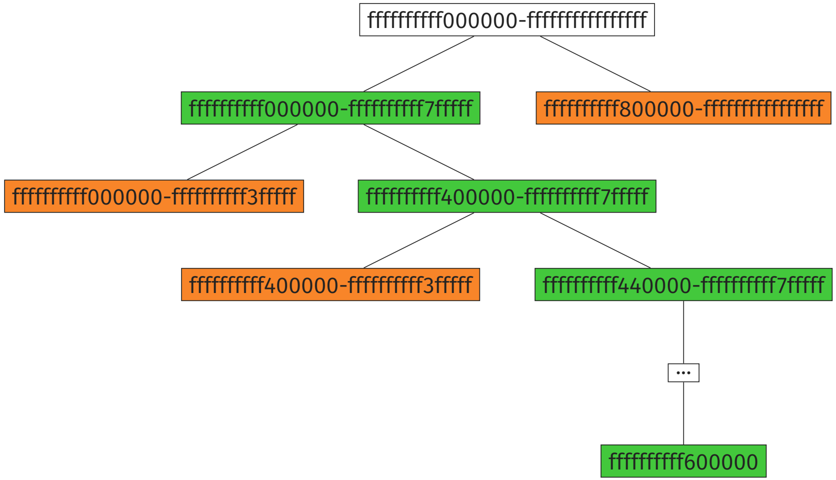

ffffffffff000000-ffffffffffffffff
ffffffffff000000-ffffffffff7fffff
ffffffffff800000-ffffffffffffffff
ffffffffff000000-ffffffffff3fffff
ffffffffff400000-ffffffffff7fffff
ffffffffff400000-ffffffffff3fffff
ffffffffff440000-ffffffffff7fffff
...
ffffffffff600000

Break ASLR

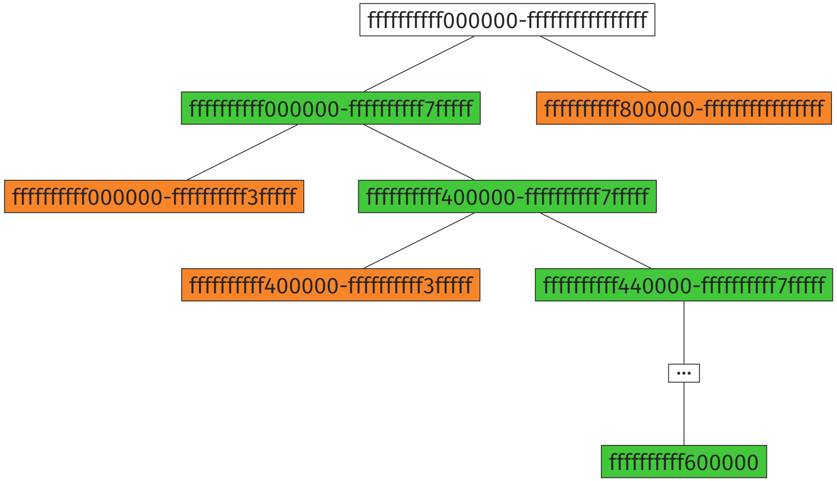


ffffffffff000000-ffffffffffffffff
ffffffffff000000-ffffffffff7fffff
ffffffffff800000-ffffffffffffffff
ffffffffff000000-ffffffffff3fffff
ffffffffff400000-ffffffffff7fffff
ffffffffff400000-ffffffffff3fffff
ffffffffff440000-ffffffffff7fffff
...
ffffffffff600000

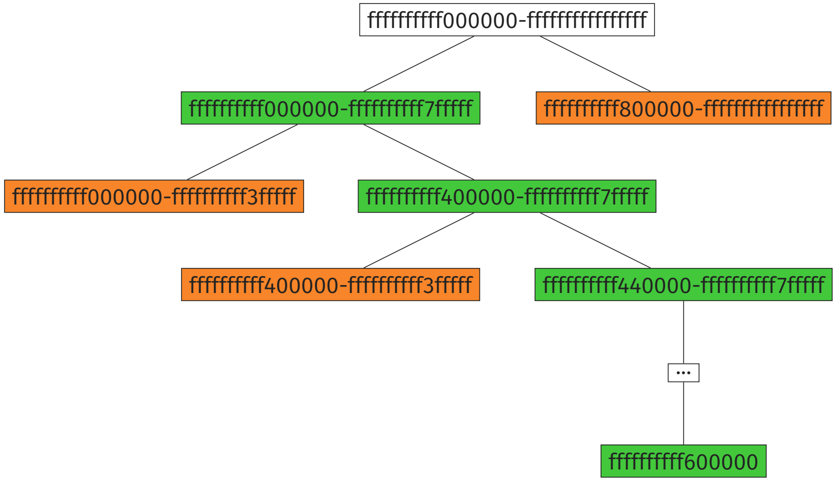
Break ASLR


ffffffffff000000-ffffffffffffffff
ffffffffff000000-ffffffffff7fffff
ffffffffff800000-ffffffffffffffff
ffffffffff000000-ffffffffff3fffff
ffffffffff400000-ffffffffff7fffff
ffffffffff400000-ffffffffff3fffff
ffffffffff440000-ffffffffff7fffff
...
ffffffffff600000

Break ASLR


ffffffffff000000-ffffffffffffffff
ffffffffff000000-ffffffffff7fffff
ffffffffff800000-ffffffffffffffff
ffffffffff000000-ffffffffff3fffff
ffffffffff400000-ffffffffff7fffff
ffffffffff400000-ffffffffff3fffff
ffffffffff440000-ffffffffff7fffff
...
ffffffffff600000

Break ASLR


ffffffffff000000-ffffffffffffffff
ffffffffff000000-ffffffffff7fffff
ffffffffff800000-ffffffffffffffff
ffffffffff000000-ffffffffff3fffff
ffffffffff400000-ffffffffff7fffff
ffffffffff400000-ffffffffff3fffff
ffffffffff440000-ffffffffff7fffff
...
ffffffffff600000

Is cache the only channel to exploit Spectre Remotely?

Spectre and the Cache

- All Spectre variants so far use the cache

Spectre and the Cache

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

Spectre and the Cache

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

Advanced Vector Instructions (AVX)

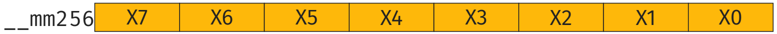
- Allow performing an operation in parallel on multiple data

Advanced Vector Instructions (AVX)

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

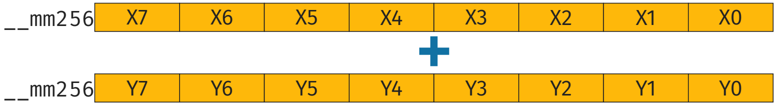
Advanced Vector Instructions (AVX)

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



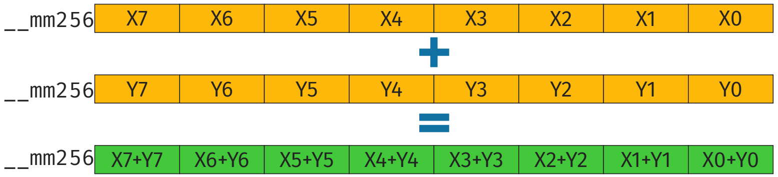
Advanced Vector Instructions (AVX)

- 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



AVX

- 256-bit instructions need a lot of power

AVX

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

AVX

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

AVX

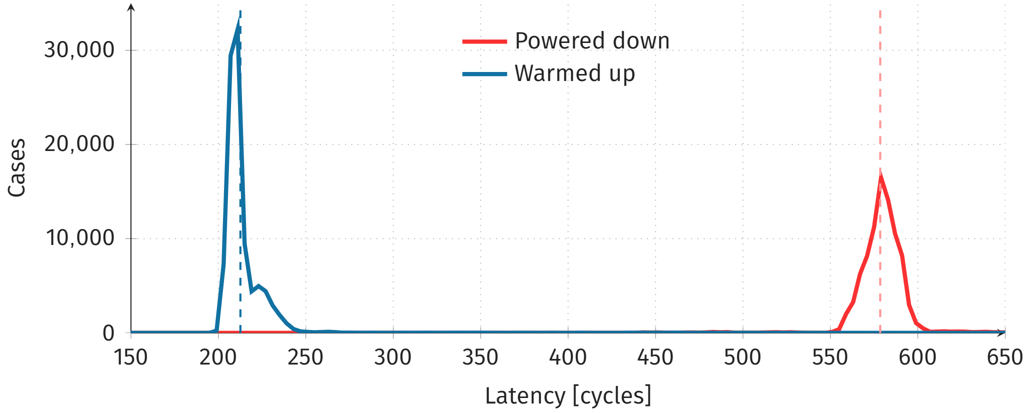
- 256-bit instructions need a lot of power
  - → On Intel, disabled by default, enabled on first use
- Requires some time to power up
- Measure execution time of AVX instruction

AVX

- 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

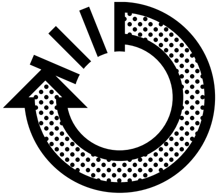


Powered down
Warmed up
Cases
30,000
20,000
10,000
0
150
200
250
300
350
400
450
500
550
600
650
Latency [cycles]

AVX Leak Gadget

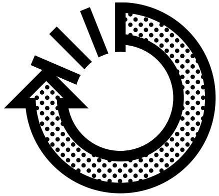
```
if (x < bitstream_length)
    if(bitstream[x])
        _mm256_instruction();
```

Resetting Microarchitectural State



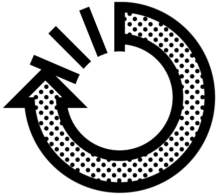
- We had to thrash cache to reset state

Resetting Microarchitectural State



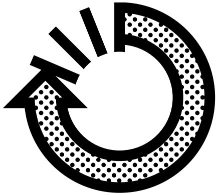
- We had to thrash cache to reset state
- Wait ≈1 ms → AVX unit powers off

Resetting Microarchitectural State



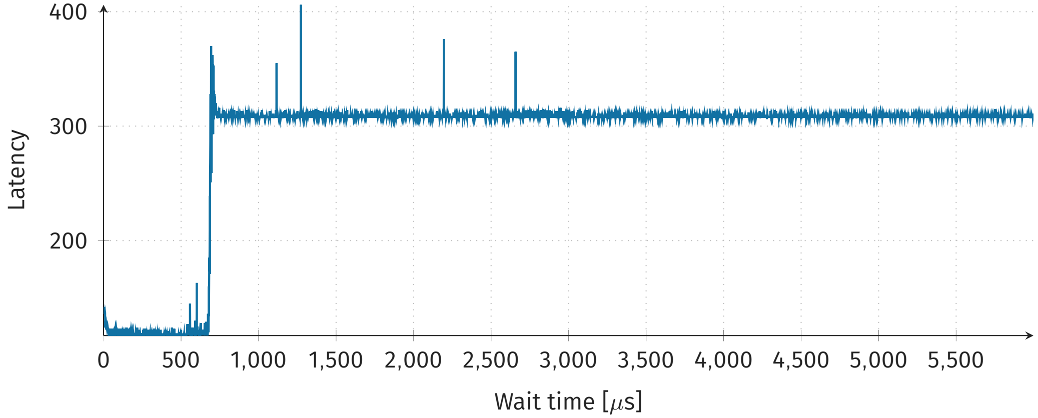
- We had to thrash cache to reset state
- Wait ≈1 ms → AVX unit powers off
- More efficient and stealthier than constantly downloading a file

Resetting Microarchitectural State

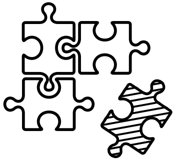


- We had to thrash cache to reset state
- Wait $\approx$1 ms $\rightarrow$ AVX unit powers off
- More efficient and stealthier than constantly downloading a file
- $\rightarrow$ higher performance than cache covert channel

AVX Cooldown

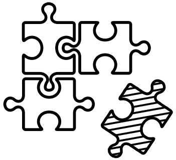


AVX-based NetSpectre



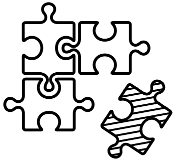
1. Mistrain branch predictor with in-bounds requests

AVX-based NetSpectre



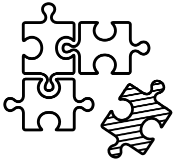
1. Mistrain branch predictor with in-bounds requests
2. Wait for AVX unit to power off (1ms)

AVX-based NetSpectre



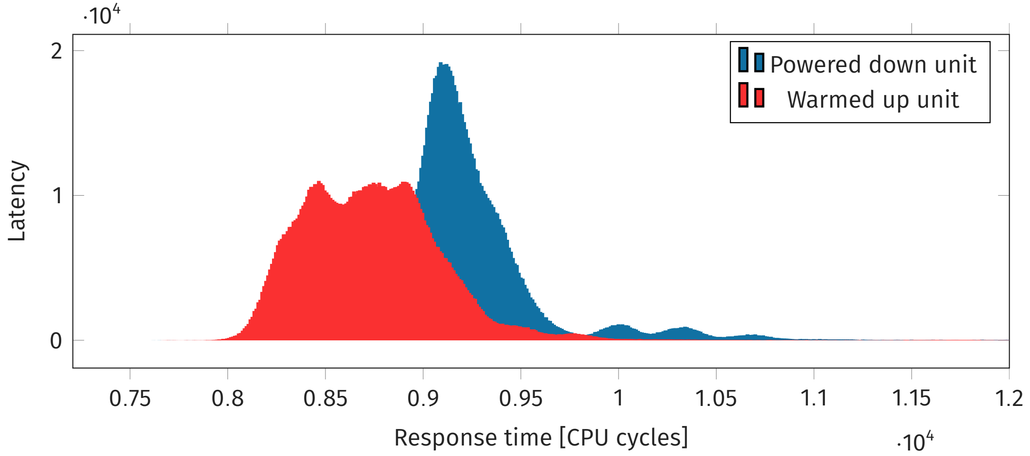
1. Mistrain branch predictor with in-bounds requests
2. Wait for AVX unit to power off (1ms)
3. Leak a bit: do nothing (‘0’) or power AVX unit (‘1’)

AVX-based NetSpectre



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

AVX Network



·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

Results

Test Environments

- NetSpectre tested in various environments

Test Environments

- NetSpectre tested in various environments

i5-6200U, i7-8550U

Test Environments

- NetSpectre tested in various environments

i5-6200U, i7-8550U

i7-6700K, i7-8700K

Test Environments

- NetSpectre tested in various environments

i5-6200U, i7-8550U

i7-6700K, i7-8700K

Skylake Xeon

Test Environments

- NetSpectre tested in various environments

i5-6200U, i7-8550U

i7-6700K, i7-8700K

Skylake Xeon

ARM Cortex A75

Attack Results

- Local Network (1 000 000 measurements/bit)

Attack Results

- Local Network (1 000 000 measurements/bit)

30 min/byte

Attack Results

- Local Network (1 000 000 measurements/bit)

30 min/byte

8 min/byte

Attack Results

- Local Network (1 000 000 measurements/bit)

30 min/byte

8 min/byte

- Cloud (20 000 000 measurements/bit)

Attack Results

- Local Network (1 000 000 measurements/bit)

30 min/byte

8 min/byte

- Cloud (20 000 000 measurements/bit)

1 h/bit

How to prevent NetSpectre

Network Countermeasures

- Mitigating NetSpectre

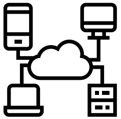
Network Countermeasures

- Mitigating NetSpectre

Network side

Network Countermeasures

- Mitigating NetSpectre



Network side

Fix Spectre

Network Countermeasures

- Prevent NetSpectre on the network side

Network Countermeasures

- Prevent NetSpectre on the network side

Firewalls and DDoS protections

Network Countermeasures

- Prevent NetSpectre on the network side

Firewalls and DDoS protections

Add random noise to packets

Network Countermeasures

- Prevent NetSpectre on the network side

Firewalls and DDoS protections

Add random noise to packets

Network segmentation

Spectre Countermeasures

- Prevent (Net)Spectre on the system side

Spectre Countermeasures

- Prevent (Net)Spectre on the system side

Hardware Fixes

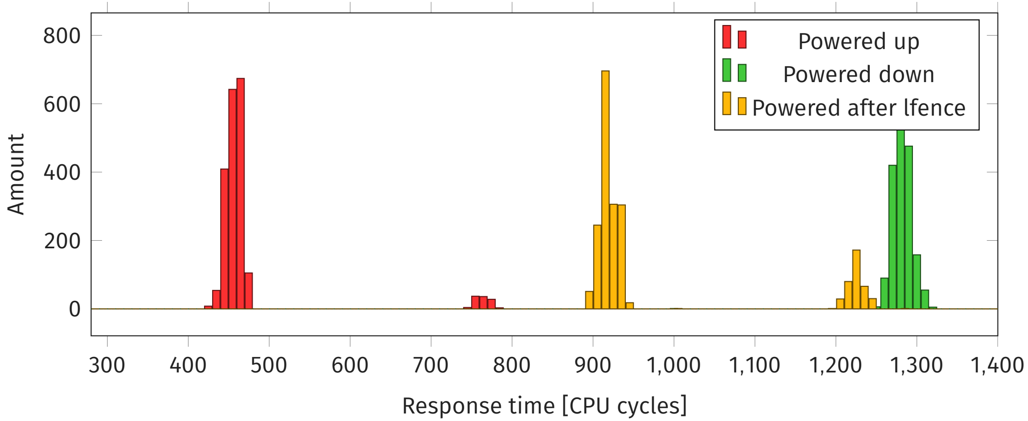
Spectre Countermeasures

- Prevent (Net)Spectre on the system side

Hardware Fixes

Software Changes

AVX vs single lfence



Limitations

- NetSpectre requires a fast and stable network connection

Limitations

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

Limitations

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

Limitations

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

Limitations

- 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?

Limitations

- 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

Limitations

- 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

Limitations

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

→ possible in the near future?

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

What do we learn from it?

- Gadgets are more versatile than expected

What do we learn from it?

- 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 $\rightarrow$ more research required

Take Aways

- 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