REPRODUCING SPECTRE ATTACK WITH GEM5: HOW TO DO IT RIGHT?

PIERRE AYOUB - EURECOM

PIERRE.AYOUB@EURECOM.FR

CLÉMENTINE MAURICE - UNIV LILLE, CNRS, INRIA

CLEMENTINE.MAURICE@INRIA.FR

EUROSEC'21

26 APRIL, 2021

INTRODUCTION

WHOAMI

WHOAMI

Intern

WHOAMI

Intern

WHOAMI

Intern

Ph.D. Student

WHOAMI

Intern

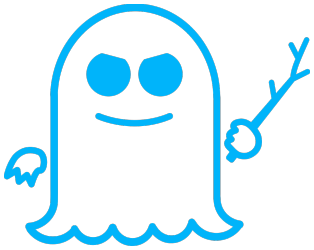
Ph.D. Student

ABOUT THIS PRESENTATION

THE INITIAL IDEA

THE INITIAL IDEA

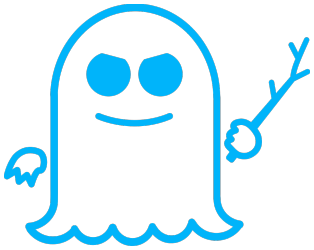
- Can we simulate transient execution attacks? $\Rightarrow$ **Spectre**.



SPECTRE

THE INITIAL IDEA

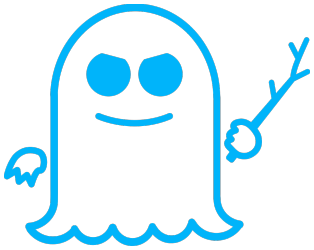
- Can we simulate transient execution attacks? $\Rightarrow$ **Spectre**.
- Why?



SPECTRE

THE INITIAL IDEA

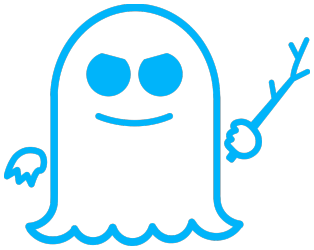
- Can we simulate transient execution attacks? $\Rightarrow$ **Spectre**.
- Why?
  - Rely on tiny CPU details to work $\Rightarrow$ **difficult to simulate**.



SPECTRE

THE INITIAL IDEA

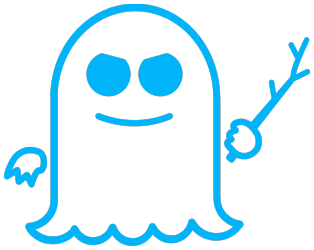
- Can we simulate transient execution attacks? $\Rightarrow$ **Spectre**.
- Why?
  - Rely on tiny CPU details to work $\Rightarrow$ **difficult to simulate**.
  - Hard to study and reproduce $\Rightarrow$ **simulation could be helping**.



SPECTRE

THE INITIAL IDEA

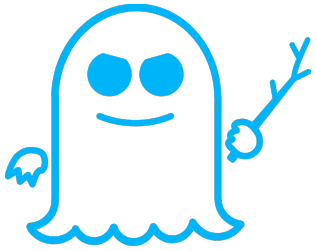
- Can we simulate transient execution attacks? $\Rightarrow$ **Spectre**.
- Why?
  - Rely on tiny CPU details to work $\Rightarrow$ **difficult to simulate**.
  - Hard to study and reproduce $\Rightarrow$ **simulation could be helping**.
- Comparing a **real system** and a **simulated system**:



SPECTRE

THE INITIAL IDEA

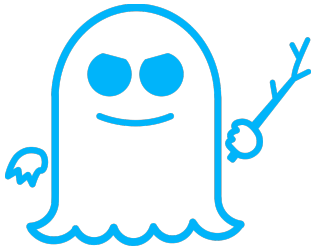
- Can we simulate transient execution attacks? ⇒ **Spectre**.
- Why?
  - Rely on tiny CPU details to work ⇒ **difficult to simulate**.
  - Hard to study and reproduce ⇒ **simulation could be helping**.
- Comparing a **real system** and a **simulated system**:
  - **Raspberry Pi**, ARM processor.



SPECTRE

THE INITIAL IDEA

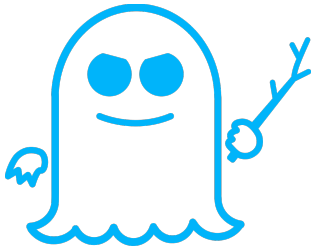
- Can we simulate transient execution attacks? ⇒ **Spectre**.
- Why?
  - Rely on tiny CPU details to work ⇒ **difficult to simulate**.
  - Hard to study and reproduce ⇒ **simulation could be helping**.
- Comparing a **real system** and a **simulated system**:
  - **Raspberry Pi**, ARM processor.
  - **gem5**, micro-architectural simulator.



SPECTRE

THE INITIAL IDEA

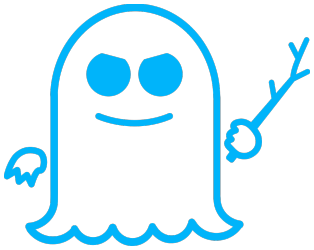
- Can we simulate transient execution attacks? ⇒ **Spectre**.
- Why?
  - Rely on tiny CPU details to work ⇒ **difficult to simulate**.
  - Hard to study and reproduce ⇒ **simulation could be helping**.
- Comparing a **real system** and a **simulated system**:
  - **Raspberry Pi**, ARM processor.
  - **gem5**, micro-architectural simulator.
- Goals:



SPECTRE

THE INITIAL IDEA

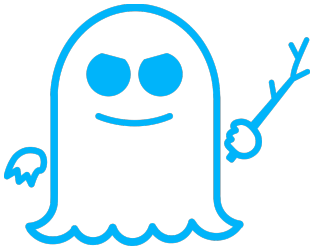
- Can we simulate transient execution attacks? $\Rightarrow$ **Spectre**.
- Why?
  - Rely on tiny CPU details to work $\Rightarrow$ **difficult to simulate**.
  - Hard to study and reproduce $\Rightarrow$ **simulation could be helping**.
- Comparing a **real system** and a **simulated system**:
  - **Raspberry Pi**, ARM processor.
  - **gem5**, micro-architectural simulator.
- Goals:
  - Attack that works **similarly on both systems**.



SPECTRE

THE INITIAL IDEA

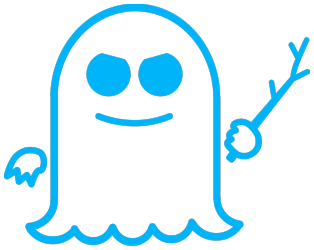
- Can we simulate transient execution attacks? ⇒ **Spectre**.
- Why?
  - Rely on tiny CPU details to work ⇒ **difficult to simulate**.
  - Hard to study and reproduce ⇒ **simulation could be helping**.
- Comparing a **real system** and a **simulated system**:
  - **Raspberry Pi**, ARM processor.
  - **gem5**, micro-architectural simulator.
- Goals:
  - Attack that works **similarly on both systems**.
  - Compare the **faithfulness of the simulation**.



SPECTRE

THE INITIAL IDEA

- Can we simulate transient execution attacks? ⇒ **Spectre**.
- Why?
  - Rely on tiny CPU details to work ⇒ **difficult to simulate**.
  - Hard to study and reproduce ⇒ **simulation could be helping**.
- Comparing a **real system** and a **simulated system**:
  - **Raspberry Pi**, ARM processor.
  - **gem5**, micro-architectural simulator.
- Goals:
  - Attack that works **similarly on both systems**.
  - Compare the **faithfulness of the simulation**.
  - Discover **how gem5 could be helpful**.



SPECTRE

WHAT REALLY HAPPENED

Theory:

Practice:

PROBLEMS

PROBLEMS

1. Available Spectre implementations **failed** on our Raspberry Pi $\Rightarrow$ **guidelines** and custom **implementation**.

PROBLEMS

1. Available Spectre implementations **failed** on our Raspberry Pi $\Rightarrow$ **guidelines** and custom **implementation**.
2. Reproducing a micro-architecture is **impossible**.

PROBLEMS

1. Available Spectre implementations **failed** on our Raspberry Pi $\Rightarrow$ **guidelines** and custom **implementation**.
2. Reproducing a micro-architecture is **impossible**.
3. gem5 needed some extension to compare it to the real system $\Rightarrow$ **patch**.

CONTRIBUTIONS

CONTRIBUTIONS

- **Guidelines** that are important for micro-architectural security research.

CONTRIBUTIONS

- **Guidelines** that are important for micro-architectural security research.
- Usage of gem5 for **helping attack development and understanding**.

CONTRIBUTIONS

- **Guidelines** that are important for micro-architectural security research.
- Usage of gem5 for **helping attack development and understanding**.
- **Simulation of Spectre** and evaluation of **faithfulness**.

CONTRIBUTIONS

- **Guidelines** that are important for micro-architectural security research.
- Usage of gem5 for **helping attack development and understanding**.
- **Simulation of Spectre** and evaluation of **faithfulness**.
- **Requirements of gem5** to simulate those attacks.

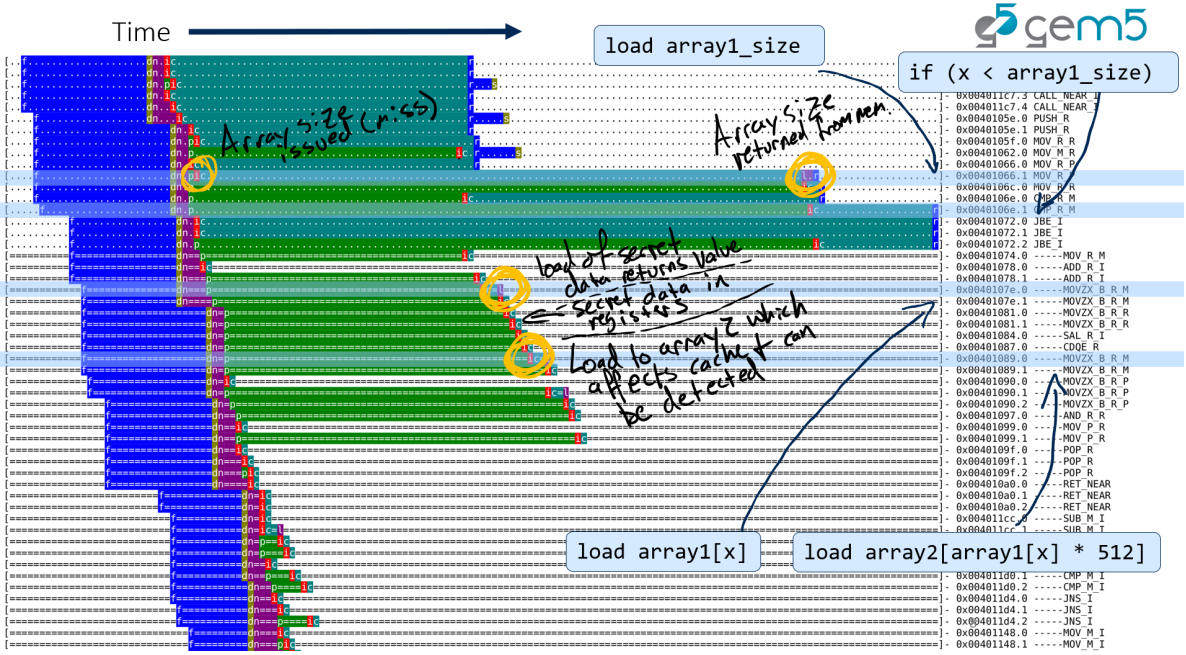
TABLE OF CONTENT

RELATED WORK

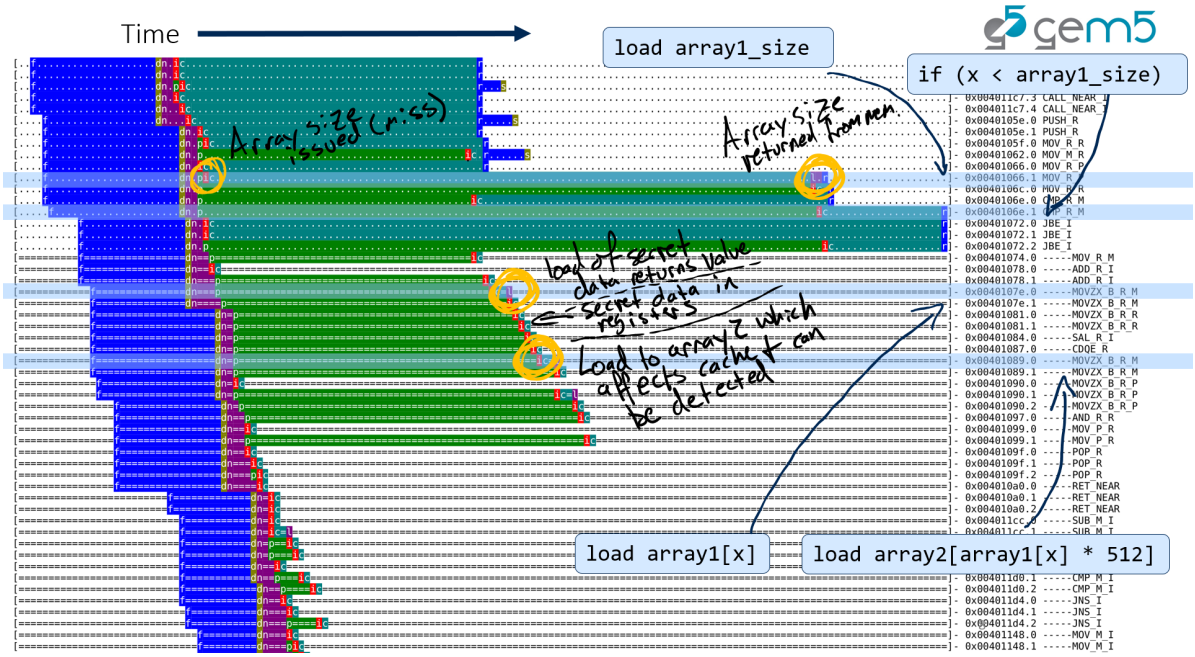
RELATED WORK

- No specific literature about simulation of micro-architectural attack.

J. LOWE-POWER - VISUALIZING SPECTRE WITH GEM5

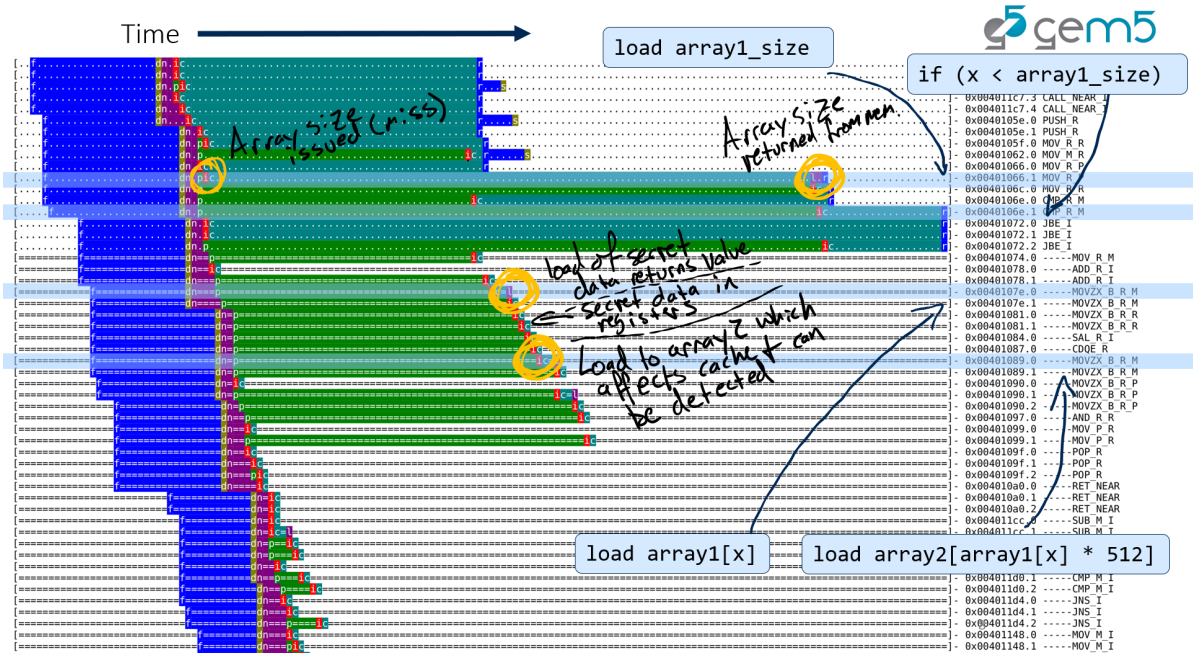


J. LOWE-POWER - VISUALIZING SPECTRE WITH GEM5



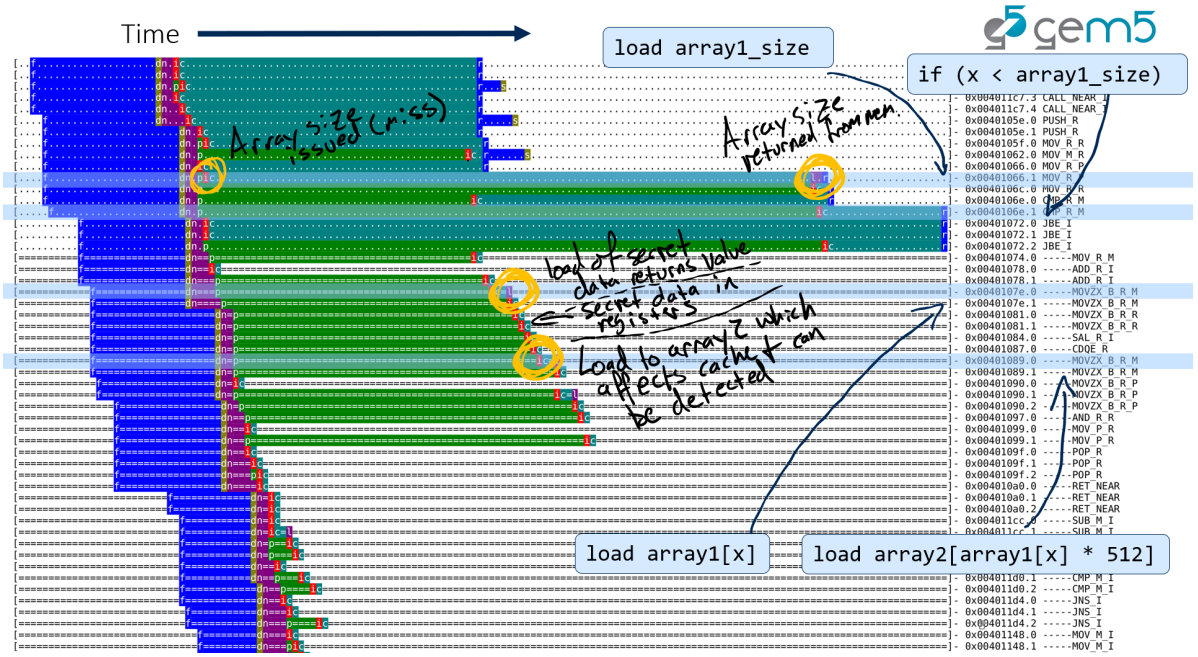
- Blog post

J. LOWE-POWER - VISUALIZING SPECTRE WITH GEM5



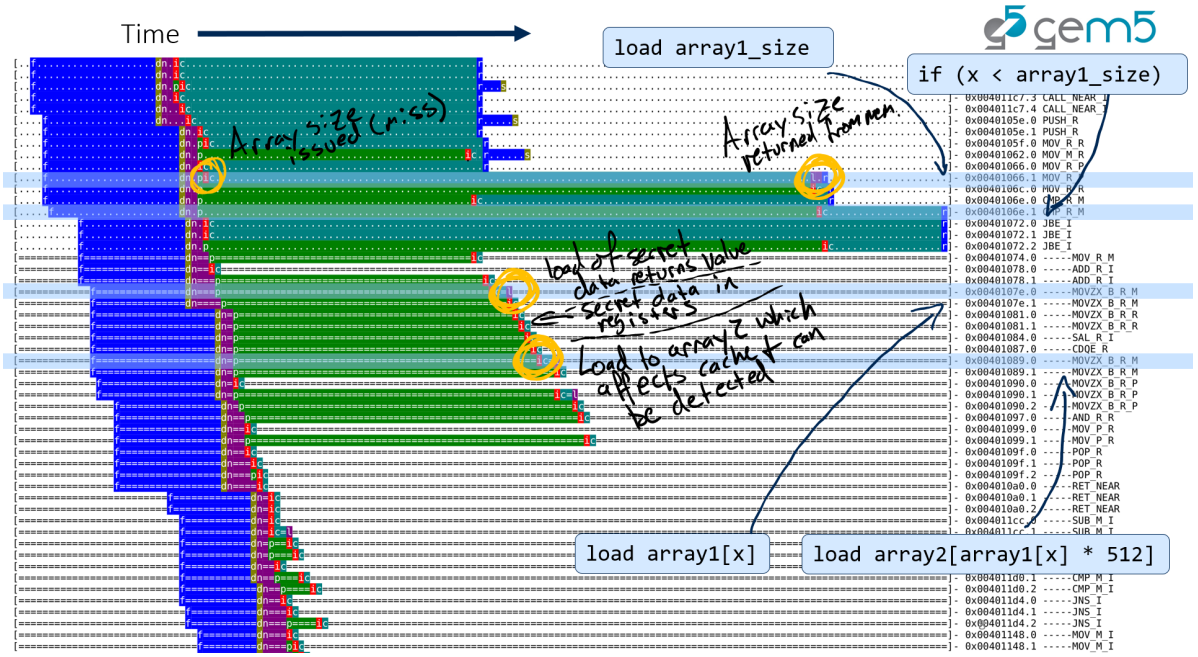
- Blog post
- x86

J. LOWE-POWER - VISUALIZING SPECTRE WITH GEM5



- Blog post
- x86
- Default gem5 configuration

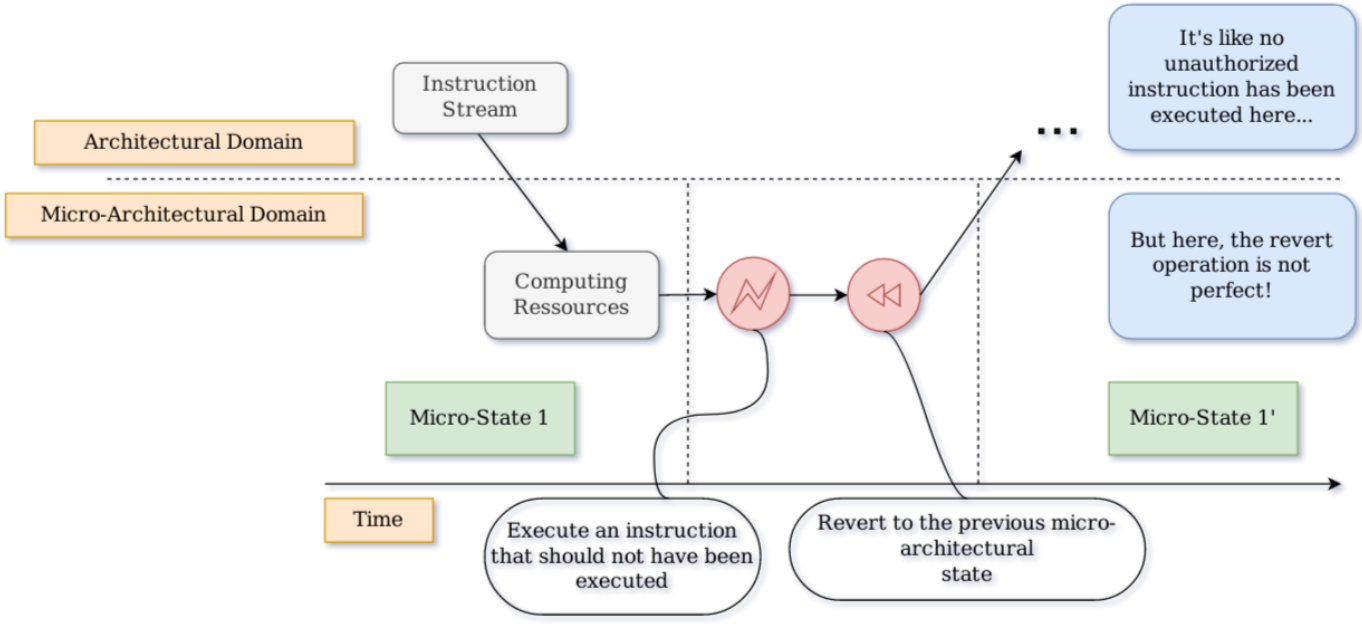
J. LOWE-POWER - VISUALIZING SPECTRE WITH GEM5



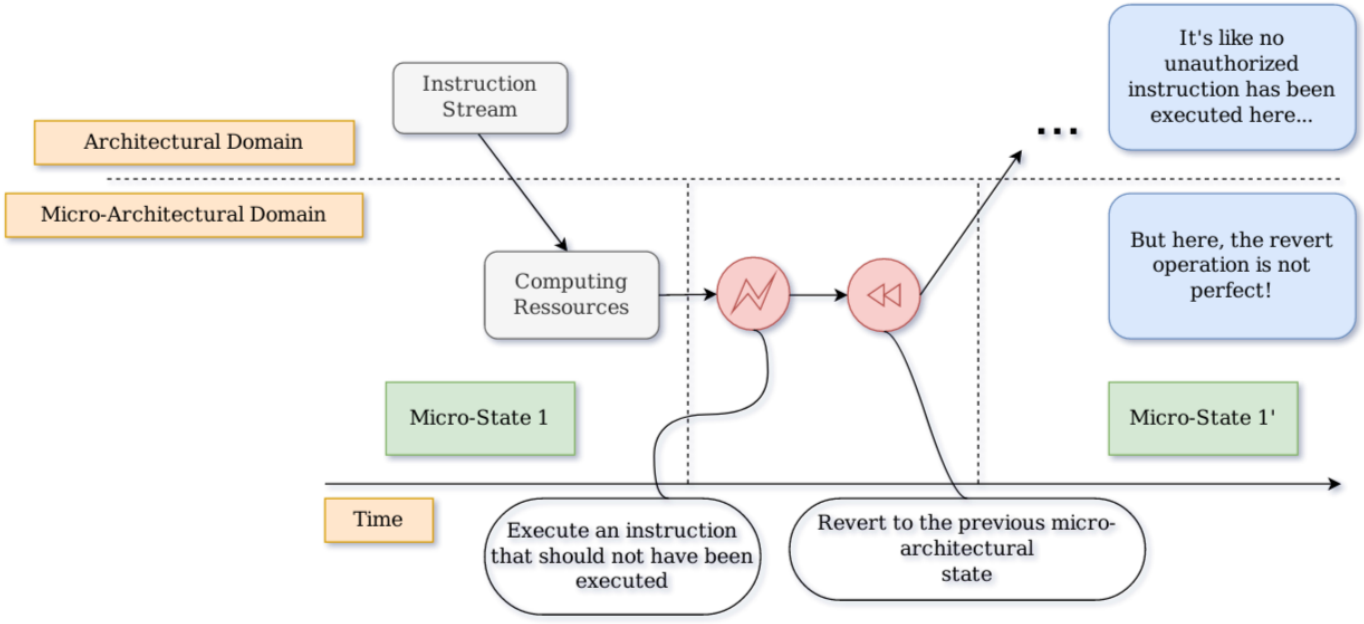
- Blog post
- x86
- Default gem5 configuration
- We wanted to go deeper!

THE SPECTRE ATTACK

A TRANSIENT INSTRUCTION?

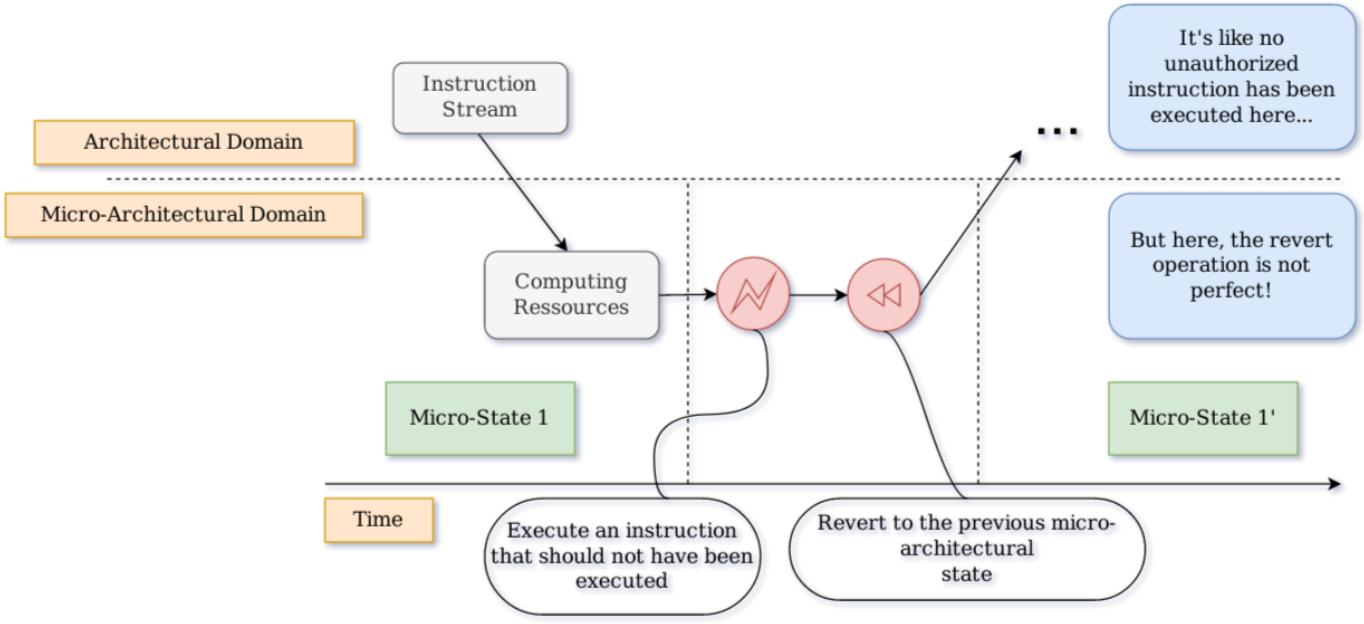


A TRANSIENT INSTRUCTION?



Summary

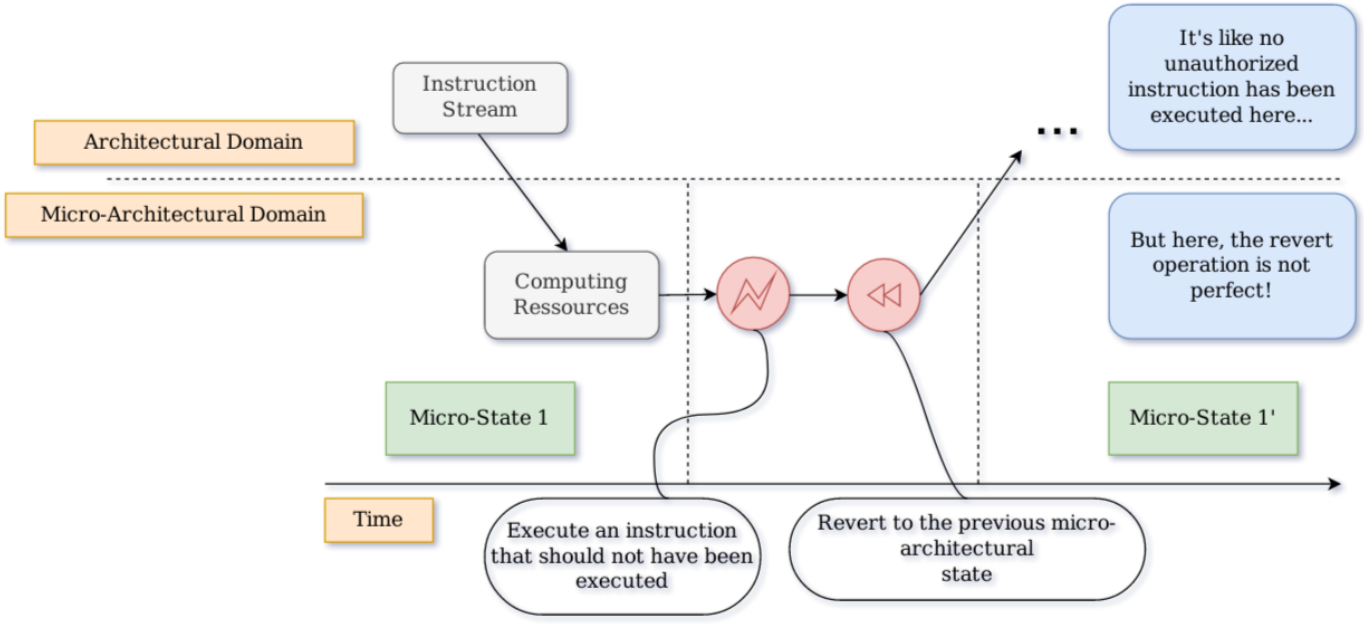
A TRANSIENT INSTRUCTION?



Summary

- An instruction has been **transiently executed** if it affects the CPU *micro-architectural state* – leaving its architectural state as prior the execution.

A TRANSIENT INSTRUCTION?



Summary

- An instruction has been **transiently executed** if it affects the CPU *micro-architectural state* – leaving its architectural state as prior the execution.
- If the the new micro-architectural state depends on a secret and the attacker is able to probe it, he can **recover the secret**.

THE BRANCH PREDICTOR

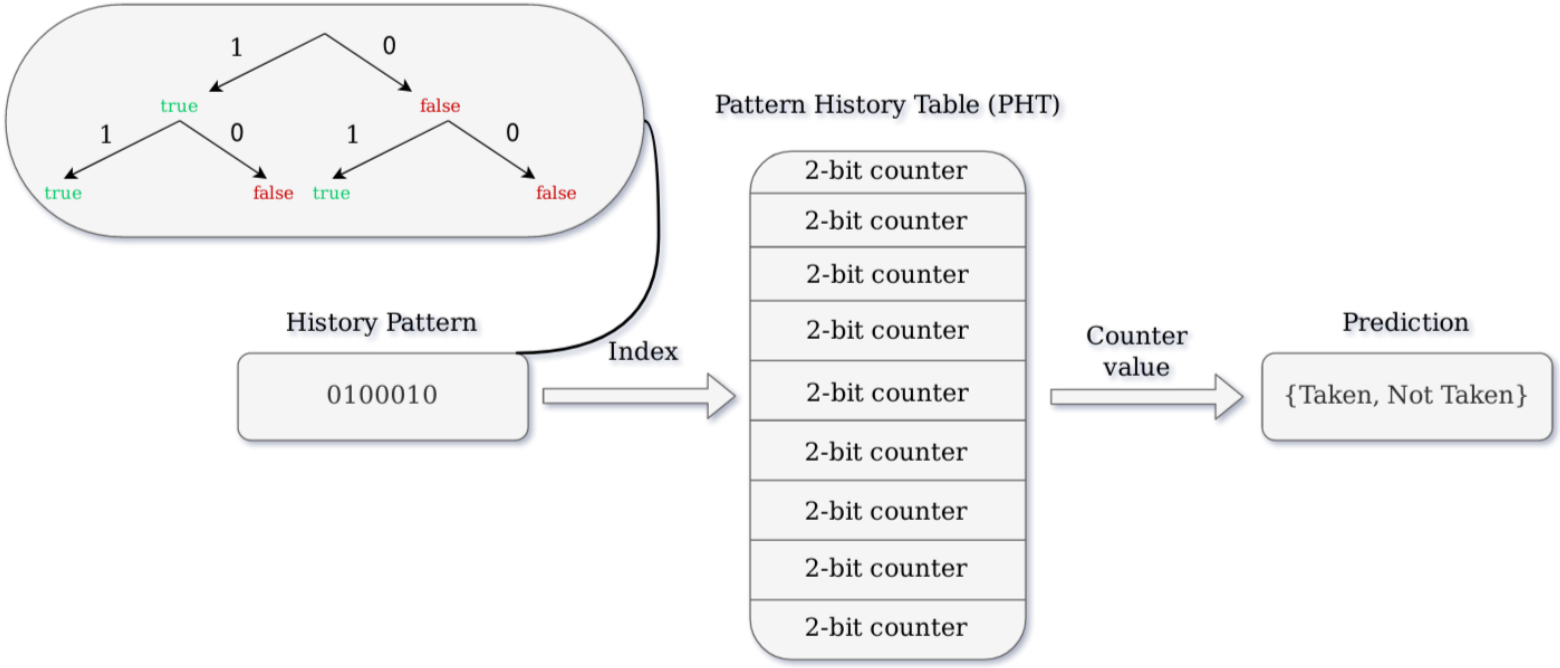
THE BRANCH PREDICTOR

- **Predict instruction flow** when branches are encountered.

THE BRANCH PREDICTOR

- **Predict instruction flow** when branches are encountered.
- Prediction is **dynamic**, it is based on previous execution.

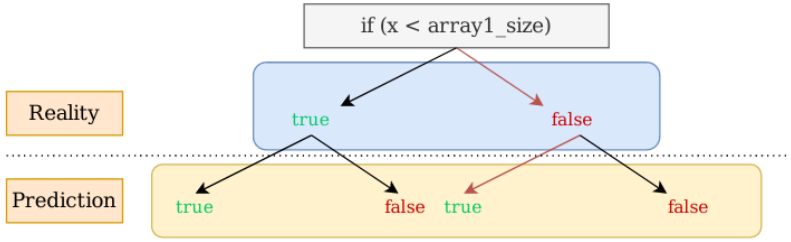
THE PHT, A STRUCTURE USED BY THE BRANCH PREDICTOR



HOW DOES SPECTRE WORK?

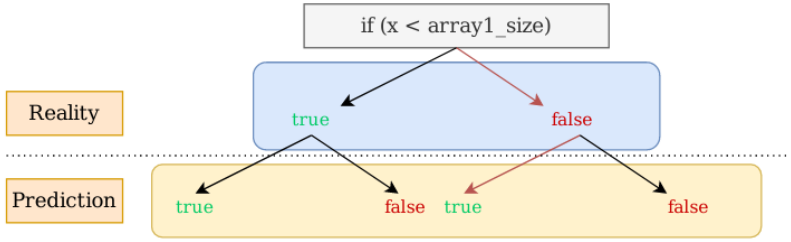
WE NEED A TARGET

```
void victim_function(int x)
{
    ...
    if (x < array1_size)
        y = array2[array1[x]];
    ...
}
```



WE NEED A TARGET

```
void victim_function(int x)
{
    ...
    if (x < array1_size)
        y = array2[array1[x]];
    ...
}
```



- If x is malicious, `array1[x]` is the secret value!

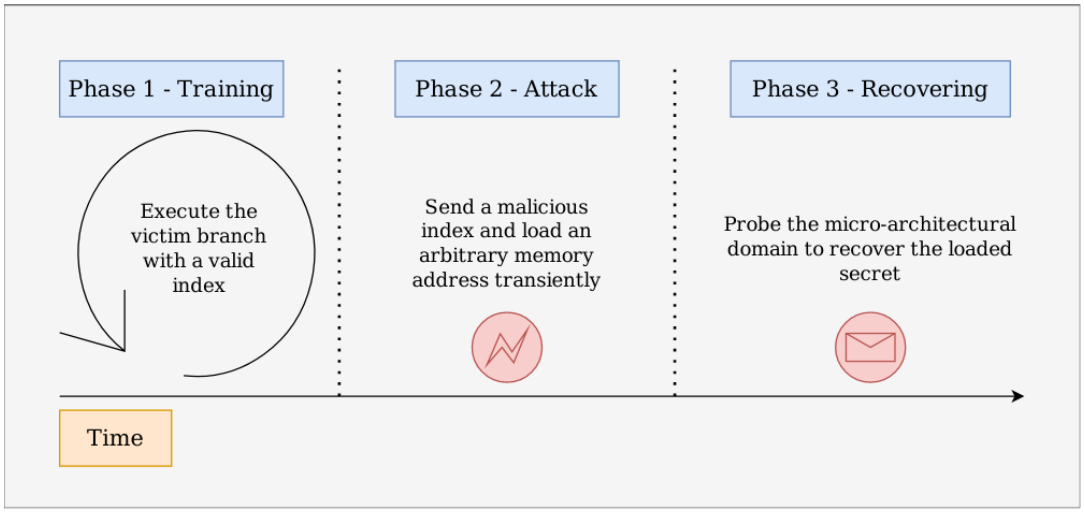
STEPS OF THE ATTACK

STEPS OF THE ATTACK

Summary

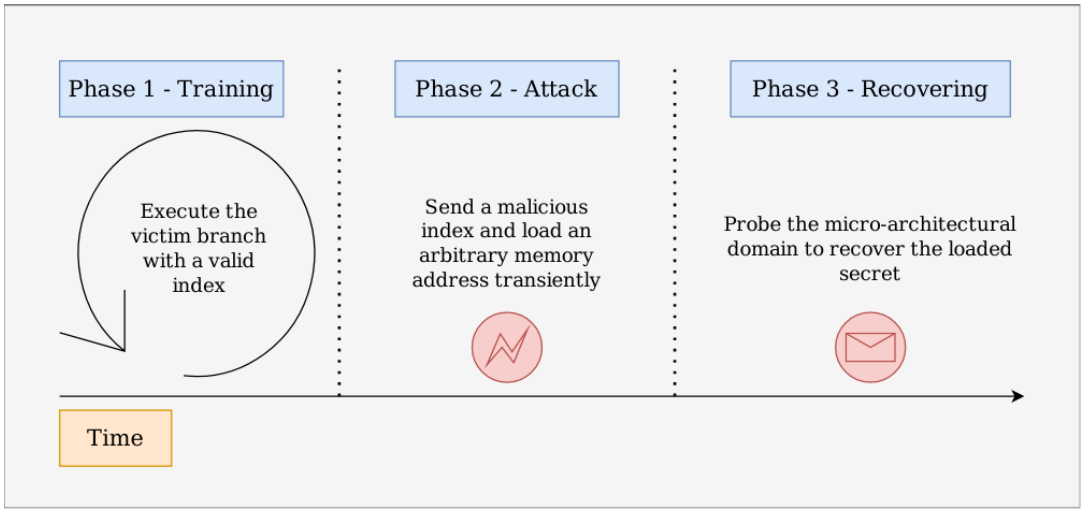
STEPS OF THE ATTACK

Summary



STEPS OF THE ATTACK

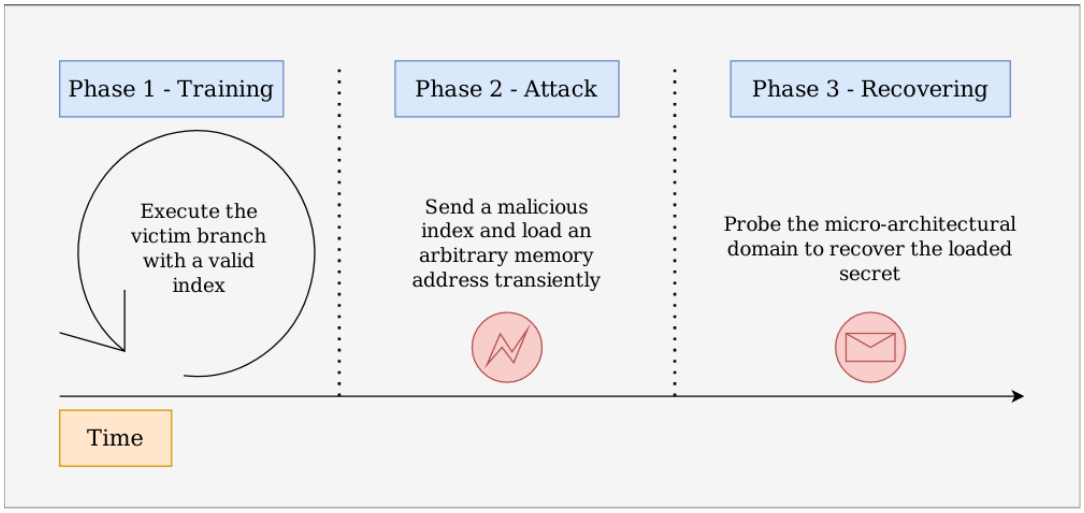
Summary



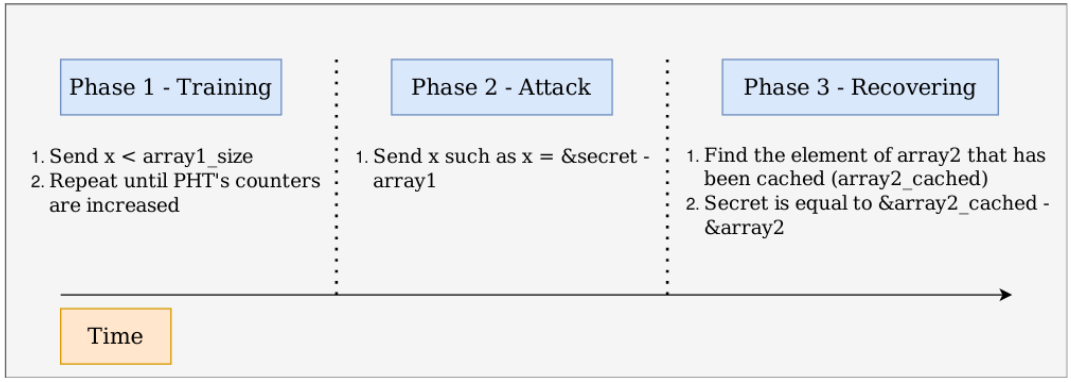
Details

STEPS OF THE ATTACK

Summary



Details



GUIDELINES: HOW TO DEVELOP AND REPRODUCE THE ATTACK

GUIDELINES: HOW TO DEVELOP AND REPRODUCE THE ATTACK

- Why?

GUIDELINES: HOW TO DEVELOP AND REPRODUCE THE ATTACK

- Why?
  - **Hard to reproduce** the attack using **already existing implementation**.

GUIDELINES: HOW TO DEVELOP AND REPRODUCE THE ATTACK

- Why?
  - **Hard to reproduce** the attack using **already existing implementation**.
  - **Hard to develop** a functional attack on a **vulnerable processor**.

GUIDELINES: HOW TO DEVELOP AND REPRODUCE THE ATTACK

- Why?
  - **Hard to reproduce** the attack using **already existing implementation**.
  - **Hard to develop** a functional attack on a **vulnerable processor**.
- Refer to the paper for more details.

DEVELOPMENT

DEVELOPMENT

Compiler version, compiler and manual optimizations

DEVELOPMENT

Compiler version, compiler and manual optimizations

DEVELOPMENT

Compiler version, compiler and manual optimizations

Timer for covert channel

DEVELOPMENT

Compiler version, compiler and manual optimizations

Timer for covert channel

DEVELOPMENT

Compiler version, compiler and manual optimizations
Timer for covert channel
Prefetcher, re-ordering

DEVELOPMENT

Compiler version, compiler and manual optimizations
Timer for covert channel
Prefetcher, re-ordering

DEVELOPMENT

Compiler version, compiler and manual optimizations
Timer for covert channel
Prefetcher, re-ordering

- Very tricky because we make blind assumptions.

DEVELOPMENT

Compiler version, compiler and manual optimizations
Timer for covert channel
Prefetcher, re-ordering

- Very tricky because we make blind assumptions.
- gem5 can help resolve this!

DEVELOPMENT

Compiler version, compiler and manual optimizations

Timer for covert channel

Prefetcher, re-ordering

- Very tricky because we make blind assumptions.
- gem5 can help resolve this!

Transient execution window

DEVELOPMENT

Compiler version, compiler and manual optimizations

Timer for covert channel

Prefetcher, re-ordering

- Very tricky because we make blind assumptions.
- gem5 can help resolve this!

Transient execution window

- Time during which transient executions can happen.

DEVELOPMENT

Compiler version, compiler and manual optimizations

Timer for covert channel

Prefetcher, re-ordering

- Very tricky because we make blind assumptions.
- gem5 can help resolve this!

Transient execution window

- Time during which transient executions can happen.
- *e.g.* `if (array1_size < 1)` vs. `if ((float) x / (float) array1_size < 1)`

DEVELOPMENT

Compiler version, compiler and manual optimizations

Timer for covert channel

Prefetcher, re-ordering

- Very tricky because we make blind assumptions.
- gem5 can help resolve this!

Transient execution window

- Time during which transient executions can happen.
- *e.g.* `if (array1_size < 1)` vs. `if ((float) x / (float) array1_size < 1)`

Our implementation is also full of tips in the comments, feel free to look at it!

REPRODUCIBILITY

REPRODUCIBILITY

- **Pinning**

REPRODUCIBILITY

- **Pinning**
- **Page size**

REPRODUCIBILITY

- **Pinning**
- **Page size**
- **Frequency**

REPRODUCIBILITY

- **Pinning**
- **Page size**
- **Frequency**
- **Mitigations**

THE GEM5 SIMULATOR

GEM5

GEM5

- Micro-architectural simulator, **cycle-accurate**.

GEM5

- Micro-architectural simulator, **cycle-accurate**.
- **State-of-the-art** project, started in 2011.

GEM5

- Micro-architectural simulator, **cycle-accurate**.
- **State-of-the-art** project, started in 2011.

Parts

GEM5

- Micro-architectural simulator, **cycle-accurate**.
- **State-of-the-art** project, started in 2011.

Parts

- **C++ core** where logic is programmed.
- **Python interface** where systems are built.

GEM5

- Micro-architectural simulator, **cycle-accurate**.
- **State-of-the-art** project, started in 2011.

Parts

- **C++ core** where logic is programmed.
- **Python interface** where systems are built.

Architecture

GEM5

- Micro-architectural simulator, **cycle-accurate**.
- **State-of-the-art** project, started in 2011.

Parts

- **C++ core** where logic is programmed.
- **Python interface** where systems are built.

Architecture

Alpha, **ARM**, Power, SPARC, x86, MIPS, RISC-V.

GEM5

- Micro-architectural simulator, **cycle-accurate**.
- **State-of-the-art** project, started in 2011.

Parts

- **C++ core** where logic is programmed.
- **Python interface** where systems are built.

Architecture

Alpha, **ARM**, Power, SPARC, x86, MIPS, RISC-V.

Generic Micro-Architecture

GEM5

- Micro-architectural simulator, **cycle-accurate**.
- **State-of-the-art** project, started in 2011.

Parts

- **C++ core** where logic is programmed.
- **Python interface** where systems are built.

Architecture

Alpha, **ARM**, Power, SPARC, x86, MIPS, RISC-V.

Generic Micro-Architecture

Very simple ones to a 7-stage out-of-order pipelined processor.

GEM5

- Micro-architectural simulator, **cycle-accurate**.
- **State-of-the-art** project, started in 2011.

Parts

- **C++ core** where logic is programmed.
- **Python interface** where systems are built.

Architecture

Alpha, **ARM**, Power, SPARC, x86, MIPS, RISC-V.

Generic Micro-Architecture

Very simple ones to a 7-stage out-of-order pipelined processor.

Branch Prediction

GEM5

- Micro-architectural simulator, **cycle-accurate**.
- **State-of-the-art** project, started in 2011.

Parts

- **C++ core** where logic is programmed.
- **Python interface** where systems are built.

Architecture

Alpha, **ARM**, Power, SPARC, x86, MIPS, RISC-V.

Generic Micro-Architecture

Very simple ones to a 7-stage out-of-order pipelined processor.

Branch Prediction

Bi-Mode, TAGE, Two-Level, Perceptron, Tournament…

HOW TO USE IT?

BUILDING A SIMULATED SYSTEM (SNIPPETS)

Configuring parameters of a cache memory

Instantiating some CPU components

Connecting components together

Passing arguments to the Linux kernel

BUILDING A SIMULATED SYSTEM (SNIPPETS)

Configuring parameters of a cache memory

```
size = '32kB'
assoc = 2
data_latency = 1
mshrs = 4
tgts_per_mshr = 8
write_buffers = 4
prefetcher = StridePrefetcher(queue_size=4, degree=4)
```

Instantiating some CPU components

Connecting components together

Passing arguments to the Linux kernel

BUILDING A SIMULATED SYSTEM (SNIPPETS)

Configuring parameters of a cache memory

```
size = '32kB'
assoc = 2
data_latency = 1
mshrs = 4
tgts_per_mshr = 8
write_buffers = 4
prefetcher = StridePrefetcher(queue_size=4, degree=4)
```

Instantiating some CPU components

```
for cpu in self.cpus:
    cpu.createThreads()
    cpu.createInterruptController()
    cpu.branchPredAdd()
if system.getMemoryMode() == "timing":
    self.cacheAddL1()
    self.cacheAddL2()
```

Connecting components together

Passing arguments to the Linux kernel

BUILDING A SIMULATED SYSTEM (SNIPPETS)

Configuring parameters of a cache memory

```
size = '32kB'
assoc = 2
data_latency = 1
mshrs = 4
tgts_per_mshr = 8
write_buffers = 4
prefetcher = StridePrefetcher(queue_size=4, degree=4)
```

Instantiating some CPU components

```
for cpu in self.cpus:
    cpu.createThreads()
    cpu.createInterruptController()
    cpu.branchPredAdd()
if system.getMemoryMode() == "timing":
    self.cacheAddL1()
    self.cacheAddL2()
```

Connecting components together

```
cpu.dtb.walker.port = bus.slave
cpu.itb.walker.port = bus.slave
cpu.dcache_port = bus.slave
cpu.icache_port = bus.slave
```

Passing arguments to the Linux kernel

BUILDING A SIMULATED SYSTEM (SNIPPETS)

Configuring parameters of a cache memory

```
size = '32kB'
assoc = 2
data_latency = 1
mshrs = 4
tgts_per_mshr = 8
write_buffers = 4
prefetcher = StridePrefetcher(queue_size=4, degree=4)
```

Instantiating some CPU components

```
for cpu in self.cpus:
    cpu.createThreads()
    cpu.createInterruptController()
    cpu.branchPredAdd()
if system.getMemoryMode() == "timing":
    self.cacheAddL1()
    self.cacheAddL2()
```

Connecting components together

```
cpu.dtb.walker.port = bus.slave
cpu.itb.walker.port = bus.slave
cpu.dcache_port = bus.slave
cpu.icache_port = bus.slave
```

Passing arguments to the Linux kernel

```
kernel_cmd = [
    "console=ttyAMA0",
    "root=/dev/vda1",
    "rw",
    "mem=2G@0x80000000",
]
```

LAUNCHING A SIMULATION

Launching a full-system simulation

Connecting to the simulation terminal

Kernel booting up

Opening a shell on the simulated system

LAUNCHING A SIMULATION

Launching a full-system simulation

```
./build/ARM/gem5.opt ./configs/example/arm/starter_fs.py --num-cores=4 --disk-image="aarch64-ubuntu.img" --kernel="vmlinux.arm64"
```

Connecting to the simulation terminal

Kernel booting up

Opening a shell on the simulated system

LAUNCHING A SIMULATION

Launching a full-system simulation

```
./build/ARM/gem5.opt ./configs/example/arm/starter_fs.py --num-cores=4 --disk-image="aarch64-ubuntu.img" --kernel="vmlinux.arm64"
```

Connecting to the simulation terminal

```
gem5/util/term/m5term localhost 3456
```

Kernel booting up

Opening a shell on the simulated system

LAUNCHING A SIMULATION

Launching a full-system simulation

```
./build/ARM/gem5.opt ./configs/example/arm/starter_fs.py --num-cores=4 --disk-image="aarch64-ubuntu.img" --kernel="vmlinux.arm64"
```

Connecting to the simulation terminal

```
gem5/util/term/m5term localhost 3456
```

Kernel booting up

```
==== m5 slave terminal: Terminal 0 ====
[    0.000000] Booting Linux on physical CPU 0x0000000000 [0x410fd070]
[    0.000000] Linux version 4.18.0+ (arm-employee@arm-computer) (gcc version 7.4.0 (Ubuntu/Linaro 7.4.0-1ubuntu1~18.04.1))
[    0.000000] Machine model: V2P-CA15
[    0.000000] Memory limited to 2048MB
...
```

Opening a shell on the simulated system

LAUNCHING A SIMULATION

Launching a full-system simulation

```
./build/ARM/gem5.opt ./configs/example/arm/starter_fs.py --num-cores=4 --disk-image="aarch64-ubuntu.img" --kernel="vmlinux.arm64"
```

Connecting to the simulation terminal

```
gem5/util/term/m5term localhost 3456
```

Kernel booting up

```
==== m5 slave terminal: Terminal 0 ====
[    0.000000] Booting Linux on physical CPU 0x0000000000 [0x410fd070]
[    0.000000] Linux version 4.18.0+ (arm-employee@arm-computer) (gcc version 7.4.0 (Ubuntu/Linaro 7.4.0-1ubuntu1~18.04.1))
[    0.000000] Machine model: V2P-CA15
[    0.000000] Memory limited to 2048MB
...
```

Opening a shell on the simulated system

```
...
[    0.256634] random: init: uninitialized urandom read (12 bytes read)
[    0.271877] init: hwclock main process (684) terminated with status 1
[    0.286689] random: mountall: uninitialized urandom read (12 bytes read)

Ubuntu 14.04 LTS aarch64-gem5 ttyAMA0

aarch64-gem5 login: root

Welcome to Ubuntu 14.04 LTS (GNU/Linux 4.18.0+ aarch64)

root@aarch64-gem5:~#
```

BENEFITS OF (PIPELINE) VISUALIZATION

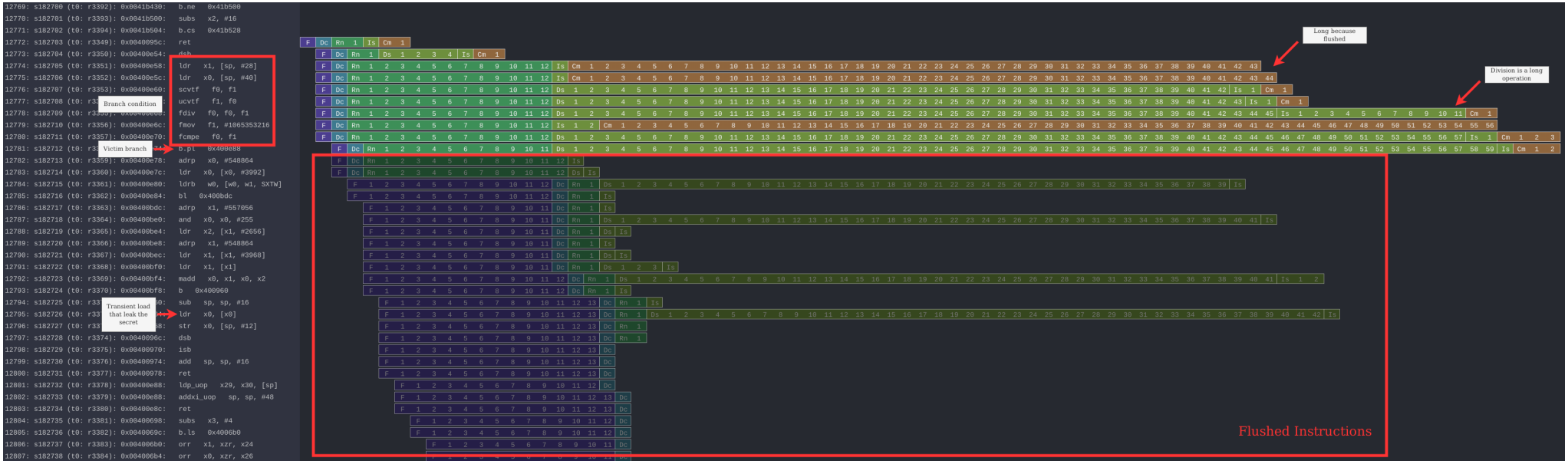
BENEFITS OF (PIPELINE) VISUALIZATION

- gem5 $\Rightarrow$ **output the state of any element** in the system.

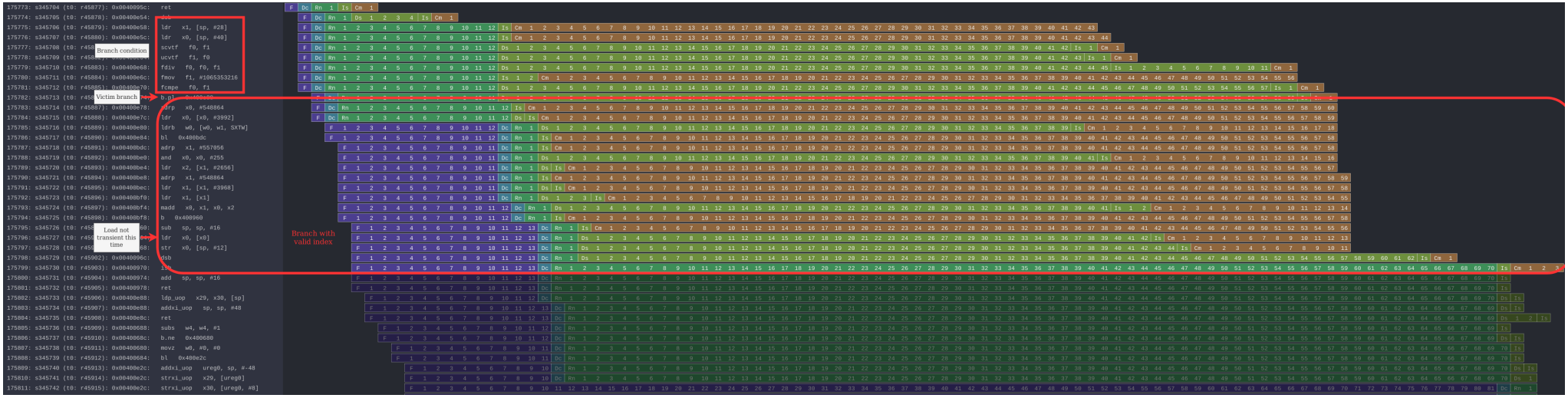
BENEFITS OF (PIPELINE) VISUALIZATION

- gem5 $\Rightarrow$ **output the state of any element** in the system.
- Konata $\Rightarrow$ **graphically visualize the pipeline** of a simulated processor.

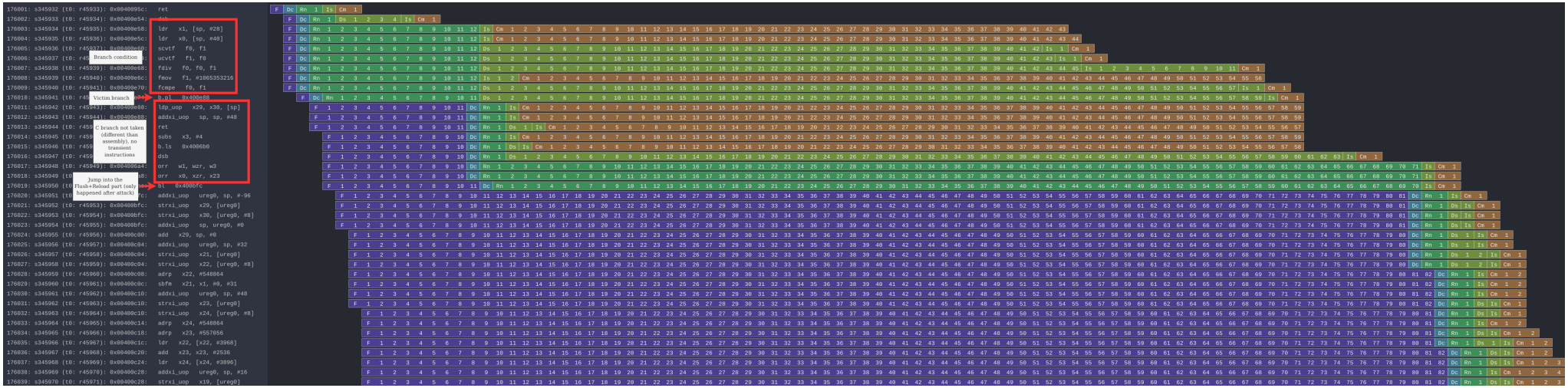
TRANSIENT EXECUTION OF A READ INSTRUCTION WITH A MALICIOUS INDEX



BRANCH PREDICTOR BEING TRAINED



SPECTRE DEFEATED BY THE BRANCH PREDICTOR



IMPLEMENTING...

...THE SPECTRE ATTACK ON ARM

...THE SPECTRE ATTACK ON ARM

- IAIK implementation **failed to perform** the attack **successfully**.

...THE SPECTRE ATTACK ON ARM

- IAIK implementation **failed to perform** the attack **successfully**.
- Needed an **implementation** with the following requirements:

...THE SPECTRE ATTACK ON ARM

- IAIK implementation **failed to perform** the attack **successfully**.
- Needed an **implementation** with the following requirements:
    1. **Stable** results,

...THE SPECTRE ATTACK ON ARM

- IAIK implementation **failed to perform** the attack **successfully**.
- Needed an **implementation** with the following requirements:
    1. **Stable** results,
    2. Follows our **guidelines**,

...THE SPECTRE ATTACK ON ARM

- IAIK implementation **failed to perform** the attack **successfully**.
- Needed an **implementation** with the following requirements:
    1. **Stable** results,
    2. Follows our **guidelines**,
    3. Usable both on the **Raspberry Pi** and on **gem5**,

...THE SPECTRE ATTACK ON ARM

- IAIK implementation **failed to perform** the attack **successfully**.
- Needed an **implementation** with the following requirements:
    1. **Stable** results,
    2. Follows our **guidelines**,
    3. Usable both on the **Raspberry Pi** and on **gem5**,
    4. **Metrics** output.

...THE SPECTRE ATTACK ON ARM

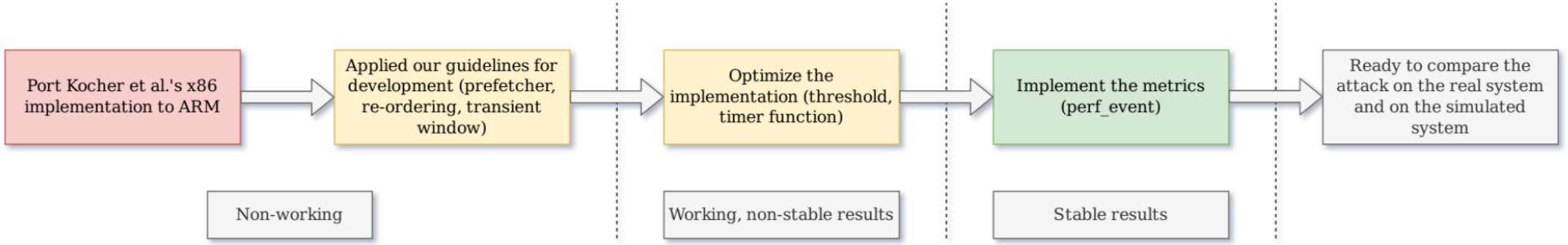
- IAIK implementation **failed to perform** the attack **successfully**.
- Needed an **implementation** with the following requirements:
    1. **Stable** results,
    2. Follows our **guidelines**,
    3. Usable both on the **Raspberry Pi** and on **gem5**,
    4. **Metrics** output.

Steps

...THE SPECTRE ATTACK ON ARM

- IAIK implementation **failed to perform** the attack **successfully**.
- Needed an **implementation** with the following requirements:
  1. **Stable** results,
  2. Follows our **guidelines**,
  3. Usable both on the **Raspberry Pi** and on **gem5**,
  4. **Metrics** output.

Steps



...AN ARM GEM5 SYSTEM

…AN ARM GEM5 SYSTEM

- Steps:

...AN ARM GEM5 SYSTEM

- Steps:
    1. **Syscall** emulation system

...AN ARM GEM5 SYSTEM

- Steps:
    1. **Syscall** emulation system
    2. **Caches**

...AN ARM GEM5 SYSTEM

- Steps:
    1. **Syscall** emulation system
    2. **Caches**
    3. **Branch predictor** $\Rightarrow$ **Spectre** working

...AN ARM GEM5 SYSTEM

- Steps:
    1. **Syscall** emulation system
    2. **Caches**
    3. **Branch predictor** $\Rightarrow$ **Spectre** working
    4. **Full-system** simulation

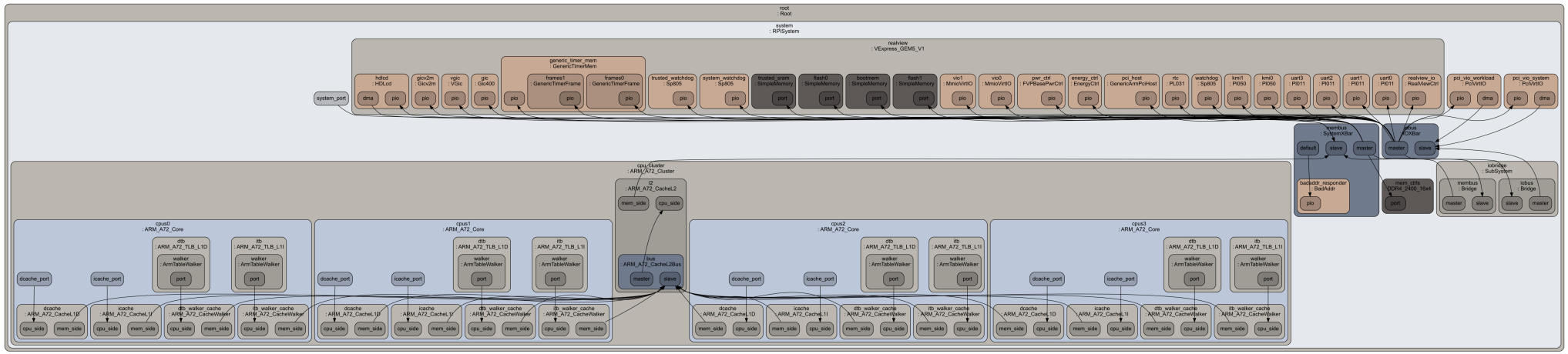
...AN ARM GEM5 SYSTEM

- Steps:
  1. **Syscall** emulation system
  2. **Caches**
  3. **Branch predictor** $\Rightarrow$ **Spectre** working
  4. **Full-system** simulation
  5. **Patch** for `perf_event` $\Rightarrow$ **measurements** working

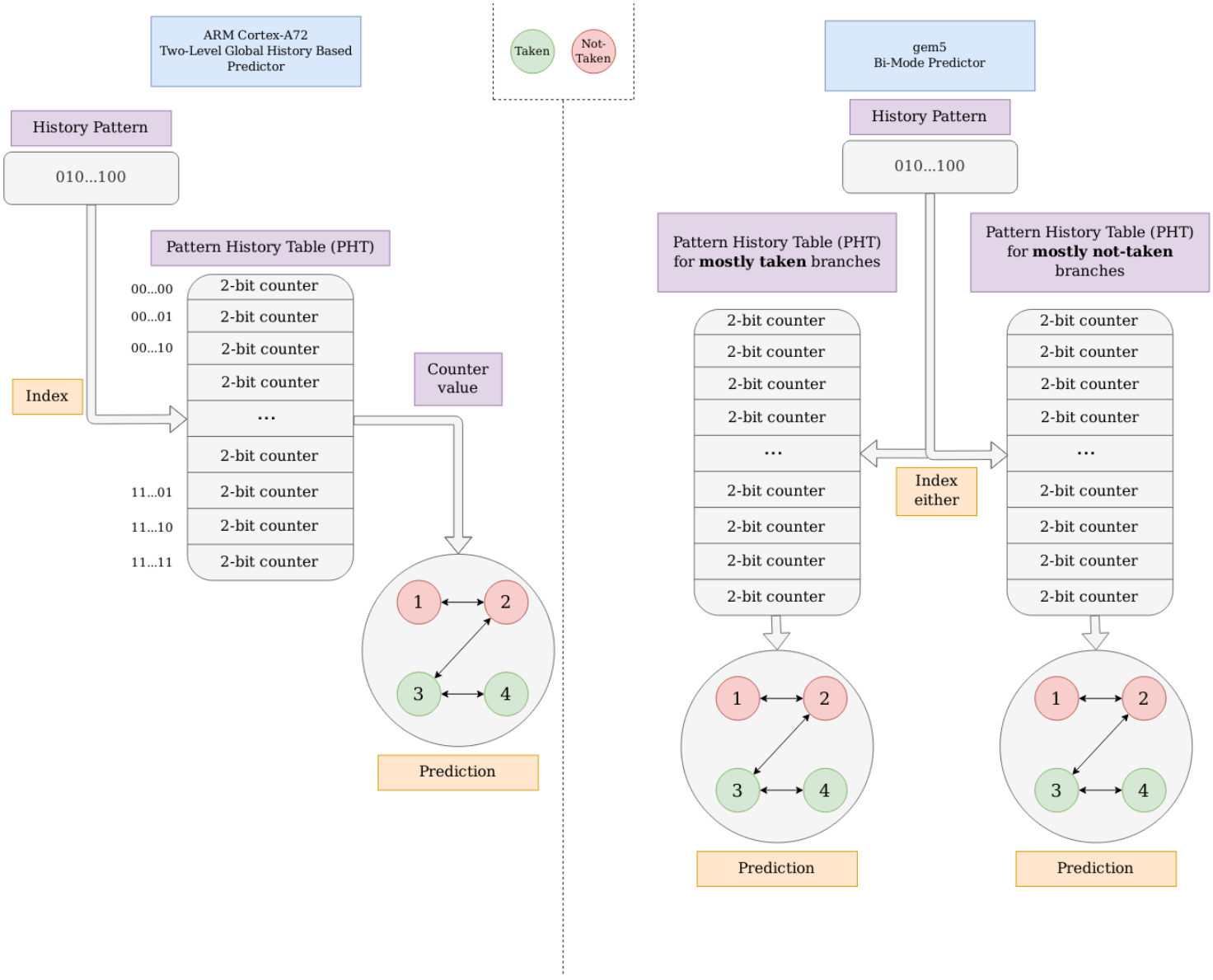
...AN ARM GEM5 SYSTEM

- Steps:
  1. **Syscall** emulation system
  2. **Caches**
  3. **Branch predictor** $\Rightarrow$ **Spectre** working
  4. **Full-system** simulation
  5. **Patch** for `perf_event` $\Rightarrow$ **measurements** working
- Getting **closer** of the **ARM Cortex-A72** of the Raspberry Pi.

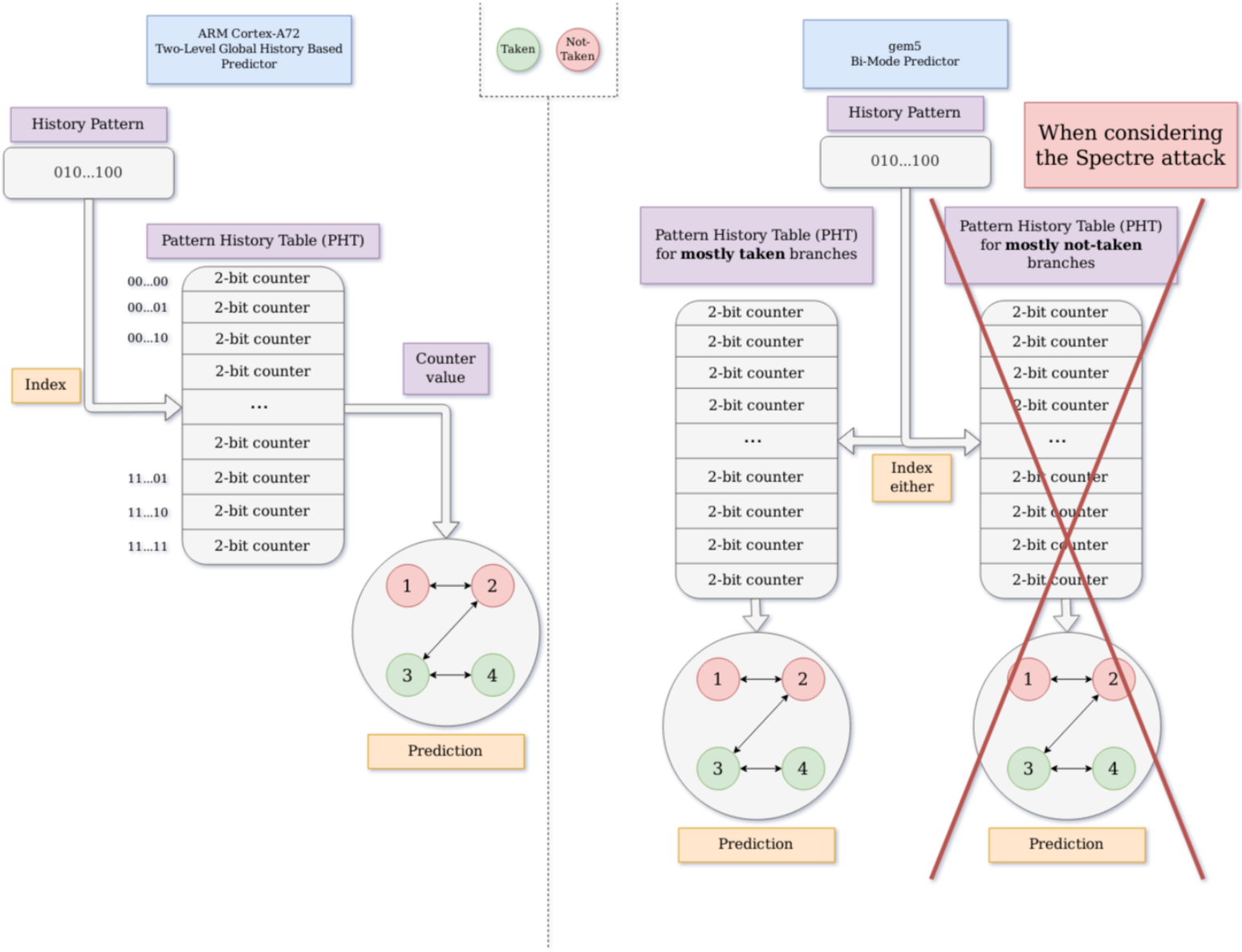
DETAILS OF THE GEM5 SYSTEM



CONFIGURATION OF THE BRANCH PREDICTOR



CONSIDERING SPECTRE, BOTH PREDICTORS ARE EQUIVALENT



IS THE SIMULATION FAITHFUL?

IS THE SIMULATION FAITHFUL?

- A simulation **not faithful** will not be so **useful**…

IS THE SIMULATION FAITHFUL?

- A simulation **not faithful** will not be so **useful**…
- Measurements of **metrics**:

IS THE SIMULATION FAITHFUL?

- A simulation **not faithful** will not be so **useful**…
- Measurements of **metrics**:
    1. **Retrieved bytes**: Similar

IS THE SIMULATION FAITHFUL?

- A simulation **not faithful** will not be so **useful**…
- Measurements of **metrics**:
    1. **Retrieved bytes**: Similar
    2. **Iterations**: Two times easier on gem5

IS THE SIMULATION FAITHFUL?

- A simulation **not faithful** will not be so **useful**…
- Measurements of **metrics**:
    1. **Retrieved bytes**: Similar
    2. **Iterations**: Two times easier on gem5
    3. **Cycles**: Three time faster on gem5

IS THE SIMULATION FAITHFUL?

- A simulation **not faithful** will not be so **useful**…
- Measurements of **metrics**:
    1. **Retrieved bytes**: Similar
    2. **Iterations**: Two times easier on gem5
    3. **Cycles**: Three time faster on gem5
    4. **Cache misses**: Aberrant result

IS THE SIMULATION FAITHFUL?

- A simulation **not faithful** will not be so **useful**…
- Measurements of **metrics**:
    1. **Retrieved bytes**: Similar
    2. **Iterations**: Two times easier on gem5
    3. **Cycles**: Three time faster on gem5
    4. **Cache misses**: Aberrant result
    5. **Mispredicted branches**: Similar

CONCLUSION

CONCLUSION

- If simulation becomes widely used:

CONCLUSION

- If simulation becomes widely used:
  - Easier to reproduce older attacks for understanding and experimentation.

CONCLUSION

- If simulation becomes widely used:
  - Easier to reproduce older attacks for understanding and experimentation.
  - With faithful models, researchers could use the simulator itself to discover new vulnerabilities.

CONCLUSION

- If simulation becomes widely used:
  - Easier to reproduce older attacks for understanding and experimentation.
  - With faithful models, researchers could use the simulator itself to discover new vulnerabilities.
- But…

CONCLUSION

- If simulation becomes widely used:
  - Easier to reproduce older attacks for understanding and experimentation.
  - With faithful models, researchers could use the simulator itself to discover new vulnerabilities.
- But…
  - Simulation is currently slow.

CONCLUSION

- If simulation becomes widely used:
  - Easier to reproduce older attacks for understanding and experimentation.
  - With faithful models, researchers could use the simulator itself to discover new vulnerabilities.
- But…
  - Simulation is currently slow.
  - Simulator still needs improvements and extensions.

CONCLUSION

- If simulation becomes widely used:
  - Easier to reproduce older attacks for understanding and experimentation.
  - With faithful models, researchers could use the simulator itself to discover new vulnerabilities.
- But…
  - Simulation is currently slow.
  - Simulator still needs improvements and extensions.
- In summary:

CONCLUSION

- If simulation becomes widely used:
  - Easier to reproduce older attacks for understanding and experimentation.
  - With faithful models, researchers could use the simulator itself to discover new vulnerabilities.
- But…
  - Simulation is currently slow.
  - Simulator still needs improvements and extensions.
- In summary:
  - Possible to simulate micro-architectural attacks and being accurate.

CONCLUSION

- If simulation becomes widely used:
  - Easier to reproduce older attacks for understanding and experimentation.
  - With faithful models, researchers could use the simulator itself to discover new vulnerabilities.
- But…
  - Simulation is currently slow.
  - Simulator still needs improvements and extensions.
- In summary:
  - Possible to simulate micro-architectural attacks and being accurate.
  - Visualization is a very powerful technique to understand the micro-architectural behavior.

WEBSITE

https://pierreay.github.io/reproduce-spectre-gem5/

QUESTIONS?

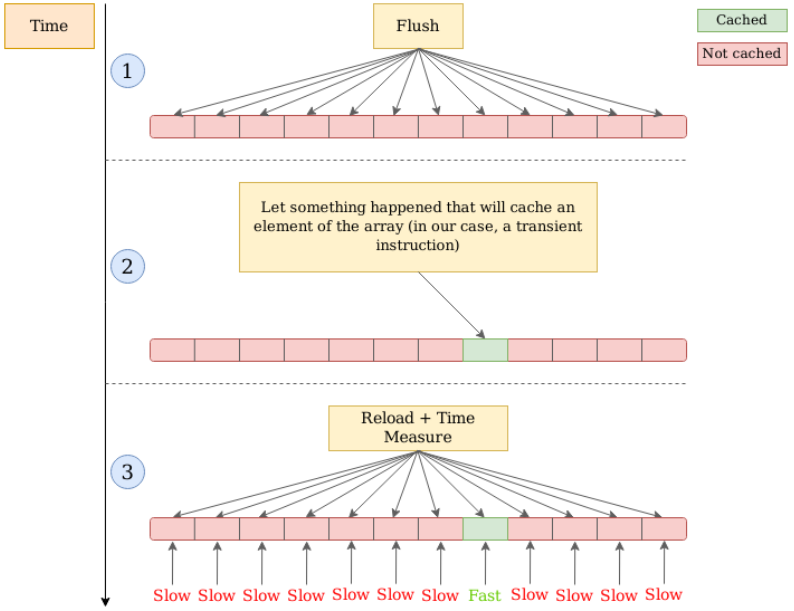
pierre.ayoub@eurecom.fr

APPENDICES

THE MICROARCHITECTURAL DOMAIN

Flush+Reload

Often used to probe the cache state.



SIMULATION MODES

SIMULATION MODES

Syscall Emulation

SIMULATION MODES

Syscall Emulation

gem5 plays the role of the operating system, as it emulates every system calls of a binary over the simulated hardware.

SIMULATION MODES

Syscall Emulation

gem5 plays the role of the operating system, as it emulates every system calls of a binary over the simulated hardware.

Full-System Simulation

SIMULATION MODES

Syscall Emulation

gem5 plays the role of the operating system, as it emulates every system calls of a binary over the simulated hardware.

Full-System Simulation

gem5 runs an entire operating system over the simulated hardware.

SIMULATION MODES

Syscall Emulation

gem5 plays the role of the operating system, as it emulates every system calls of a binary over the simulated hardware.

Full-System Simulation

gem5 runs an entire operating system over the simulated hardware.

Baremetal

SIMULATION MODES

Syscall Emulation

gem5 plays the role of the operating system, as it emulates every system calls of a binary over the simulated hardware.

Full-System Simulation

gem5 runs an entire operating system over the simulated hardware.

Baremetal

gem5 runs native assembly code over the simulated hardware, without any operating system layer.

RESULTS

Table 1: Ratio between gem5 and Raspberry Pi runs for each metric. A value below 1 means that gem5's metric is lower than the Raspberry Pi's metric.

	Accuracy Ratio	Accuracy Ratio
Metric	**Mean**	**Standard Deviation**
Retrieved Bytes	1.05	NaN
Iterations	0.57	3.81
Cycles	0.31	2.12
Cache Misses	584.08	4581.02
Mispredicted Branches	0.99	2.41