



black hat[®]
USA 2024

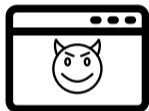
AUGUST 7-8, 2024
BRIEFINGS

Arbitrary Data Manipulation and Leakage with CPU Zero-Day Bugs on RISC-V

Fabian Thomas, Lorenz Hetterich



The Challenge: Operating System



Application



Hardware

The Challenge: Operating System

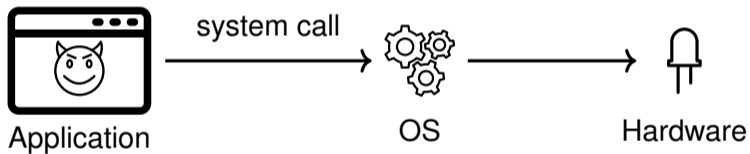


Application

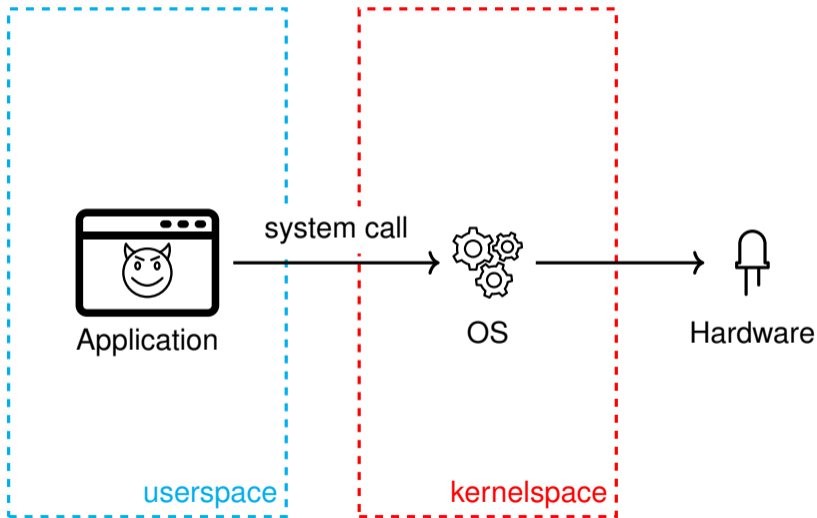


Hardware

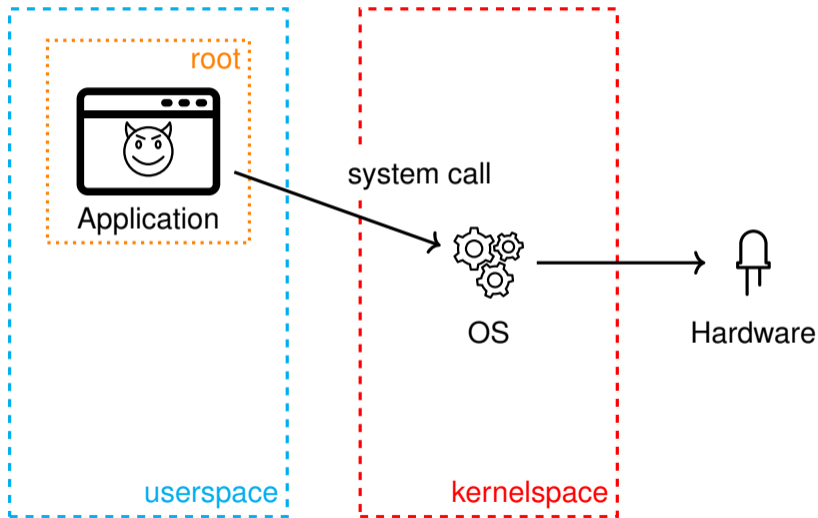
The Challenge: Operating System



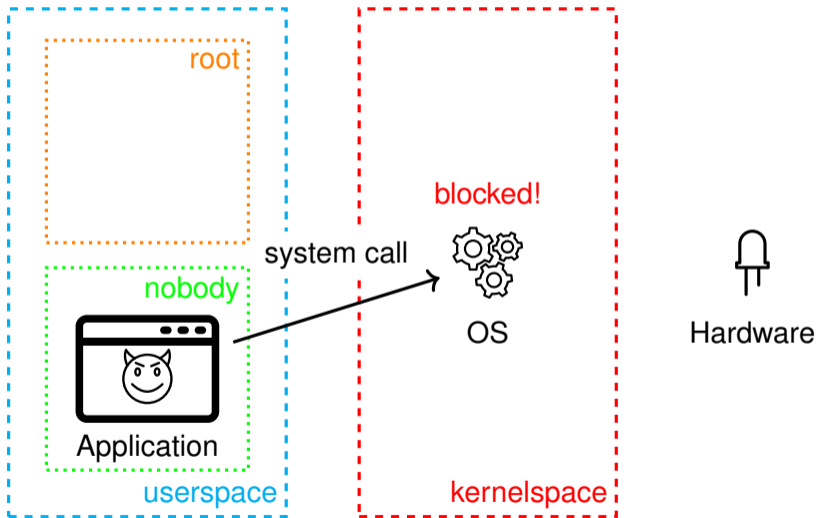
The Challenge: Operating System



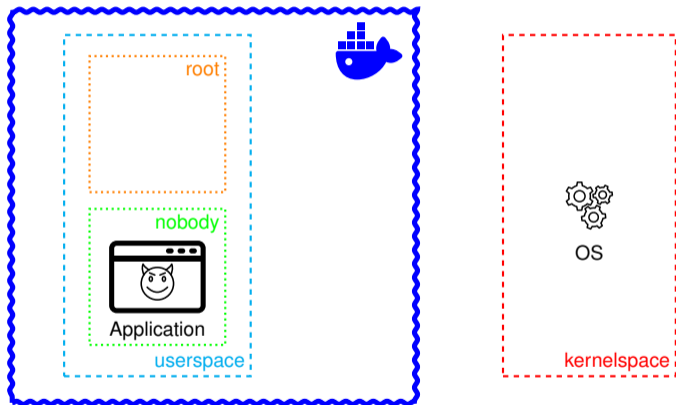
The Challenge: Sandboxing



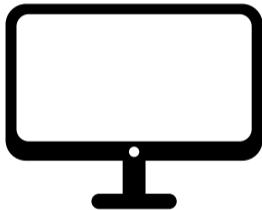
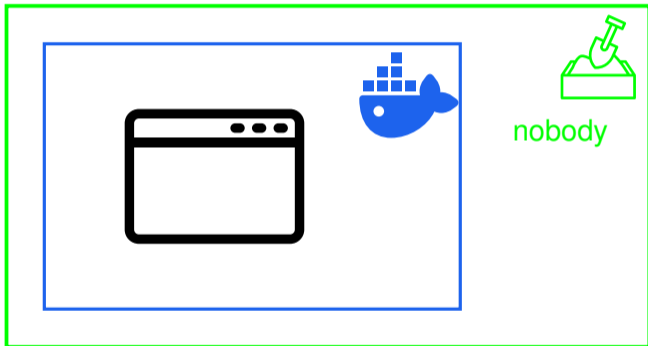
The Challenge: Sandboxing



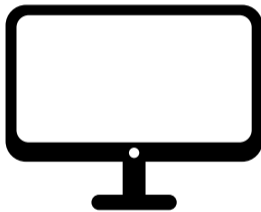
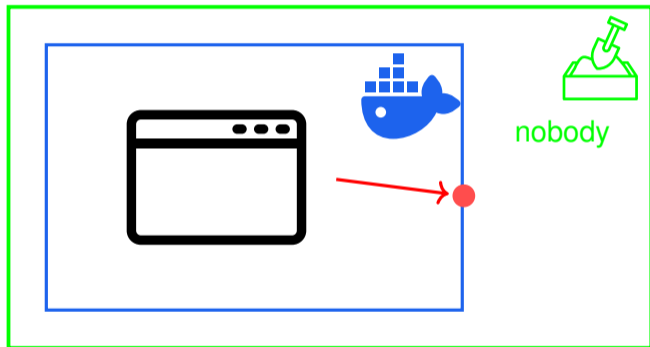
The Challenge: Container



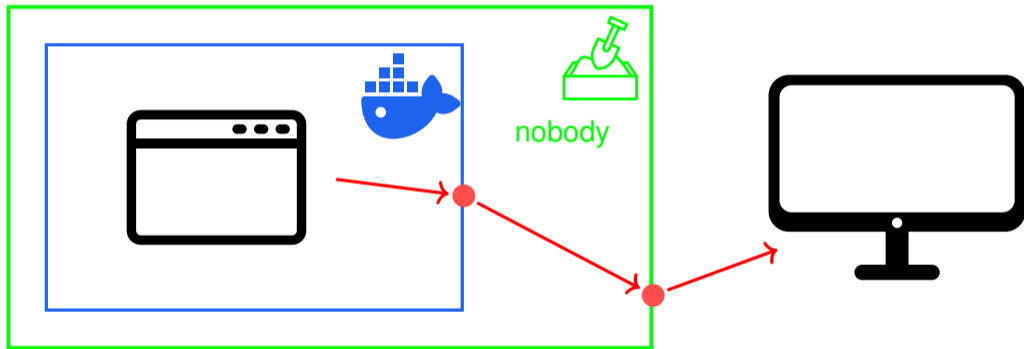
The Challenge: Full Isolation



The Challenge: Full Isolation



The Challenge: Full Isolation





~~You can't break out...~~: GhostWrite



Fabian Thomas

PhD student @CISPA (Germany)

E-Mail fabian.thomas@cispa.de

Web fabianthomas.de

Twitter [@fth0mas](https://twitter.com/fth0mas)



Lorenz Hetterich

PhD student @CISPA (Germany)

E-Mail lorenz.hetterich@cispa.de

Twitter [@hetterichlorenz](https://twitter.com/hetterichlorenz)



CISPA

HELMHOLTZ CENTER FOR
INFORMATION SECURITY

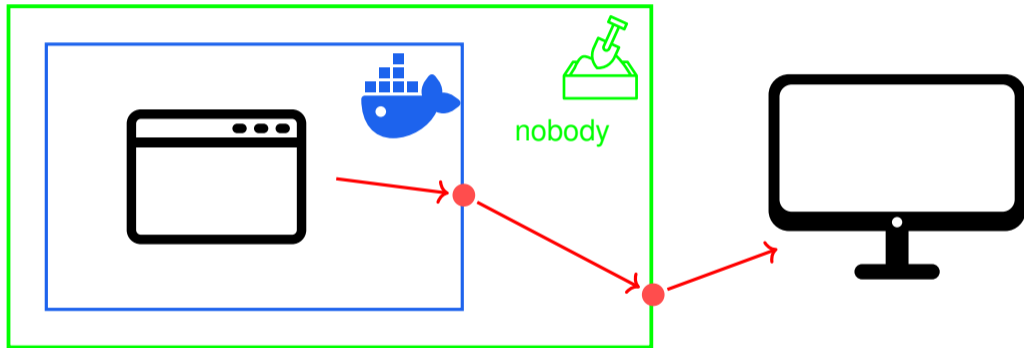


Research Group Schwarz

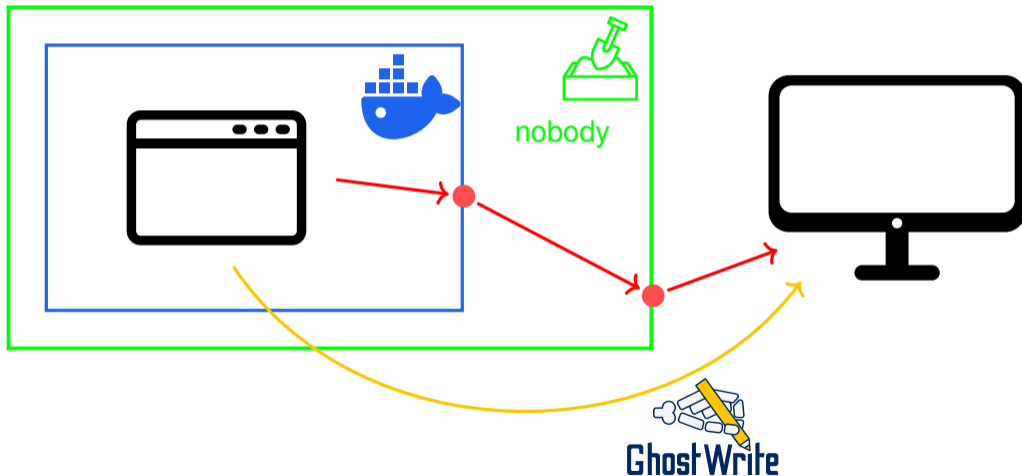
- Research focus:
 - Hardware vulnerabilities
 - ... from software
- Recent discoveries:



Full Isolation: GhostWrite



Full Isolation: GhostWrite



Software World

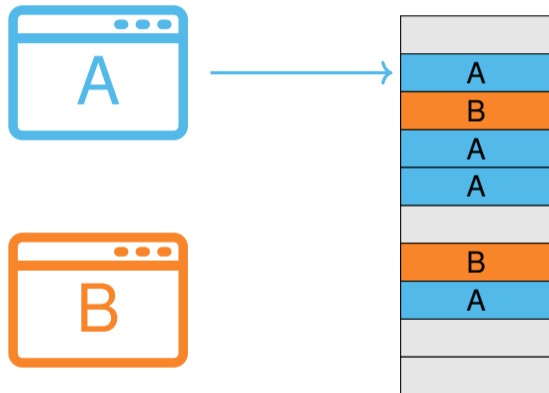


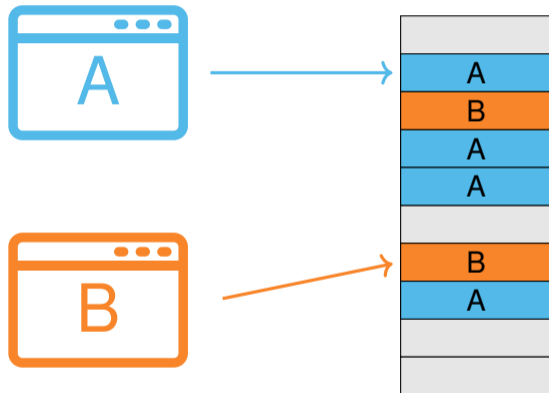
Software World

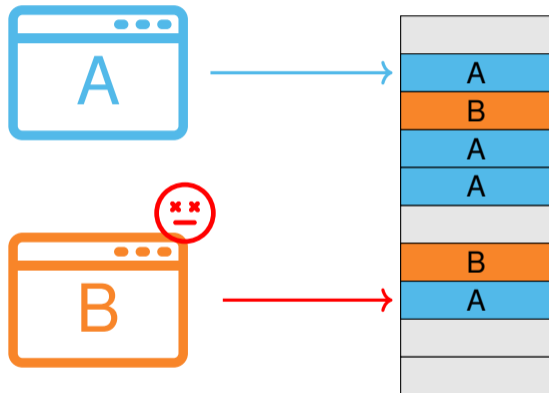


Hardware World

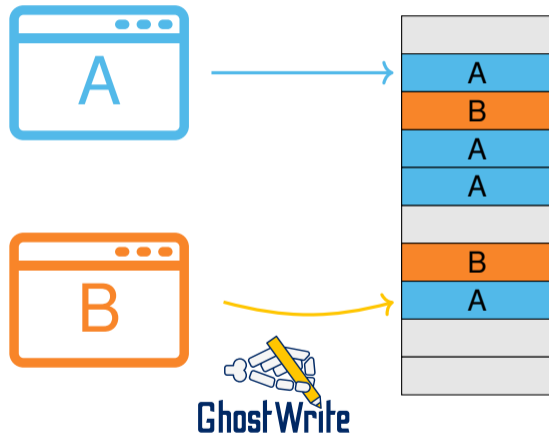




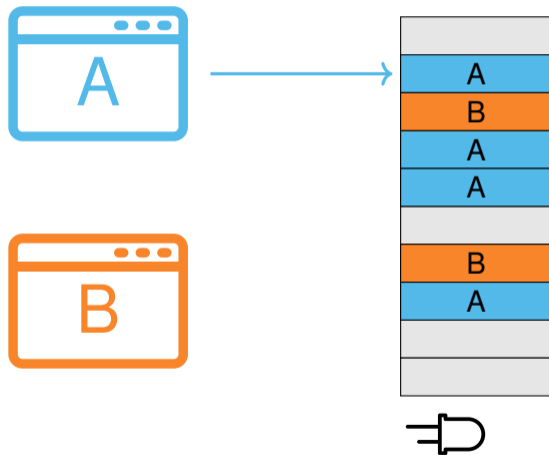




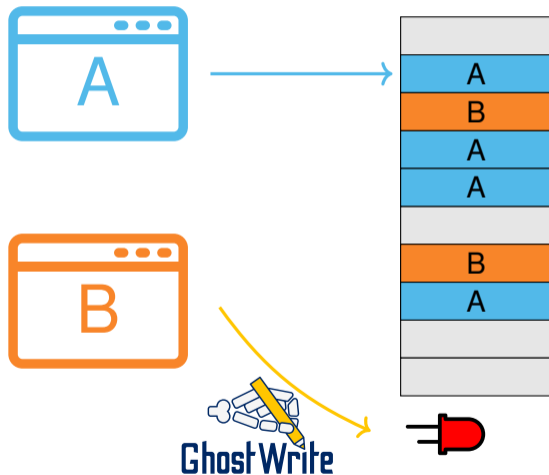
Memory Isolation: GhostWrite



Memory Isolation: GhostWrite



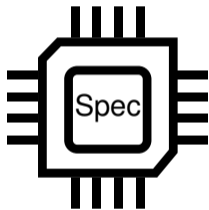
Memory Isolation: GhostWrite



```
mv t0, phys_addr  
vmv.v.x v0, value  
vsetvli zero, zero, e8, m1  
vse128.v v0, 0(t0)
```

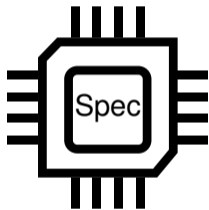
```
mv t0, phys_addr  
vmv.v.x v0, value  
vsetvli zero, zero, e8, m1  
vse128.v v0, 0(t0)
```

```
vse128.v v0, 0(t0)
```

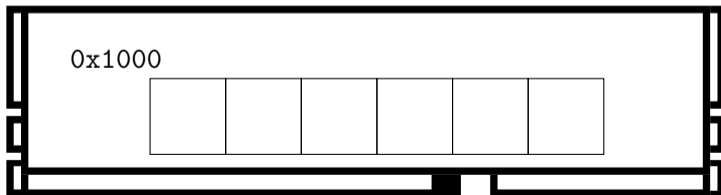


t0	v0	v1	v2	v3	v4	v5
0x1000	R	I	S	C	-	V

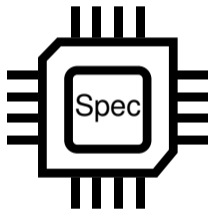
```
vse128.v v0, 0(t0)
```



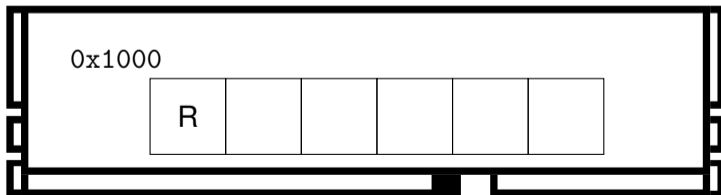
t0	v0	v1	v2	v3	v4	v5
0x1000	R	I	S	C	-	V



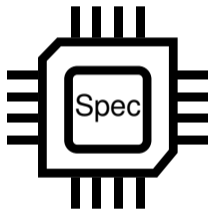
vse128.v v0, 0(t0)



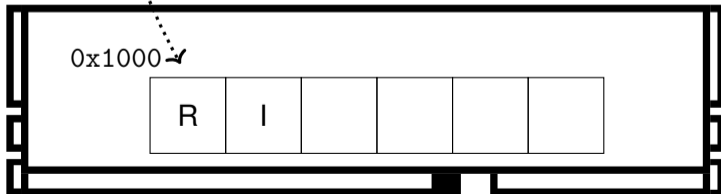
t0	v0	v1	v2	v3	v4	v5
0x1000	R	I	S	C	-	V



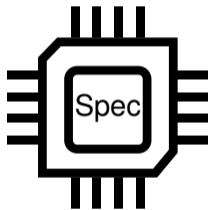

```
vse128.v v0, 0(t0)
```



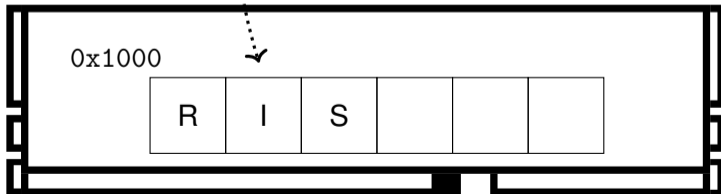
t0	v0	v1	v2	v3	v4	v5
0x1000	R	I	S	C	-	V



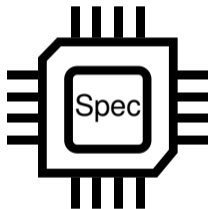
```
vse128.v v0, 0(t0)
```



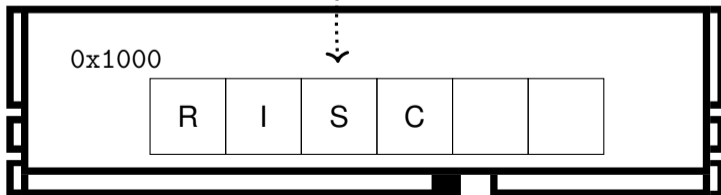
t0	v0	v1	v2	v3	v4	v5
0x1000	R	I	S	C	-	V



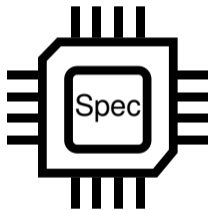
```
vse128.v v0, 0(t0)
```



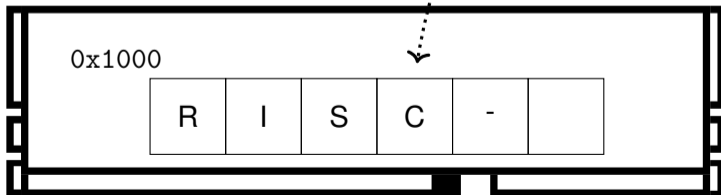
t0	v0	v1	v2	v3	v4	v5
0x1000	R	I	S	C	-	V



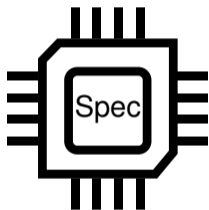
```
vse128.v v0, 0(t0)
```



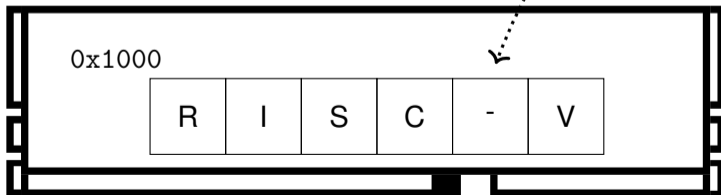
t0	v0	v1	v2	v3	v4	v5
0x1000	R	I	S	C	-	V



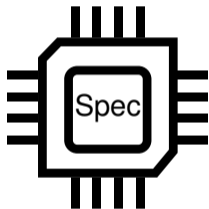
```
vse128.v v0, 0(t0)
```



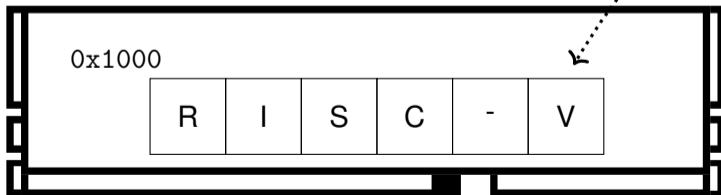
t0	v0	v1	v2	v3	v4	v5
0x1000	R	I	S	C	-	V



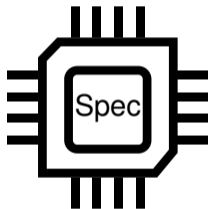
```
vse128.v v0, 0(t0)
```



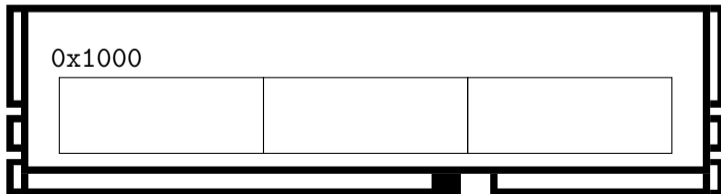
t0	v0	v1	v2	v3	v4	v5
0x1000	R	I	S	C	-	V



```
vse128.v v0, 0(t0)
```

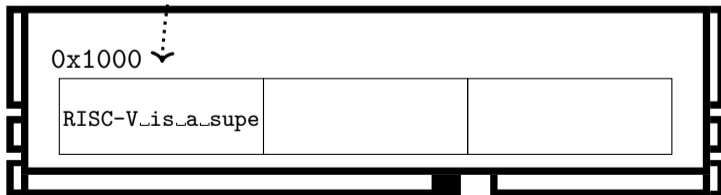
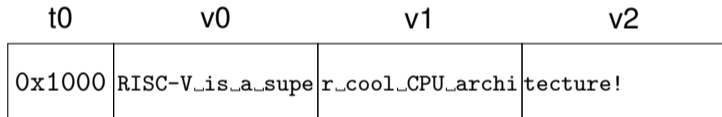
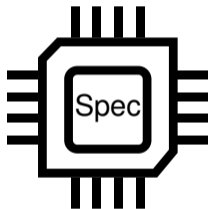


t0	v0	v1	v2
0x1000	RISC-V_is_a_super	r_cool_CPU_archi	ecture!

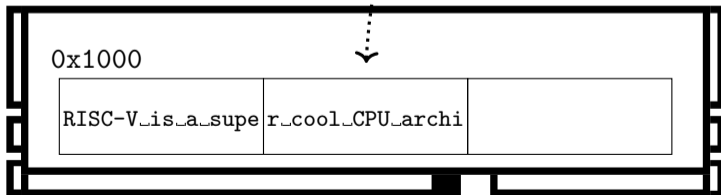
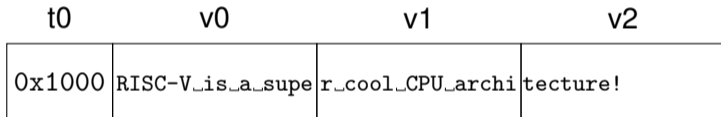
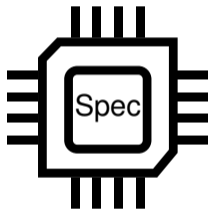


GhostWrite: Vector Instructions

```
vse128.v v0, 0(t0)
```

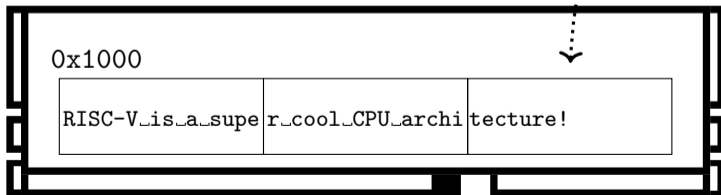
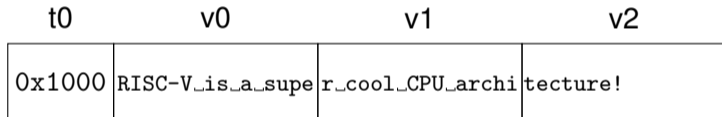
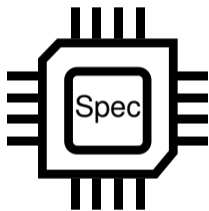


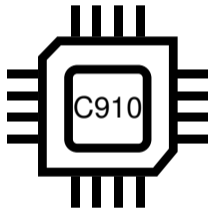

```
vse128.v v0, 0(t0)
```



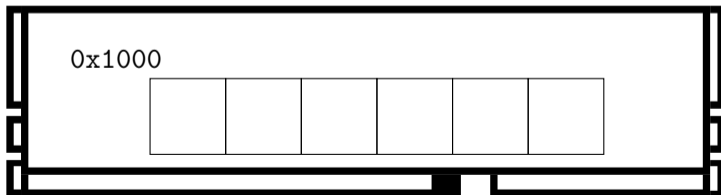
GhostWrite: Vector Instructions

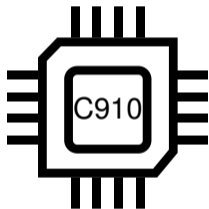
```
vse128.v v0, 0(t0)
```



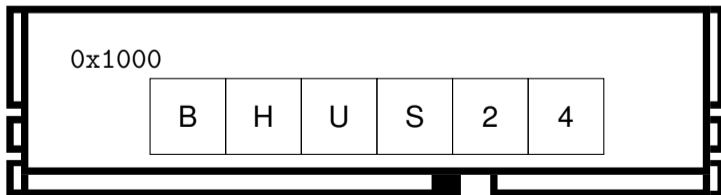


t0	v0	v1	v2	v3	v4	v5
0x1000	R	I	S	C	-	V

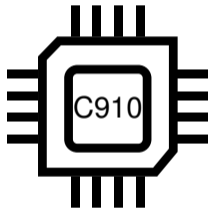




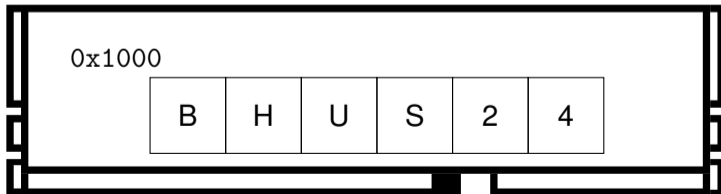
t0	v0	v1	v2	v3	v4	v5
0x1000	R	I	S	C	-	V



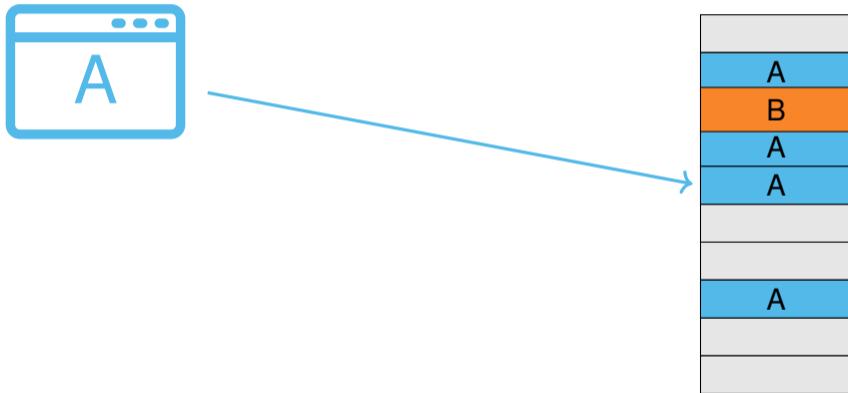
```
vse128.v v0, 0(t0)
```

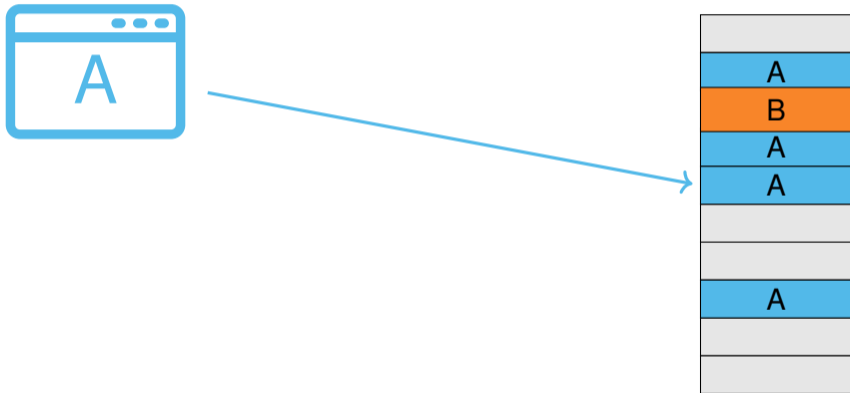


t0	v0	v1	v2	v3	v4	v5
0x1000	R	I	S	C	-	V



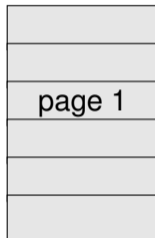
GhostWrite: Virtual Memory



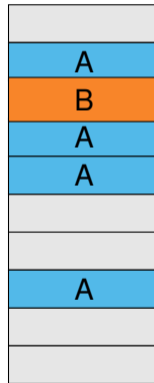


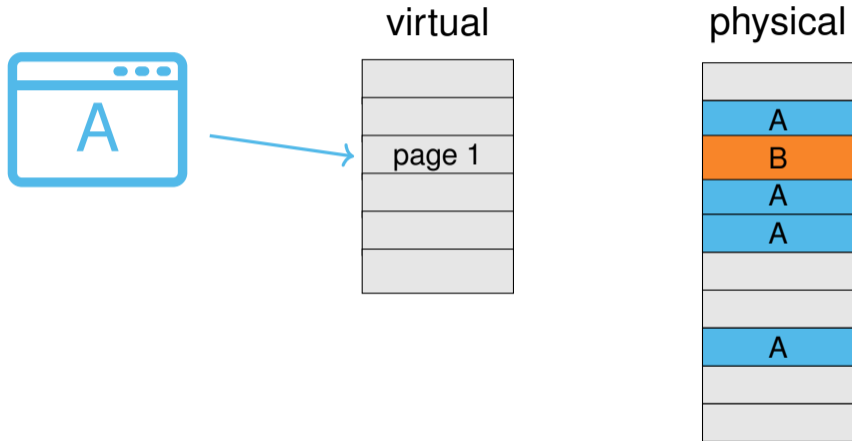


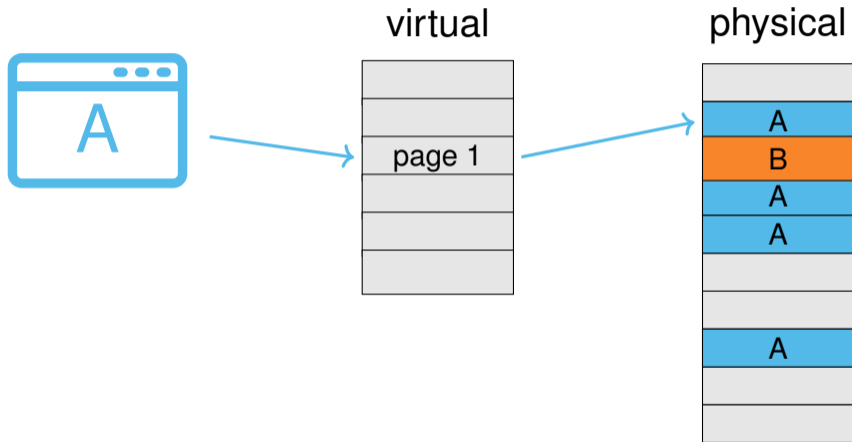
virtual



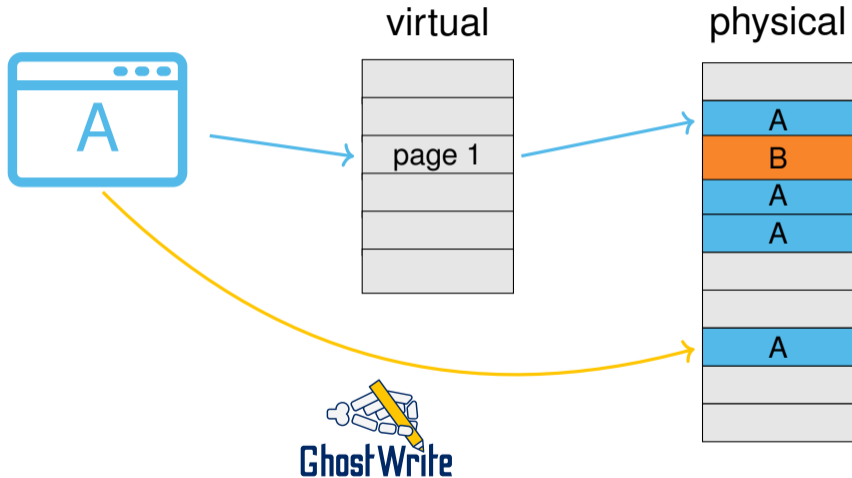
physical

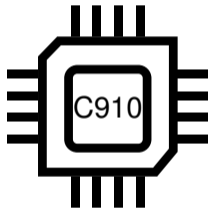




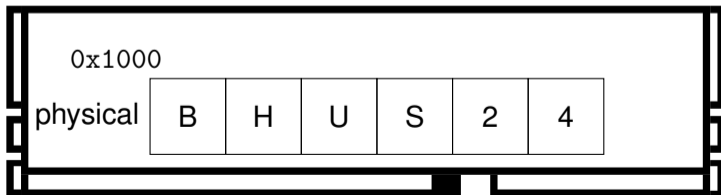


GhostWrite: Virtual Memory

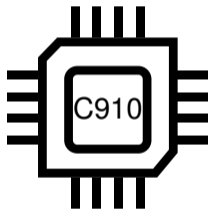




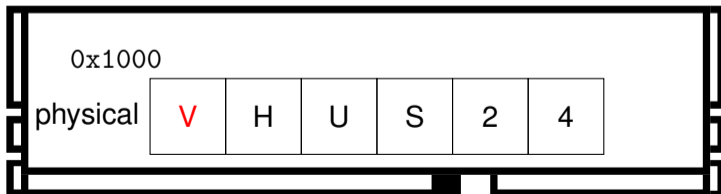
t0	v0	v1	v2	v3	v4	v5
0x1000	R	I	S	C	-	V



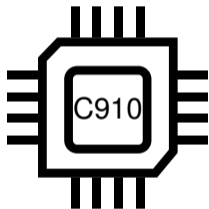
```
vse128.v v0, 0(t0)
```



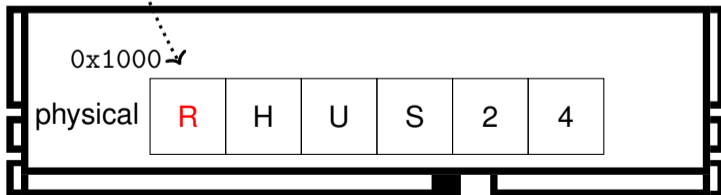
t0	v0	v1	v2	v3	v4	v5
0x1000	R	I	S	C	-	V



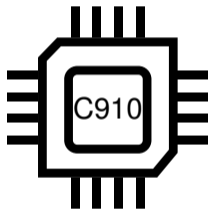
```
vse128.v v0, 0(t0)
```



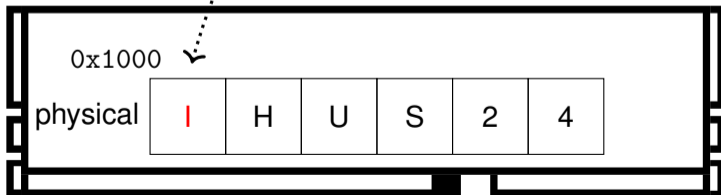
t0	v0	v1	v2	v3	v4	v5
0x1000	R	I	S	C	-	V



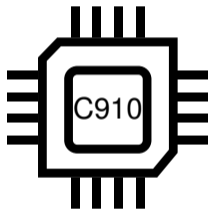
```
vse128.v v0, 0(t0)
```



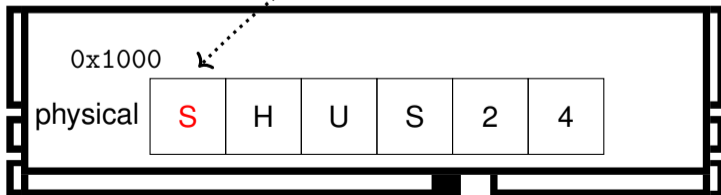
t0	v0	v1	v2	v3	v4	v5
0x1000	R	I	S	C	-	V



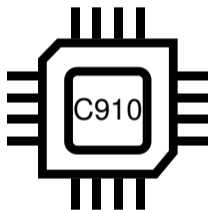
```
vse128.v v0, 0(t0)
```



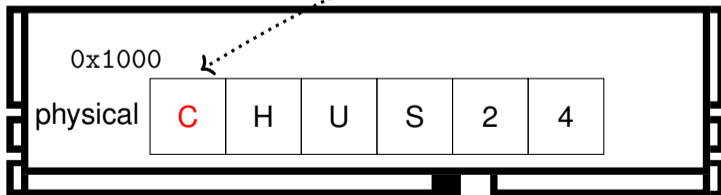
t0	v0	v1	v2	v3	v4	v5
0x1000	R	I	S	C	-	V



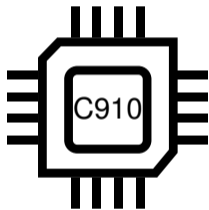

```
vse128.v v0, 0(t0)
```



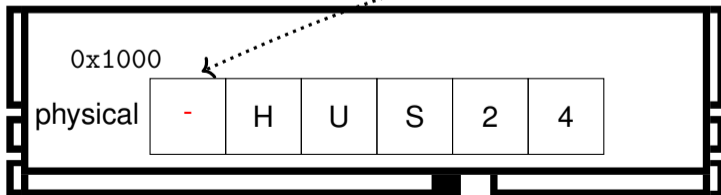
t0	v0	v1	v2	v3	v4	v5
0x1000	R	I	S	C	-	V



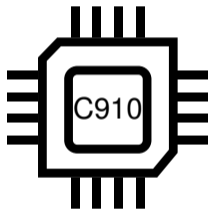
```
vse128.v v0, 0(t0)
```



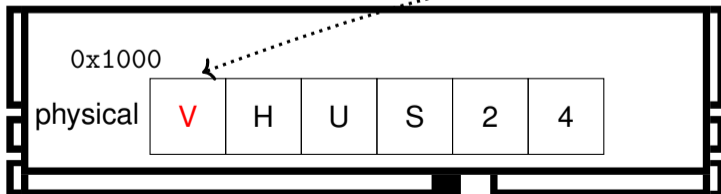
t0	v0	v1	v2	v3	v4	v5
0x1000	R	I	S	C	-	V



```
vse128.v v0, 0(t0)
```



t0	v0	v1	v2	v3	v4	v5
0x1000	R	I	S	C	-	V



Is every system vulnerable?



Is every system vulnerable?

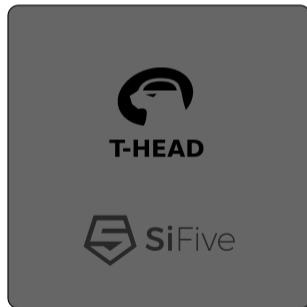
x86



arm



RISC-V®



Is every system vulnerable?

x86



arm



RISC-V®





- one of the fastest RISC-V CPUs



- one of the fastest RISC-V CPUs
- 4 cores, 2GHz, vector extension



- one of the fastest RISC-V CPUs
- 4 cores, 2GHz, vector extension
- available in the cloud





- one of the fastest RISC-V CPUs
- 4 cores, 2GHz, vector extension
- available in the cloud
- available in laptops





Specifies behavior

Instruction Set Architecture (ISA)



Specifies behavior



Defines legal programs

Instruction Set Architecture (ISA)



Specifies behavior



Defines legal programs



Licensing fees

Instruction Set Architecture (ISA)



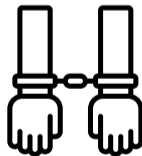
Specifies behavior



Defines legal programs



Licensing fees



Limited customization



open, community-driven



open, community-driven



no licensing fees



open, community-driven



no licensing fees



well designed



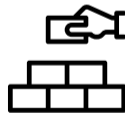
open, community-driven



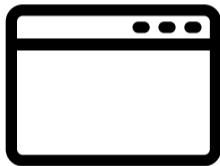
no licensing fees



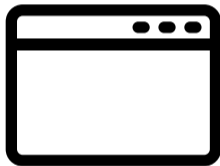
well designed



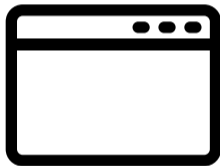
extensible



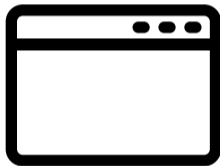
01011010011110100101 →



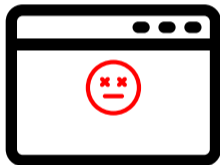
01001010010101010101 →

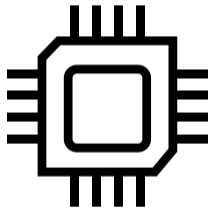


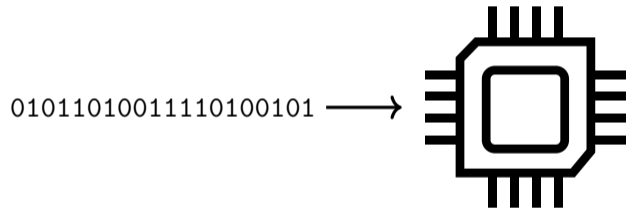
10101010010101101111 →



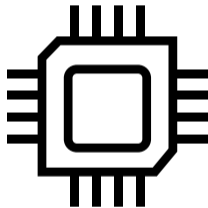
00000001101010100101 →



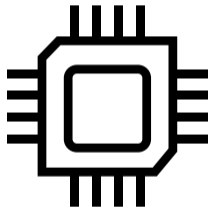




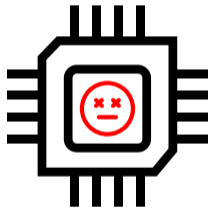
01001010010101010101 →



10101010010101101111 →



00000001101010100101 →





01011010011110100101



01011010011110100101



a=3

b=7



a=3

b=7

01001010010101010101



a=21

b=2



a=21

b=2

10101010010101101111



a=34

b=9



a=34

b=9

00000001101010100101



a=42

b=5



a=142

b=5

```
fadd f3, f4, f4  
li x1, 42  
  
x1: 0
```





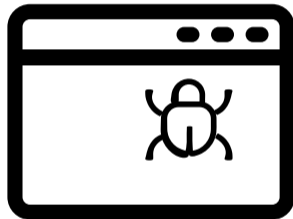
推进RISC-V架构创新

领导基金会数据中心、存储管理、安全等11个技术方向

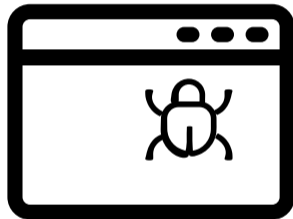


Demo: Freezing the C906

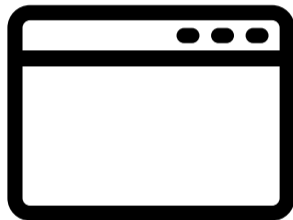
How to fix Hardware Bugs?



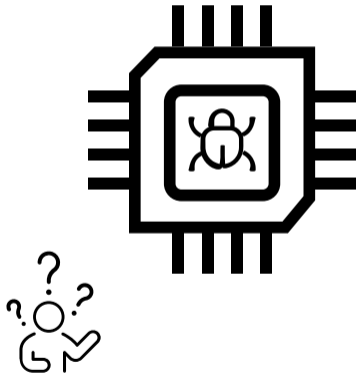
How to fix Hardware Bugs?



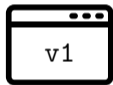
How to fix Hardware Bugs?



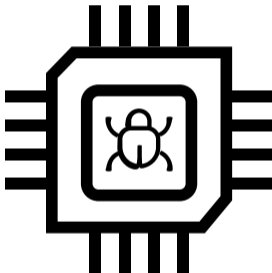
How to fix Hardware Bugs?



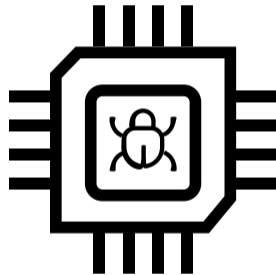
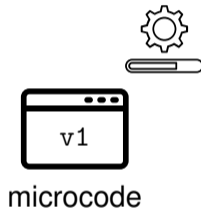
How to fix Hardware Bugs?



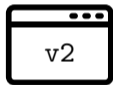
microcode



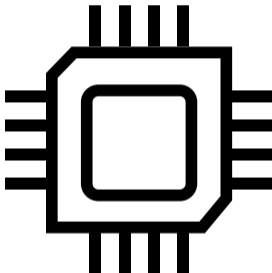
How to fix Hardware Bugs?



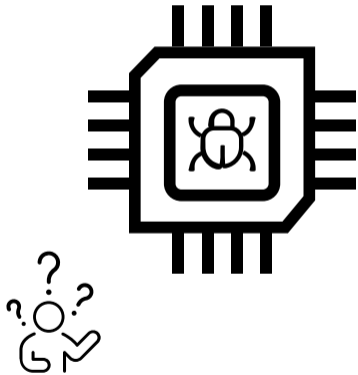
How to fix Hardware Bugs?



microcode

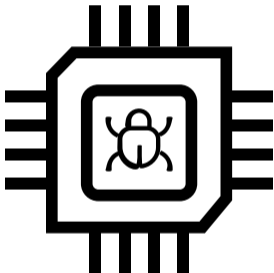


How to fix Hardware Bugs?



How to fix Hardware Bugs?

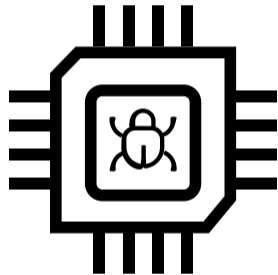
disable vector
extension



How to fix Hardware Bugs?



OS

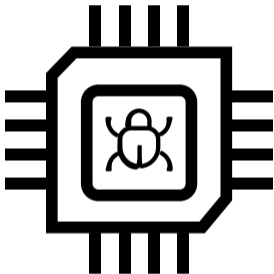


How to fix Hardware Bugs?

disable vector
extension



OS



How to fix Hardware Bugs?



OS: disable extension

How to fix Hardware Bugs?



OS: disable extension

up to 33% overhead

lose ~ 50% instructions

How to fix Hardware Bugs?



OS: disable extension

up to 33% overhead

lose ~ 50% instructions

How to fix Hardware Bugs?



OS: disable extension

up to 33% overhead

lose ~ 50% instructions



OS: disable extension

How to fix Hardware Bugs?

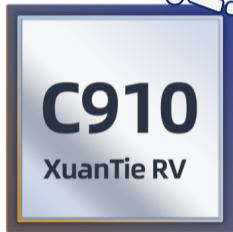


OS: disable extension
up to 33% overhead
lose ~ 50% instructions



OS: disable extension
up to 77% overhead
lose ~ 50% instructions

How to fix Hardware Bugs?



OS: disable extension
up to 33% overhead
lose ~ 50% instructions



OS: disable extension
up to 77% overhead
lose ~ 50% instructions



How to fix Hardware Bugs?



C910
XuanTie RV

OS: disable extension
up to 33% overhead
lose ~ 50% instructions



C908
XuanTie RV

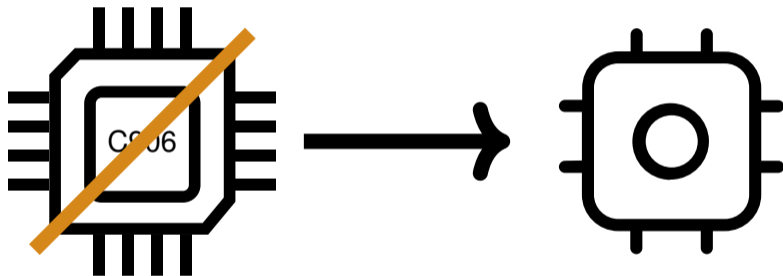
OS: disable extension
up to 77% overhead
lose ~ 50% instructions



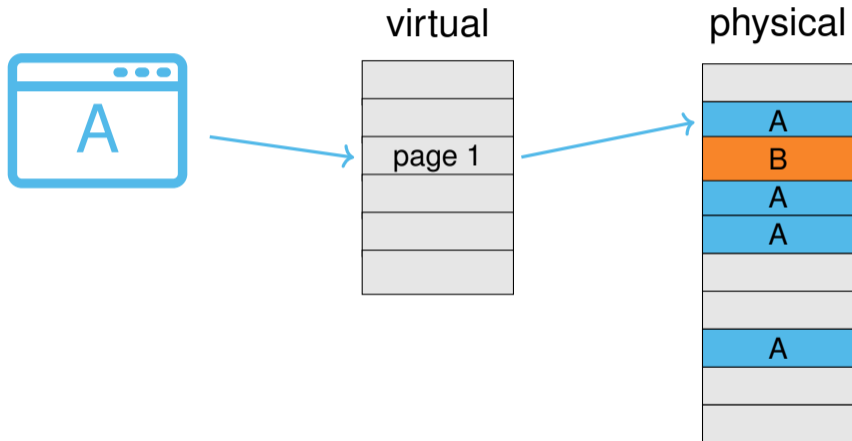
C906
XuanTie RV

no mitigation

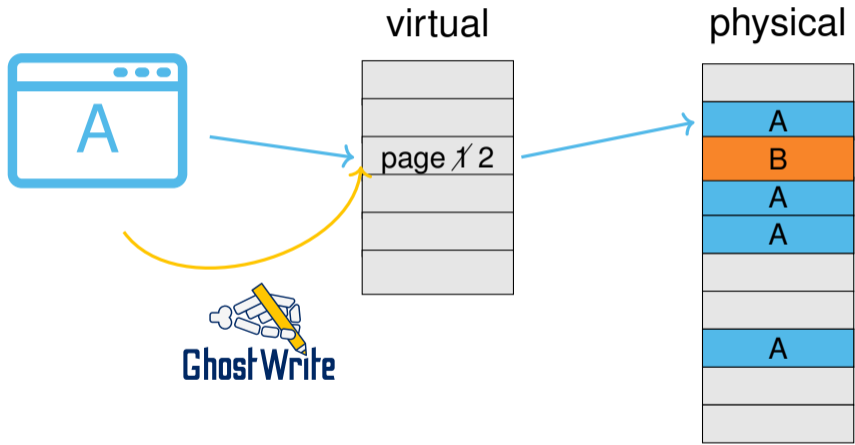
How to the fix C906?



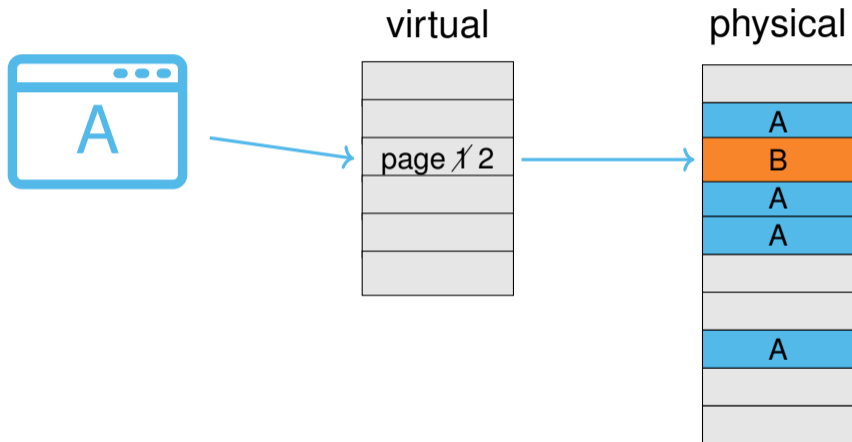
Reading Arbitrary Memory



Reading Arbitrary Memory



Reading Arbitrary Memory



Demo: Reading Arbitrary Memory

```
if kernel_get_user() is not root then
    require_authentication()
start_root_shell()
```



```
kernel_get_user:
```

```
    process = get_current_process()
    user = user_for_process(process)
    return user
```

OS

```
if kernel_get_user() is not root then  
    require_authentication()  
start_root_shell()
```



syscall

```
kernel_get_user:
```

```
    process = get_current_process()  
    user = user_for_process(process)  
    return user
```

OS

```
if kernel_get_user() is not root then  
    require_authentication()  
start_root_shell()
```



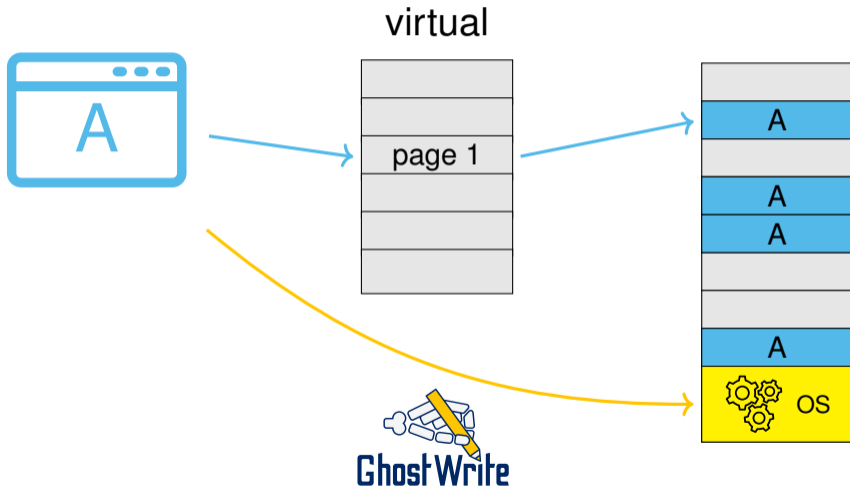
syscall

```
kernel_get_user:
```

```
    process = get_current_process()  
    user = user_for_process(process)  
    return root
```

OS

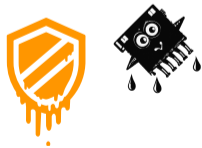
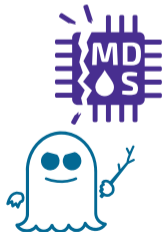
Getting root: Patching the Kernel



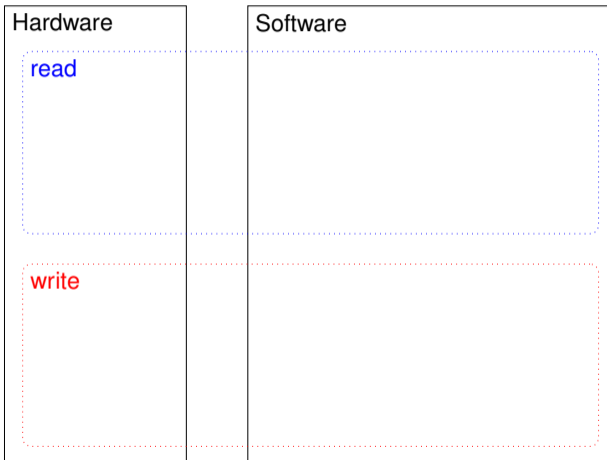
Is GhostWrite the only bug?



Is GhostWrite the only bug?



GhostWrite: The Big Picture



GhostWrite: The Big Picture

software

black hat
USA 2013

SICHER CLEARTEXT ANALYSIS
FOR CHEAT SHEETS

COLIN O'FLYNN

black hat
USA 2013

write

black hat
USA 2017

REGISTER NOW

JULY 22-27, 2017
MANDALAY BAY/LAS VEGAS, NV

ATTEND TRAININGS BRIEFINGS ARSENAL FEATURES SCHEDULE SPECIAL EVENTS SPONSORS PROPOSALS

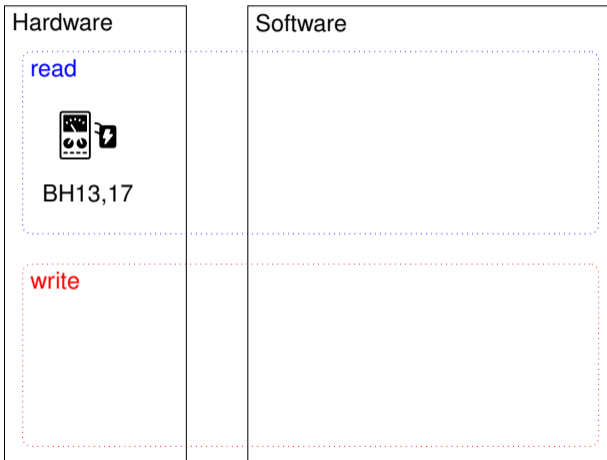
BACK TO TRAINING

ON THIS PAGE

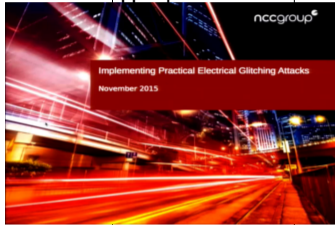
ADVANCED HARDWARE HACKING: HANDS-ON
POWER ANALYSIS & GLITCHING WITH THE
CHIPWHISPERER

NEWAE TECHNOLOGY INC. JULY 22-23 & JULY 24-25

GhostWrite: The Big Picture

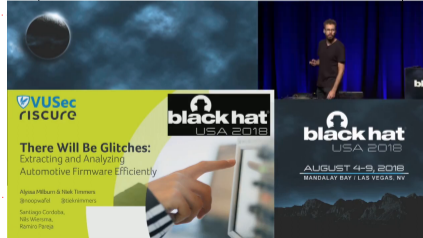


GhostWrite: The Big Picture

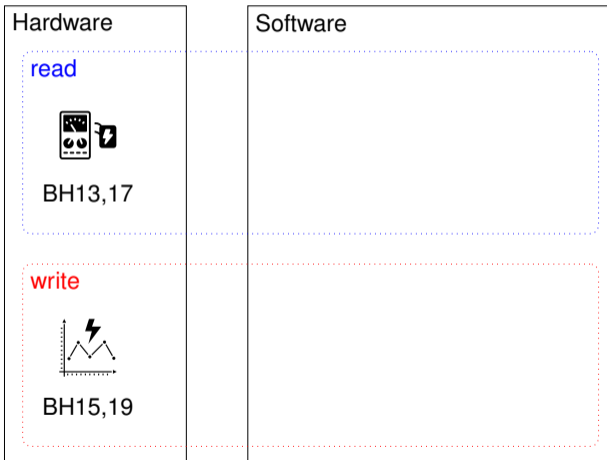


Software

write



GhostWrite: The Big Picture



Software

black hat
USA 2016

black hat
USA 2016

Side-Channel Attacks on
Everyday Applications

Taylor Hornby^{1,2}
(With thanks to Prof. John Aycock¹)
University of Calgary¹
Zcash²

write

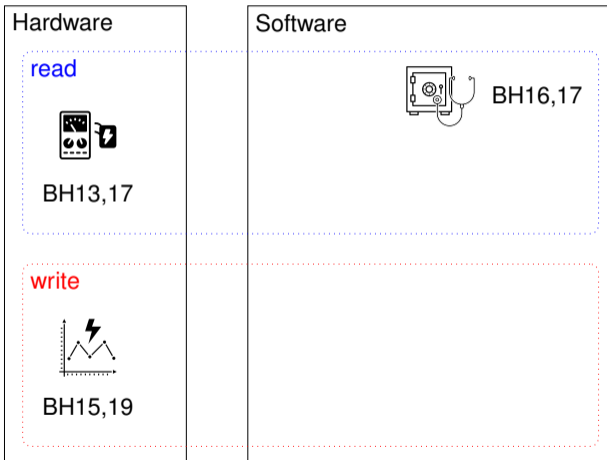
BH15,19


Hello from the Other Side:
SSH over Robust Cache Covert Channels in the Cloud

Michael Schwarz and Manuel Weber
March 30th, 2017

black hat
ASIA 2017

GhostWrite: The Big Picture






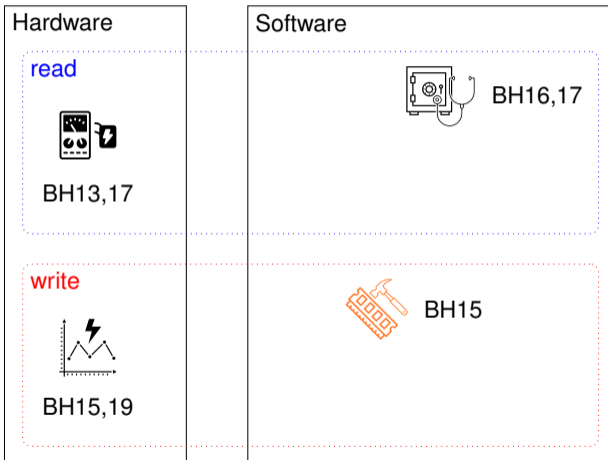
**Exploiting the DRAM
rowhammer bug to gain kernel
privileges**

How to cause and exploit
single bit errors

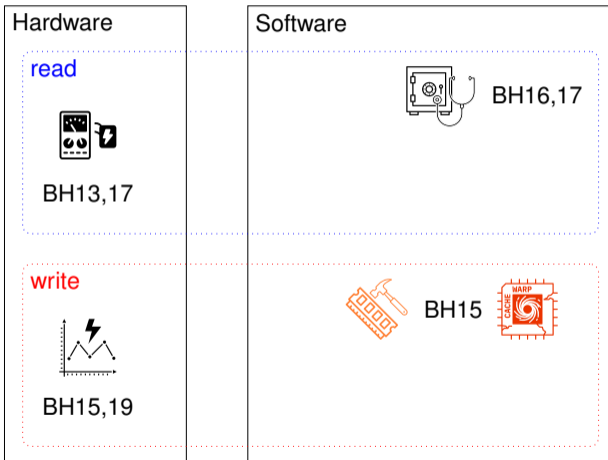
- Mark Seaborn and Halvar Flake



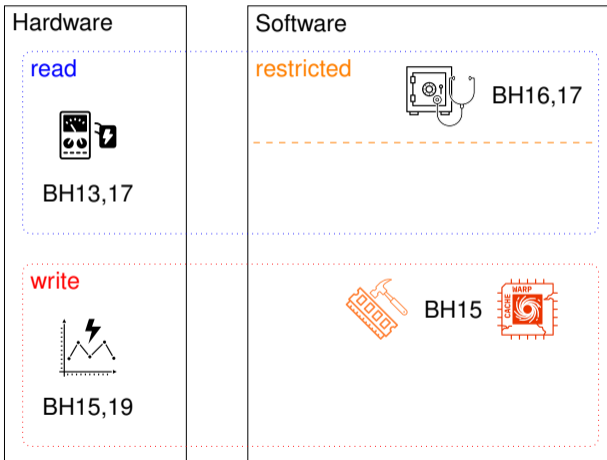
GhostWrite: The Big Picture



GhostWrite: The Big Picture



GhostWrite: The Big Picture



black hat
USA 2018

black hat
USA 2018

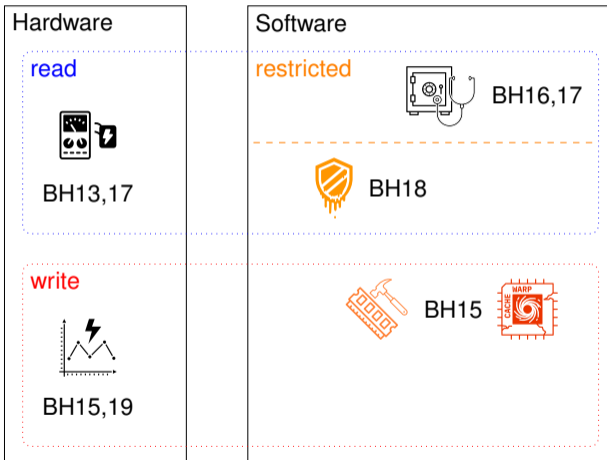
AUGUST 4-9, 2018
MANDALAY BAY / LAS VEGAS, NV

Meltdown
Basics, Details, Consequences


Black Hat USA 2018
9 August, 2018 - Las Vegas, NV, USA

Moritz Lipp (@mlqxyz)
Michael Schwarz (@misc0110)
Daniel Gruss (@lavados)

GhostWrite: The Big Picture



black hat
USA 2023



Downfall

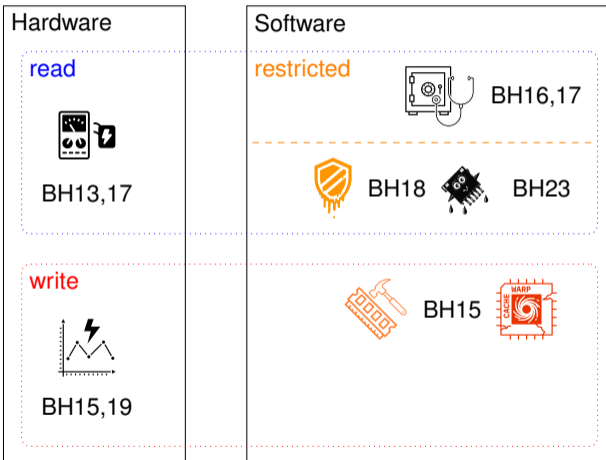
Single Instruction Multiple Data
Leaks in Cutting-edge CPUs

Daniel Moghimi – August 9, 2023
BlackHat USA, Las Vegas

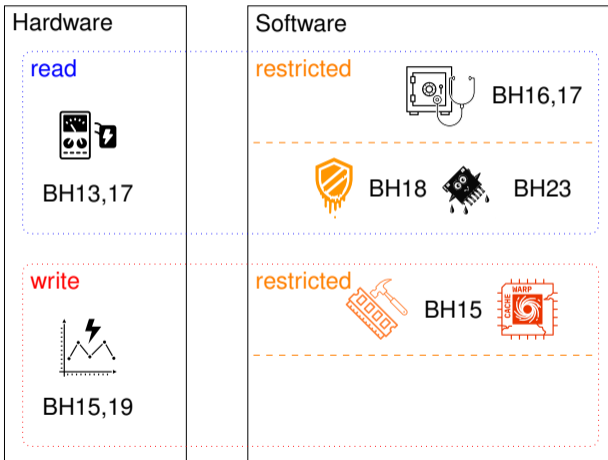
I'm going to talk about downfall,
single instruction, multiple data

AUGUST 5-10, 2023
MANDALAY BAY / LAS VEGAS

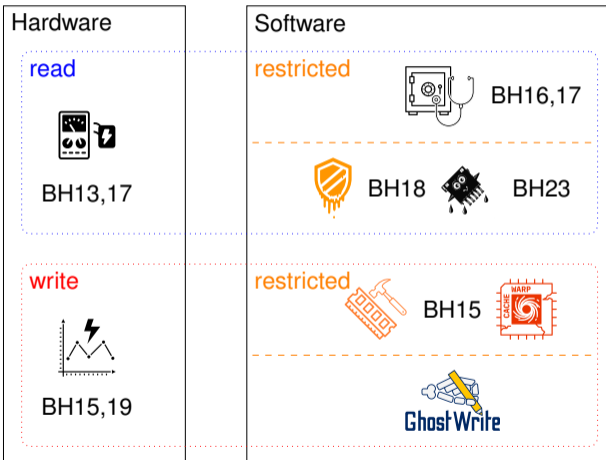
GhostWrite: The Big Picture



GhostWrite: The Big Picture



GhostWrite: The Big Picture



GhostWrite: Comparison



Rowhammer



CacheWarp



GhostWrite

	Rowhammer	CacheWarp	GhostWrite
Restrictions	<i>bit flips</i>		
Speed			
Practicality			

GhostWrite: Comparison







Rowhammer



CacheWarp



GhostWrite

	Rowhammer	CacheWarp	GhostWrite
Restrictions	<i>bit flips</i>	<i>old state</i>	
Speed			
Practicality			

GhostWrite: Comparison



Rowhammer



CacheWarp



GhostWrite

	Rowhammer	CacheWarp	GhostWrite
Restrictions	<i>bit flips</i>	<i>old state</i>	—
Speed	😭	😊	😊
Practicality	😞	😐	😊

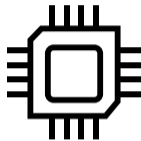


RISC-V is great

What can we learn?



RISC-V is great

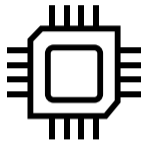


Only C910

What can we learn?



RISC-V is great



Only C910

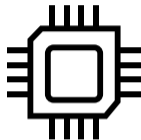


Quality control important

What can we learn?



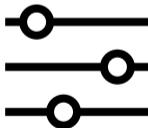
RISC-V is great



Only C910



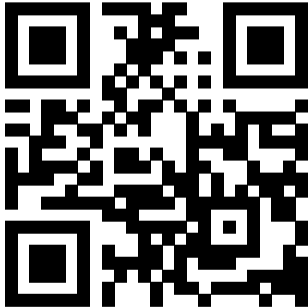
Quality control important



Configurable hardware



ghostwriteattack.com



Microarchitecture Vulnerabilities: Past, Present, and Future





- GhostWrite destroys all isolations on C910 RISC-V CPU

@fth0mas @hetterichlorenz



ghostwriteattack.com



- GhostWrite destroys all isolations on C910 RISC-V CPU
- Mitigation: disable vector extension, up to 33% overhead

@fth0mas @hetterichlorenz

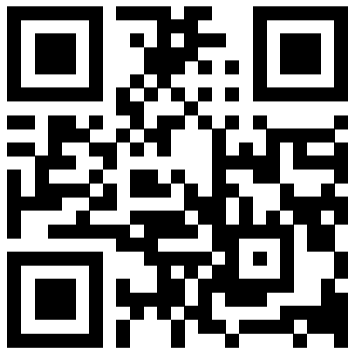


ghostwriteattack.com



- GhostWrite destroys all isolations on C910 RISC-V CPU
- Mitigation: disable vector extension, up to 33% overhead
- Hardware bugs are everywhere

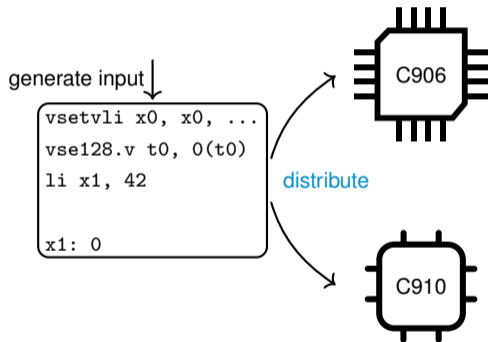
@fth0mas @hetterichlorenz



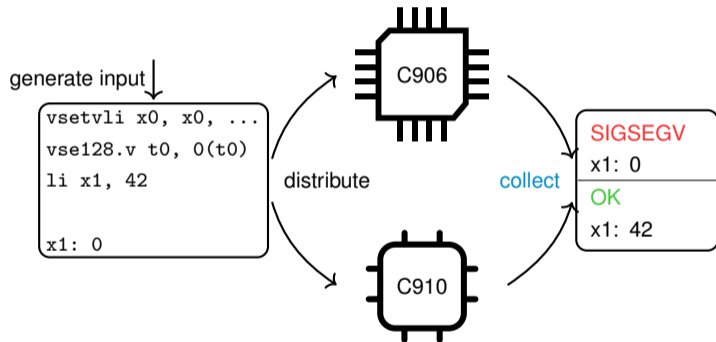
ghostwriteattack.com

generate input ↓

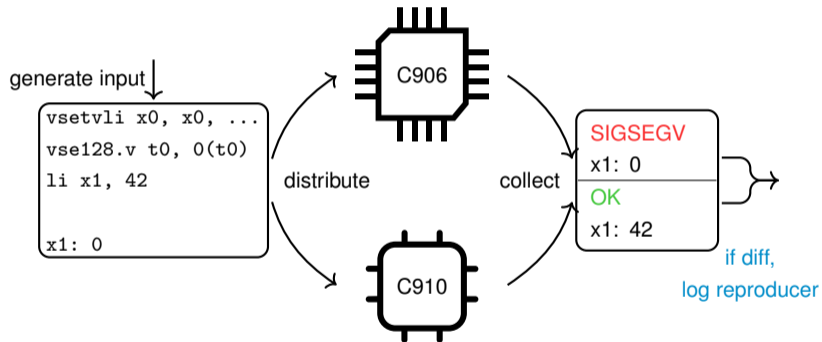
```
vsetvli x0, x0, ...  
vse128.v t0, 0(t0)  
li x1, 42  
  
x1: 0
```



GhostWrite: The Framework



GhostWrite: The Framework



GhostWrite: The Framework

