Chapter 7 - Software-based Power Attacks

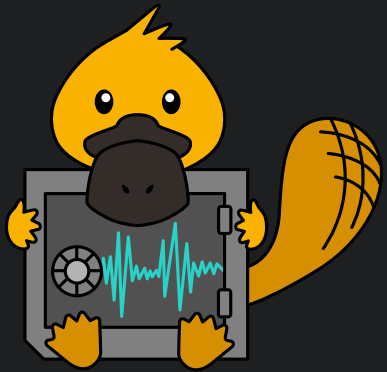
Attacking CPUs with Power Side Channels from Software

Mathias Oberhuber

18th April 2024

```
➜  ~      cat /sys/class/powercap/intel-rapl:0/intel-rapl:0:0/energy_uj
90211251602
```

```
➜  ~ sudo cat /sys/class/powercap/intel-rapl:0/intel-rapl:0:0/energy_uj
90211251602
```



Background covered in the lecture

- Side-channel attacks
- Power analysis e.g. **SPA**, **DPA**
- RSA implementations and optimizations
- Different types of attacks

CPU Power Management

- CPU power management is complex

CPU Power Management

- CPU power management is complex
- In order to save power, you can …

CPU Power Management

- CPU power management is complex
- In order to save power, you can …

Shut down resources

CPU Power Management

- CPU power management is complex
- In order to save power, you can …

Shut down resources

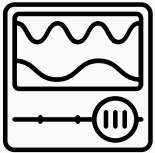
Reduce voltage

CPU Power Management

- CPU power management is complex
- In order to save power, you can …

Shut down resources

Reduce voltage



Reduce frequency

Intel RAPL

- Therefore, the CPU requires:

Intel RAPL

- Therefore, the CPU requires:
  - Thermal Management

Intel RAPL

- Therefore, the CPU requires:
  - Thermal Management
  - Platform Power Limiting

Intel RAPL

- Therefore, the CPU requires:
  - Thermal Management
  - Platform Power Limiting
  - Power/Performance Budgeting

Intel RAPL

- Therefore, the CPU requires:
  - Thermal Management
  - Platform Power Limiting
  - Power/Performance Budgeting
- Domains: PKG, CORE, MC

Intel RAPL

- Therefore, the CPU requires:
  - Thermal Management
  - Platform Power Limiting
  - Power/Performance Budgeting
- Domains: PKG, CORE, MC
- **Intel Running Average Power Limit** (RAPL) provides:

Intel RAPL

- Therefore, the CPU requires:
  - Thermal Management
  - Platform Power Limiting
  - Power/Performance Budgeting
- Domains: PKG, CORE, MC
- **Intel Running Average Power Limit** (RAPL) provides:

power limiting

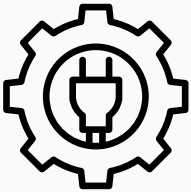
Intel RAPL

- Therefore, the CPU requires:
  - Thermal Management
  - Platform Power Limiting
  - Power/Performance Budgeting
- Domains: PKG, CORE, MC
- **Intel Running Average Power Limit** (RAPL) provides:

power limiting

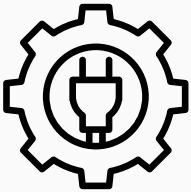
energy reading

Intel RAPL



- **Linux**: accessed via `powercap` framework
  `/sys/devices/virtual/powercap/intel-rapl`

Intel RAPL



- **Linux**: accessed via `powercap` framework
  `/sys/devices/virtual/powercap/intel-rapl`
- **macOS** and **Windows**: Intel driver needs to be installed

Intel RAPL: Properties

Unprivileged power meter

Intel RAPL: Properties

Unprivileged power meter

No physical access

Intel RAPL: Properties

Unprivileged power meter

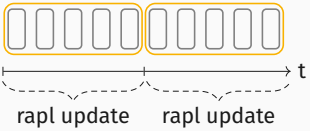
No physical access

Low refresh rate

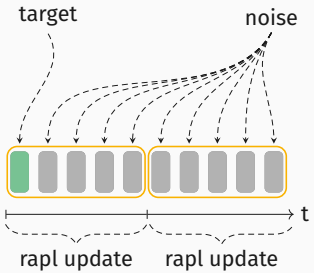
RAPL: Measurement Techniques

target noise

- Measure an **instruction** by

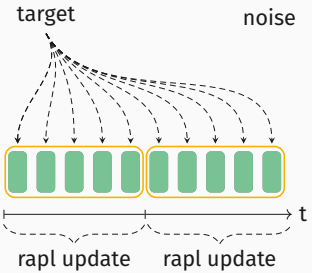


RAPL: Measurement Techniques



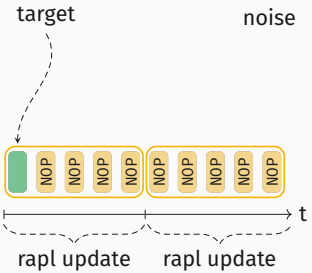
- Measure an **instruction** by
  - executing it once

RAPL: Measurement Techniques



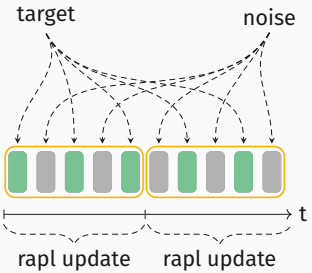
- Measure an **instruction** by
  - executing it once
  - executing it repeatedly

RAPL: Measurement Techniques

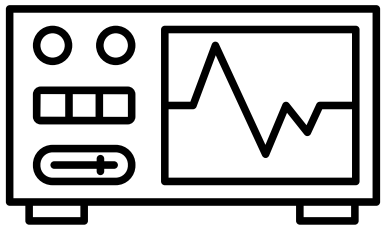


- Measure an **instruction** by
  - executing it once
  - executing it repeatedly
  - padding it with known instructions

RAPL: Measurement Techniques



- Measure an **instruction** by
  - executing it once
  - executing it repeatedly
  - padding it with known instructions
  - reissue the instruction after an interrupt



What can we do with this?

Distinguishing Instructions

- Measure the energy consumption of **different instructions**

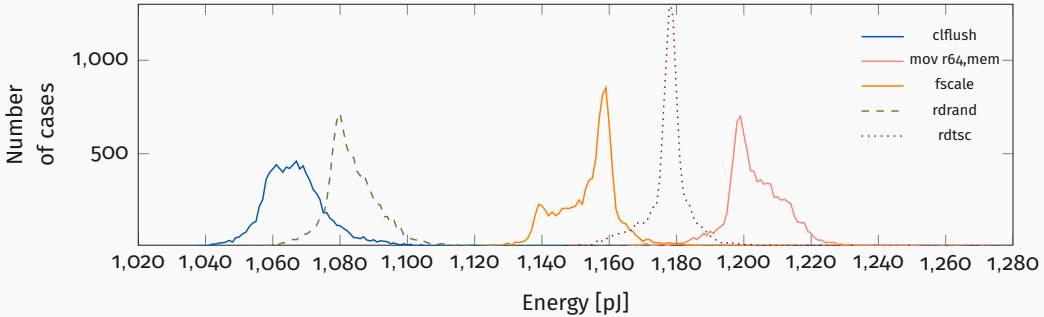
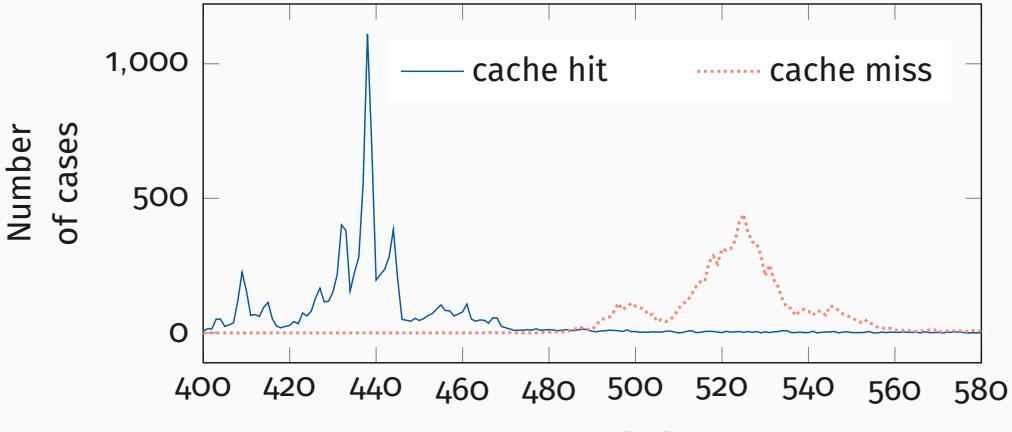


Figure 1: A histogram of the power consumption of various instructions on the i7-6700K (desktop) system.

Distinguishing Load Targets

- Measure the energy consumption of **different load targets**



Distinguishing Operands

- Measure the energy consumption of **different operands**

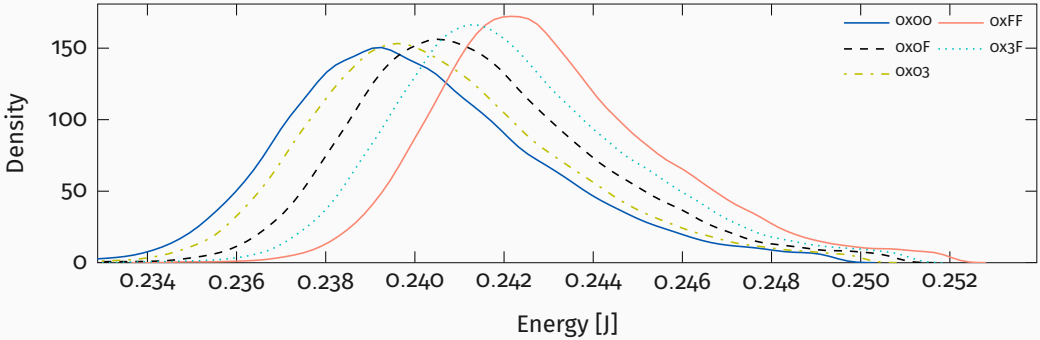


Figure 3: Measured energy consumption of the `imul` instruction with one operand fixed to 8 and the other varying in its Hamming weight.

Distinguishing Data

- Measure the energy consumption of **different load values**

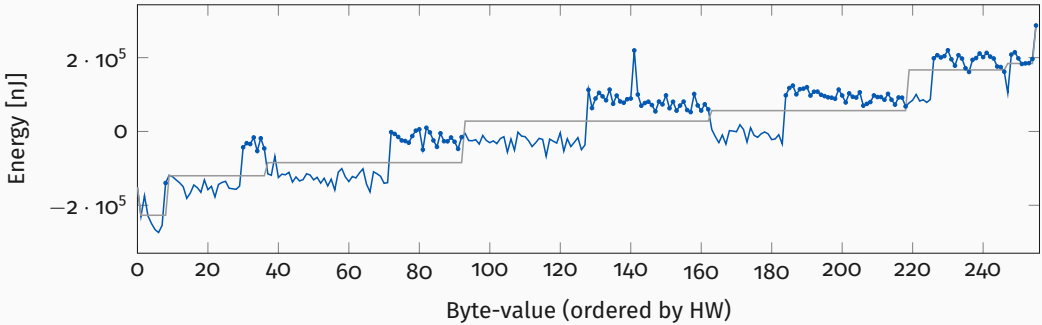


Figure 4: Energy consumption of the `movb` instruction for all byte values, ordered by Hamming Weight (HW) and value. The circle marks values where the most-significant bit is set.

Let’s exploit this!

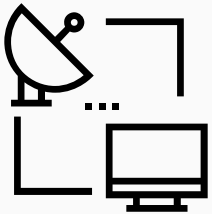
Covert Channel

- Hidden communication channel

Covert Channel

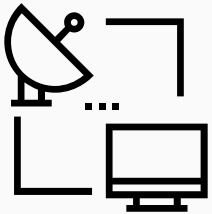
- Hidden communication channel
- Leveraging the power side channel

Covert Channel



- 2 Processes, Sender and Receiver
  - **Send a 1**: Perform energy-consuming instructions
  - **Send a 0**: Idle

Covert Channel



- 2 Processes, Sender and Receiver
  - **Send a 1**: Perform energy-consuming instructions
  - **Send a 0**: Idle
- Receiver measures power consumption

→ **Deduces transmitted bit**

Covert Channel

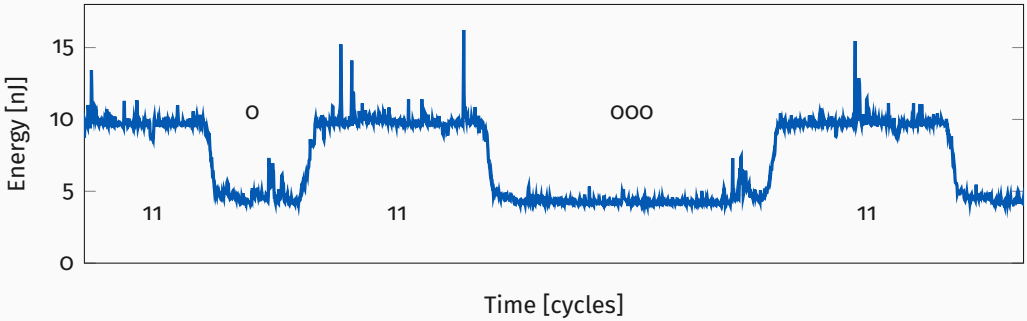
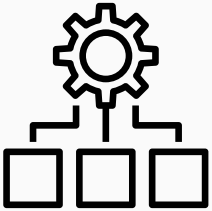


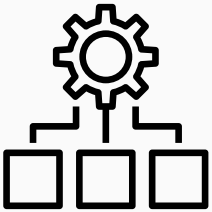
Figure 5: Transmission of bits `1101100011` using the time-less covert channel.

Covert Channel



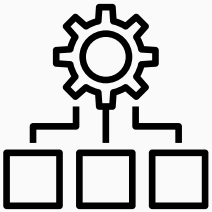
- Not limited to 2 processes

Covert Channel



- Not limited to 2 processes
- **Xen Hypervisor** granted guests access to the RAPL registers

Covert Channel



- Not limited to 2 processes
- **Xen Hypervisor** granted guests access to the RAPL registers
- Establish a covert channel between 2 guests

```
    switch ( msr )
    {
    case MSR_AMD_PATCHLOADER:
    case MSR_IA32_UCODE_WRITE:
    case MSR_PRED_CMD:
    case MSR_FLUSH_CMD:
        /* Write-only */
    case MSR_TEST_CTRL:
    case MSR_CORE_CAPABILITIES:
    case MSR_TSX_FORCE_ABORT:
    case MSR_TSX_CTRL:
    case MSR_MCU_OPT_CTRL:
    case MSR_RTIT_OUTPUT_BASE ... MSR_RTIT_ADDR_B(7):
    case MSR_RAPL_POWER_UNIT:
    case MSR_PKG_POWER_LIMIT  ... MSR_PKG_POWER_INFO:
    case MSR_DRAM_POWER_LIMIT ... MSR_DRAM_POWER_INFO:
    case MSR_PP0_POWER_LIMIT  ... MSR_PP0_POLICY:
    case MSR_PP1_POWER_LIMIT  ... MSR_PP1_POLICY:
    case MSR_PLATFORM_ENERGY_COUNTER:
    case MSR_PLATFORM_POWER_LIMIT:
    case MSR_U_CET:
    case MSR_S_CET:
    case MSR_PL0_SSP ... MSR_INTERRUPT_SSP_TABLE:
    case MSR_AMD64_LWP_CFG:
    case MSR_AMD64_LWP_CBADDR:
    case MSR_PPIN_CTL:
    case MSR_PPIN:
    case MSR_F15H_CU_POWER ... MSR_F15H_CU_MAX_POWER:
    case MSR_AMD_RAPL_POWER_UNIT ... MSR_AMD_PKG_ENERGY_STATUS:
    case MSR_AMD_PPIN_CTL:
    case MSR_AMD_PPIN:
        /* Not offered to guests. */
        goto gp_fault;
```

Breaking KASLR

- Kernel Address Space Layout Randomization (KASLR)

Breaking KASLR

- Kernel Address Space Layout Randomization (KASLR)
- Exploit **energy consumption differences** between

Breaking KASLR

- Kernel Address Space Layout Randomization (KASLR)
- Exploit **energy consumption differences** between
  - Mapped addresses

Breaking KASLR

- Kernel Address Space Layout Randomization (KASLR)
- Exploit **energy consumption differences** between
  - Mapped addresses
  - Unmapped addresses

Breaking KASLR

- Kernel Address Space Layout Randomization (KASLR)
- Exploit **energy consumption differences** between
  - Mapped addresses
  - Unmapped addresses
- **Valid address translations** are cached in the **TLB**

Figure 6: Repeated Page-table walks for unmapped pages require more power

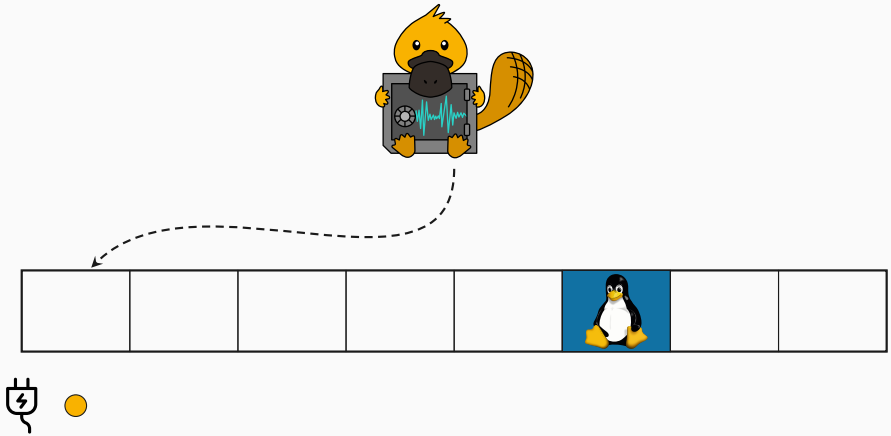


Figure 6: Repeated Page-table walks for unmapped pages require more power

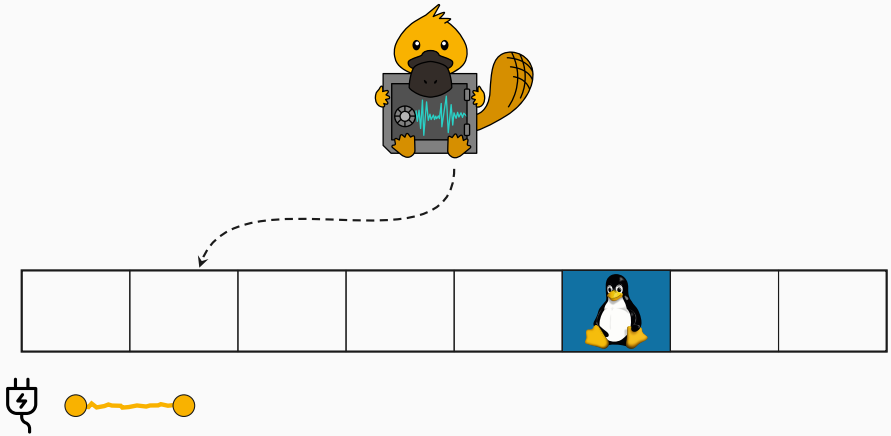


Figure 6: Repeated Page-table walks for unmapped pages require more power

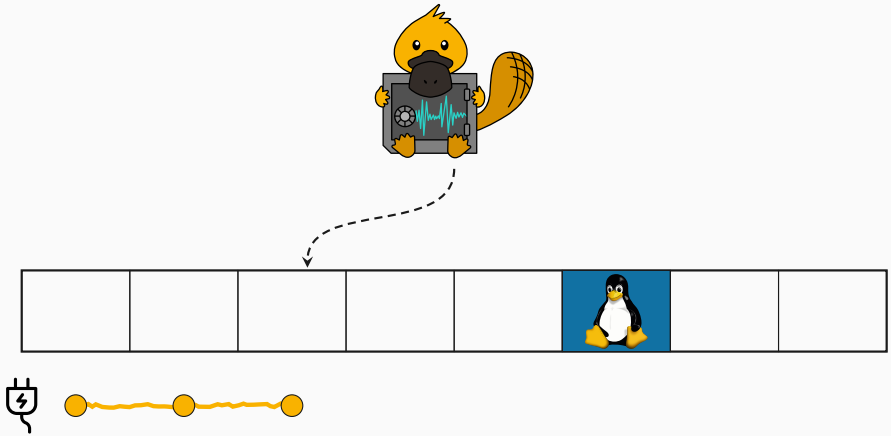


Figure 6: Repeated Page-table walks for unmapped pages require more power

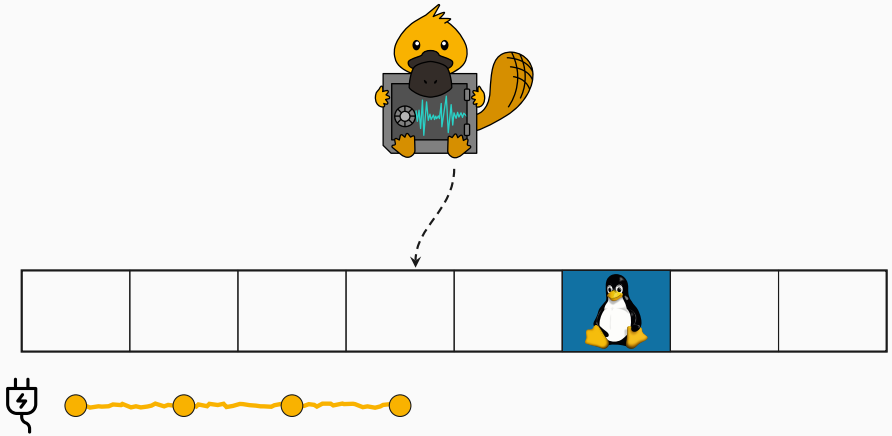


Figure 6: Repeated Page-table walks for unmapped pages require more power

Breaking KASLR

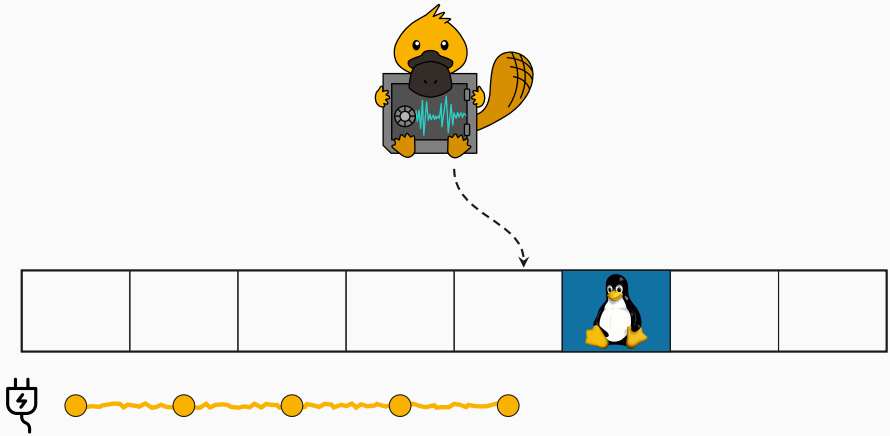


Figure 6: Repeated Page-table walks for unmapped pages require more power

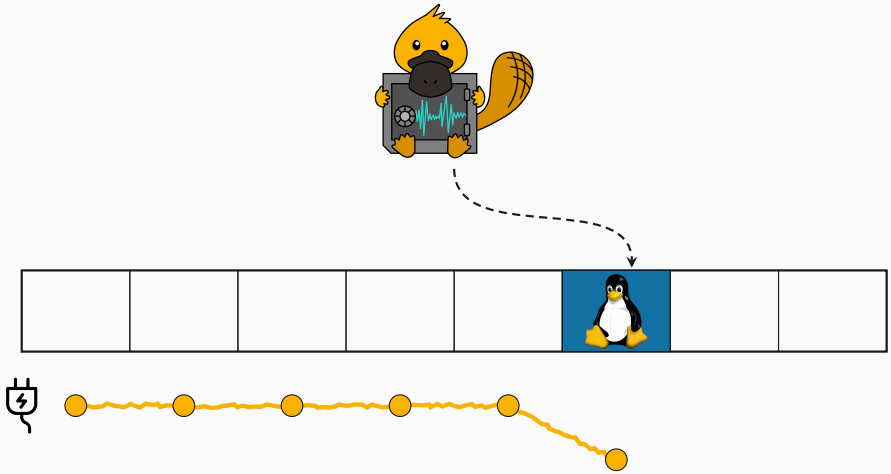


Figure 6: Repeated Page-table walks for unmapped pages require more power

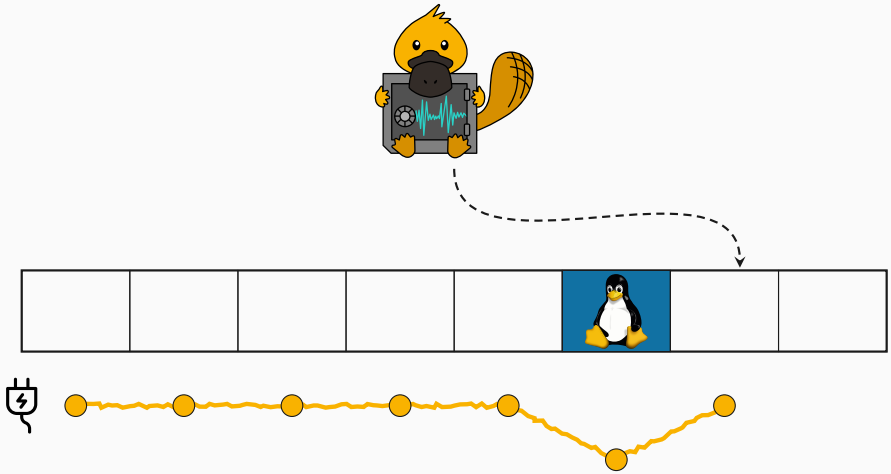


Figure 6: Repeated Page-table walks for unmapped pages require more power

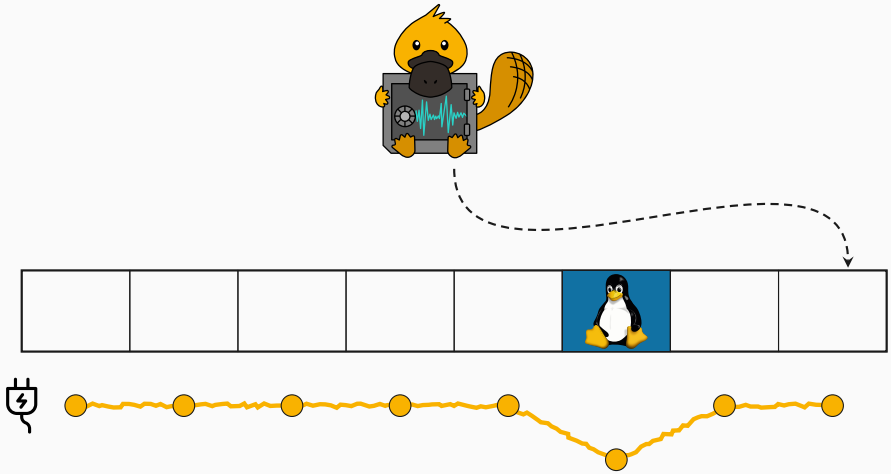


Figure 6: Repeated Page-table walks for unmapped pages require more power

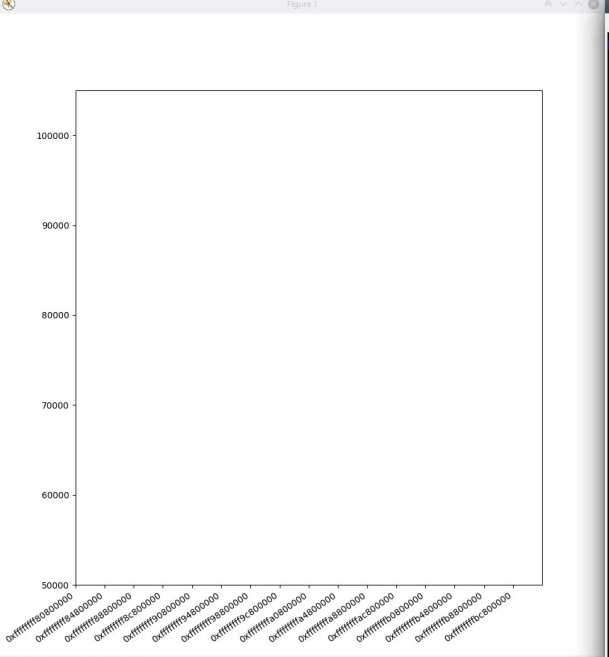
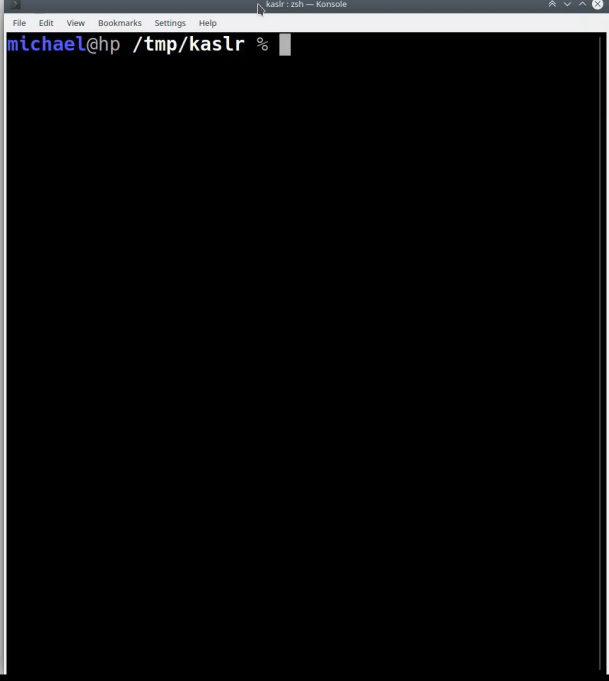
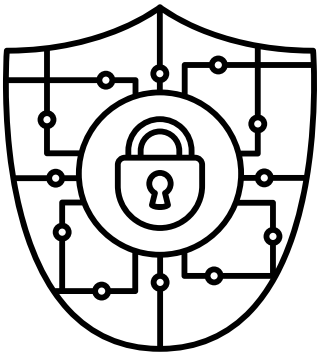


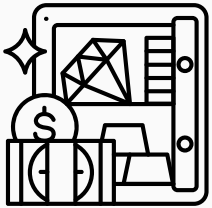
Figure 1
100000
90000
80000
70000
60000
50000
0xffffff8e8800000
0xffffff8f4800000
0xffffff8f8800000
0xffffff8fc800000
0xffffff908800000
0xffffff914800000
0xffffff918800000
0xffffff91c800000
0xffffffa0800000
0xffffffa4800000
0xffffffa8800000
0xffffffac800000
0xffffffb0800000
0xffffffb4800000
0xffffffb8800000
0xffffffbc800000


kaslr : zsh — Konsole
File
Edit
View
Bookmarks
Settings
Help
michael@hp /tmp/kaslr %



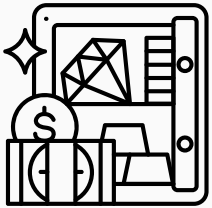
Attacking Crypto: RSA Key Recovery

Attacking Intel SGX



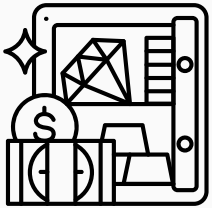
- Instruction-set extension

Attacking Intel SGX



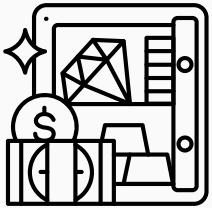
- Instruction-set extension
- Integrity and confidentiality in **untrusted environments**

Attacking Intel SGX



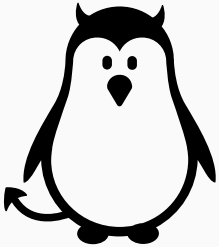
- Instruction-set extension
- Integrity and confidentiality in **untrusted environments**
- **Enclaves** offer protected areas of memory

Attacking Intel SGX



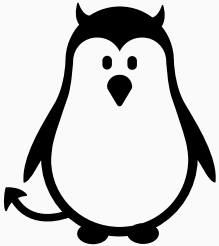
- Instruction-set extension
- Integrity and confidentiality in **untrusted environments**
- **Enclaves** offer protected areas of memory
- Operating system can be **compromised**

Attacking Intel SGX: Threat Model



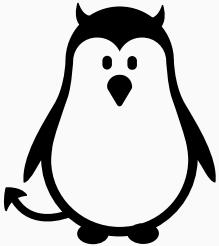
- More power as an evil operating system

Attacking Intel SGX: Threat Model



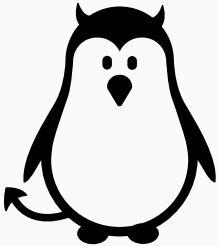
- More power as an evil operating system
- Hook the SGX Enclave exit point

Attacking Intel SGX: Threat Model



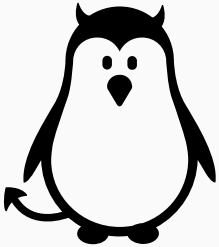
- More power as an evil operating system
- Hook the SGX Enclave exit point
- Directly read out the **RAPL values** from the MSRs

Attacking Intel SGX: Threat Model



- More power as an evil operating system
- Hook the SGX Enclave exit point
- Directly read out the **RAPL values** from the MSRs
- No operating system overhead!

Attacking Intel SGX: Threat Model



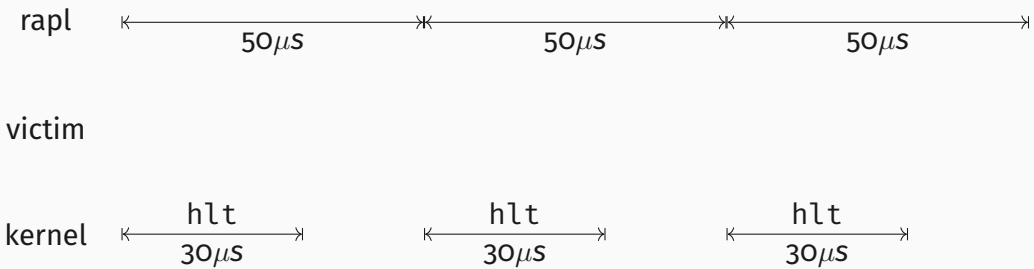
- More power as an evil operating system
- Hook the SGX Enclave exit point
- Directly read out the **RAPL values** from the MSRs
- No operating system overhead!
- Interrupt victim often to increase resolution

Halt Delay

- RAPL domains have a nearly fixed update interval

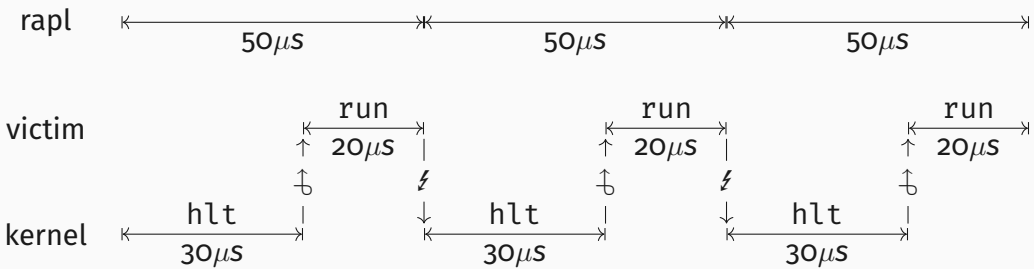
Halt Delay

- RAPL domains have a nearly fixed update interval
- Delay the interrupt return with the halt delay in the ISR

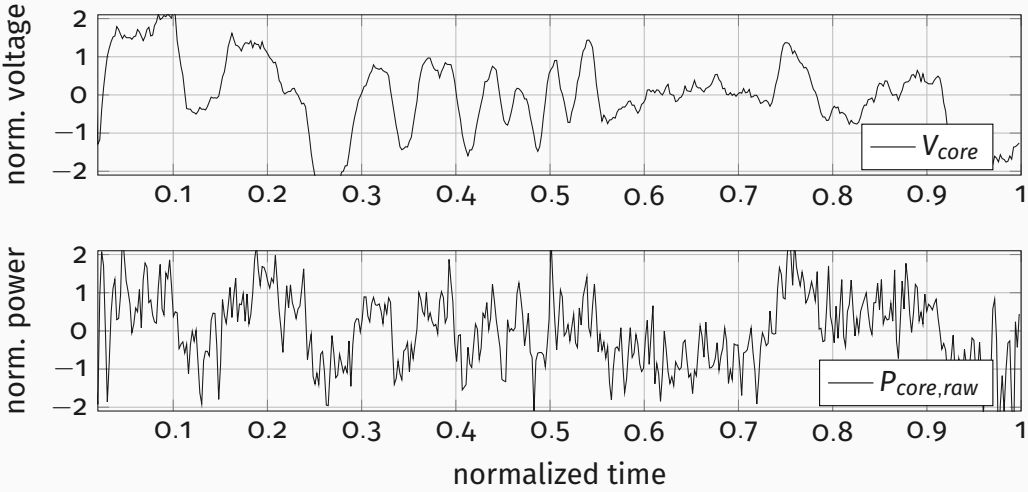


Halt Delay

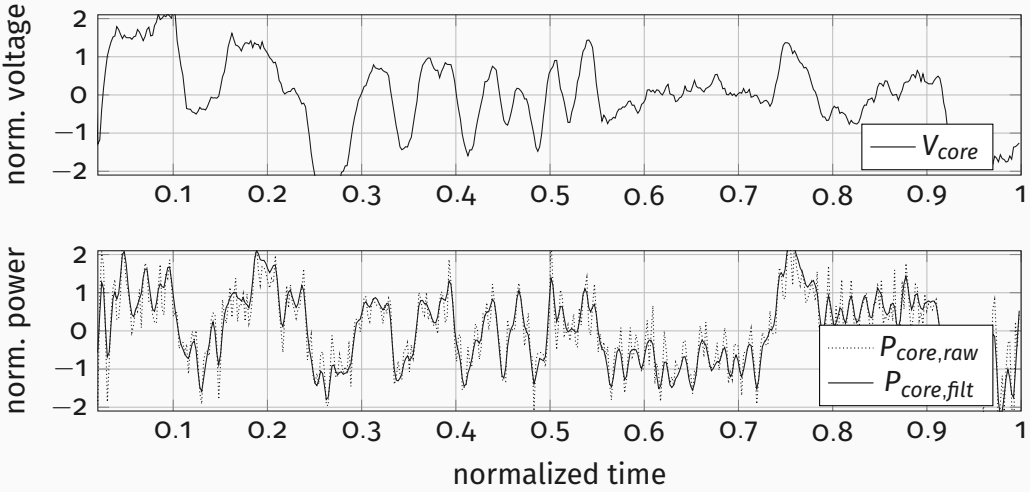
- RAPL domains have a nearly fixed update interval
- Delay the interrupt return with the halt delay in the ISR
- Reduces the execution time of the victim in the current interval



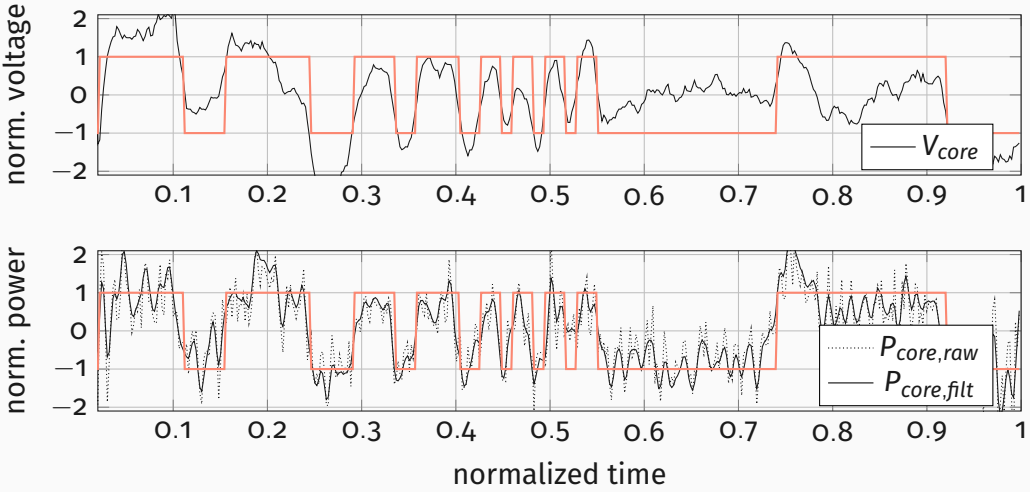
SPA Attack - Results



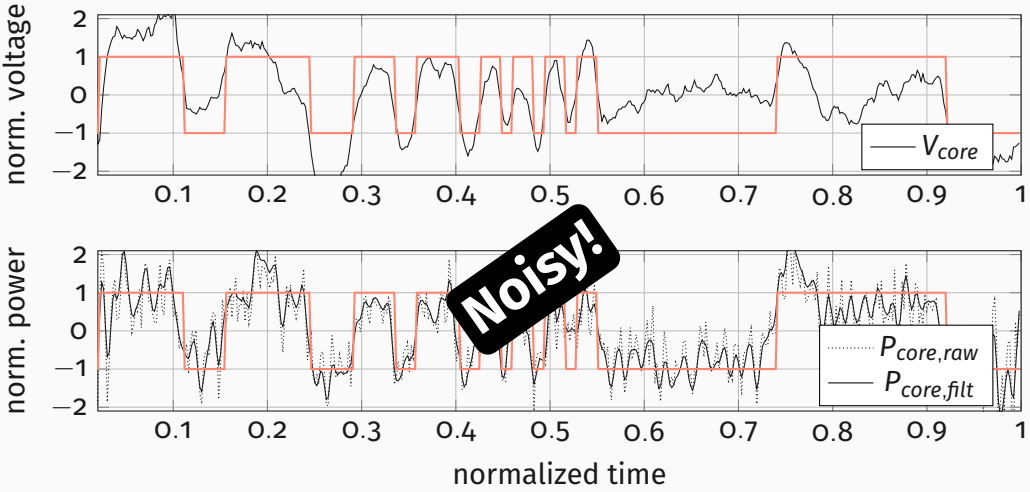
SPA Attack - Results



SPA Attack - Results



SPA Attack - Results



SGX Step

- **SGX-step** is an open-source Linux kernel framework

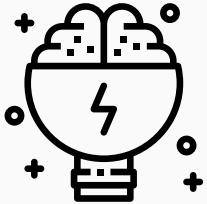
SGX Step

- **SGX-step** is an open-source Linux kernel framework
- Configure APIC timer interrupts

SGX Step

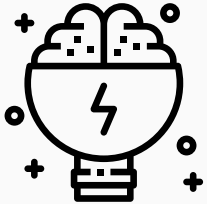
- **SGX-step** is an open-source Linux kernel framework
- Configure APIC timer interrupts
- Single and zero-step enclave execution

SGX Step



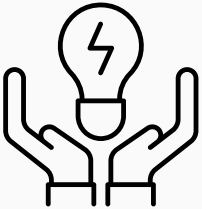
- **Combine** Intel RAPL with SGX-step

SGX Step



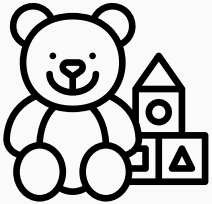
- **Combine** Intel RAPL with SGX-step
- Measure the energy consumption of **single instructions**

Attacking RSA



- Implemented using a Square-and-multiply algorithm
  - **Keybit 0**: Compute square operation
  - **Keybit 1**: Compute square operation and multiplication
- Branches consume different amounts of energy

Toy Example

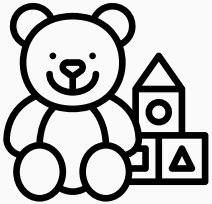


- Square-and-Multiply example with few instructions

```
1  case_one:
2  vpmuludq  %ymm0,%ymm0,%ymm0
3  vpmuludq  %ymm0,%ymm1,%ymm0
4  movslq  -0x4(%rax),%rdx
5  add     $0x4,%rax
6  mov     -0x28(%rsp,%rdx,8),%rdx
7  jmpq    *%rdx
8
```

```
1  case_zero:
2  vpmuludq  %ymm0,%ymm0,%ymm0
3  vpmuludq  %ymm2,%ymm1,%ymm0
4  movslq  -0x4(%rax),%rdx
5  add     $0x4,%rax
6  mov     -0x28(%rsp,%rdx,8),%rdx
7  jmpq    *%rdx
8
```

Toy Example

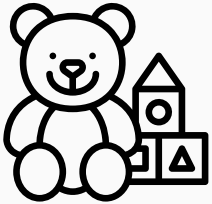


- Square-and-Multiply example with few instructions
- If the key bit is zero discard multiplication

```
1  case_one:
2  vpmuludq  %ymm0,%ymm0,%ymm0
3  vpmuludq  %ymm0,%ymm1,%ymm0
4  movslq  -0x4(%rax),%rdx
5  add     $0x4,%rax
6  mov     -0x28(%rsp,%rdx,8),%rdx
7  jmpq    *%rdx
8
```

```
1  case_zero:
2  vpmuludq  %ymm0,%ymm0,%ymm0
3  vpmuludq  %ymm2,%ymm1,%ymm0
4  movslq  -0x4(%rax),%rdx
5  add     $0x4,%rax
6  mov     -0x28(%rsp,%rdx,8),%rdx
7  jmpq    *%rdx
8
```

Toy Example

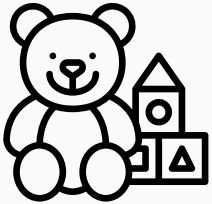


- Square-and-Multiply example with few instructions
- If the key bit is zero discard multiplication
- Use jump table to determine next key bit

```
1  case_one:
2  vpmuludq  %ymm0,%ymm0,%ymm0
3  vpmuludq  %ymm0,%ymm1,%ymm0
4  movslq  -0x4(%rax),%rdx
5  add     $0x4,%rax
6  mov     -0x28(%rsp,%rdx,8),%rdx
7  jmpq    *%rdx
8
```

```
1  case_zero:
2  vpmuludq  %ymm0,%ymm0,%ymm0
3  vpmuludq  %ymm2,%ymm1,%ymm0
4  movslq  -0x4(%rax),%rdx
5  add     $0x4,%rax
6  mov     -0x28(%rsp,%rdx,8),%rdx
7  jmpq    *%rdx
8
```

Toy Example

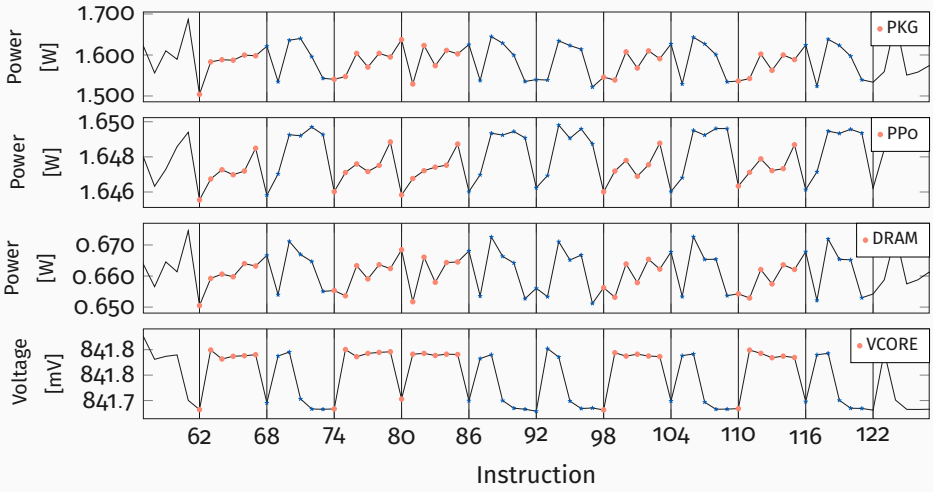


- Square-and-Multiply example with few instructions
- If the key bit is zero discard multiplication
- Use jump table to determine next key bit
- Target each instruction in the enclave function

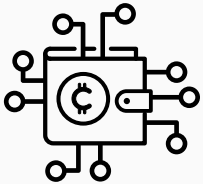
```
1  case_one:
2  vpmuludq  %ymm0,%ymm0,%ymm0
3  vpmuludq  %ymm0,%ymm1,%ymm0
4  movslq  -0x4(%rax),%rdx
5  add     $0x4,%rax
6  mov     -0x28(%rsp,%rdx,8),%rdx
7  jmpq    *%rdx
8
```

```
1  case_zero:
2  vpmuludq  %ymm0,%ymm0,%ymm0
3  vpmuludq  %ymm2,%ymm1,%ymm0
4  movslq  -0x4(%rax),%rdx
5  add     $0x4,%rax
6  mov     -0x28(%rsp,%rdx,8),%rdx
7  jmpq    *%rdx
8
```

RSA Toy Cipher

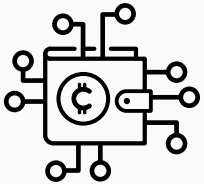


Attacking mbed TLS



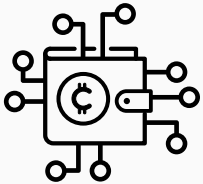
- **Extract RSA** key from mbed TLS 2.13.0

Attacking mbed TLS



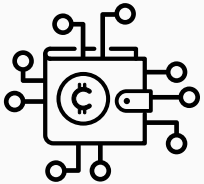
- **Extract RSA** key from mbed TLS 2.13.0
- Square-and-multiply algorithm

Attacking mbed TLS



- **Extract RSA** key from mbed TLS 2.13.0
- Square-and-multiply algorithm
- Multiplication function uses AVX memset

Attacking mbed TLS



- **Extract RSA** key from mbed TLS 2.13.0
- Square-and-multiply algorithm
- Multiplication function uses AVX memset
- Number of instructions executed depends on the key

Attacking mbed TLS

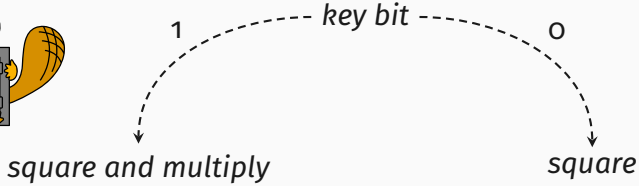
key bit

Attacking mbed TLS

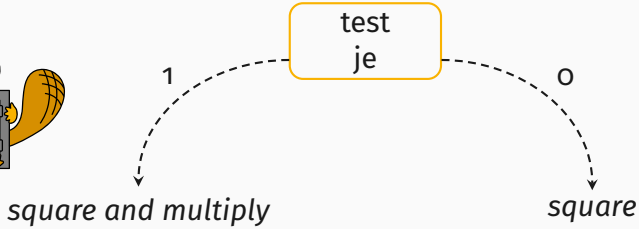
1 ------ *key bit*

square and multiply

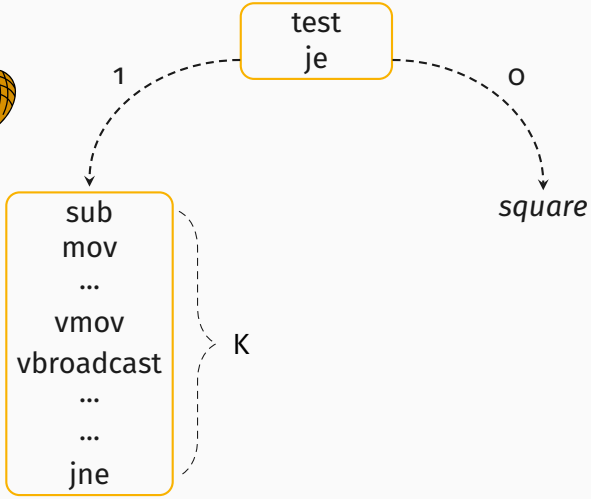
Attacking mbed TLS



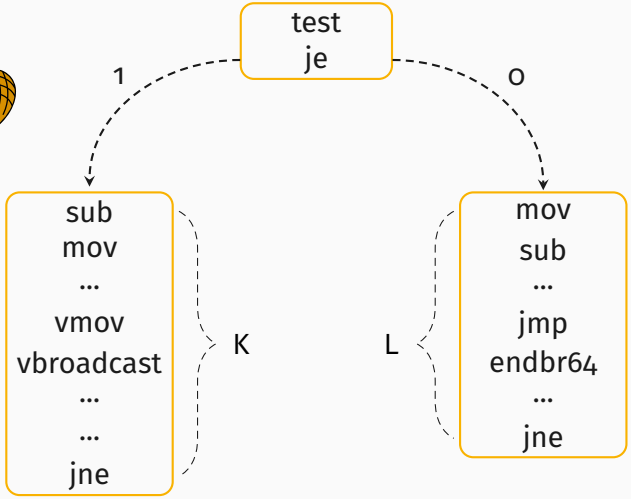
Attacking mbed TLS



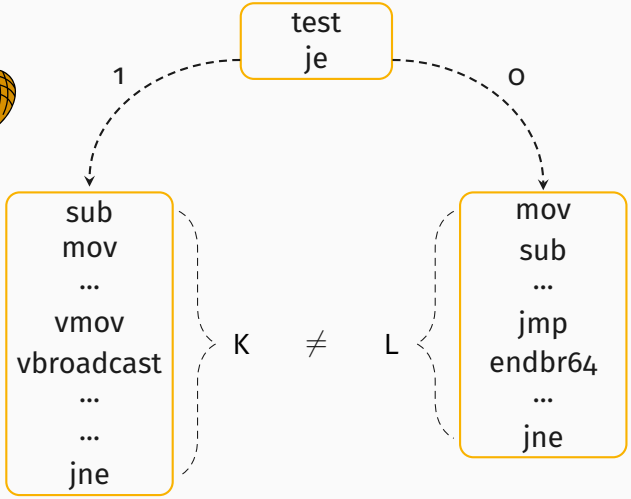
Attacking mbed TLS



Attacking mbed TLS



Attacking mbed TLS

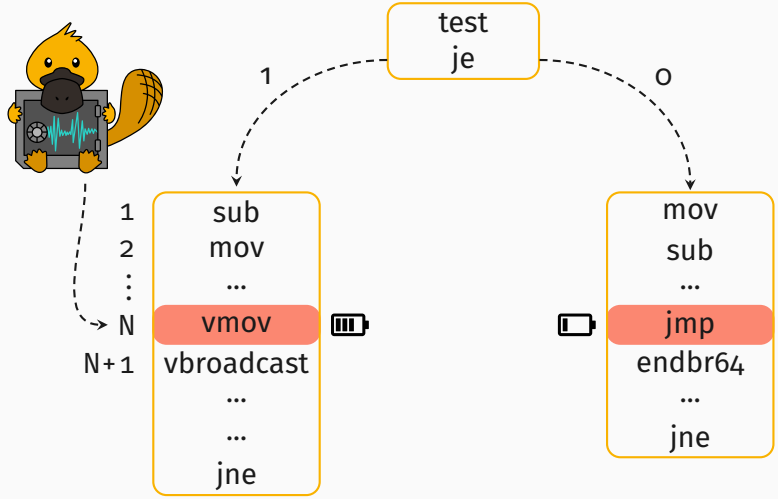


Attacking mbed TLS

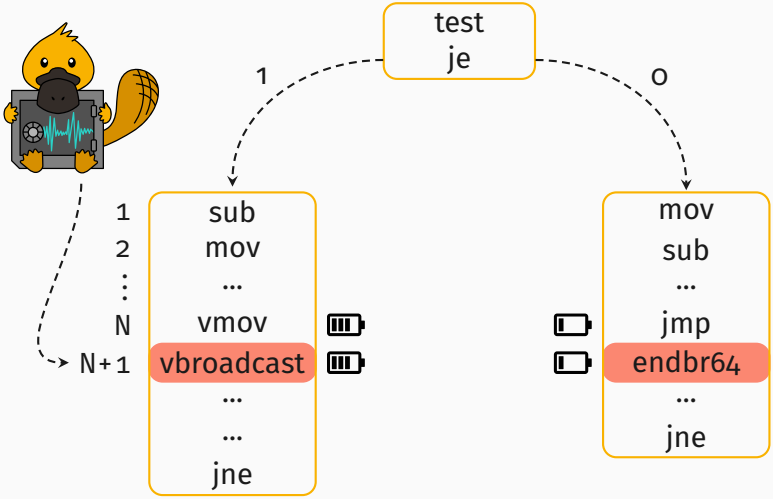
Attacking mbed TLS

Attacking mbed TLS

Attacking mbed TLS



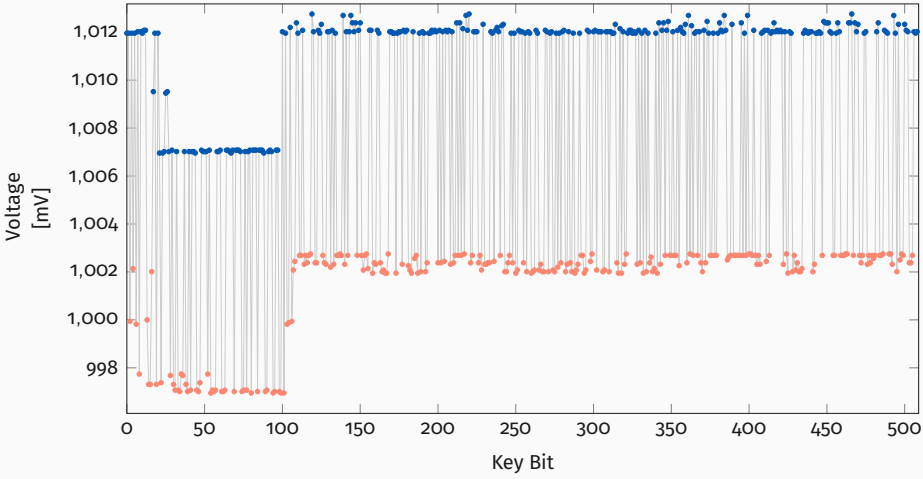
Attacking mbed TLS



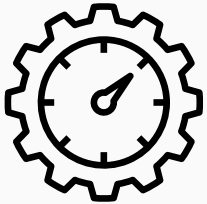
Attacking mbed TLS

Attacking mbed TLS

Attacking mbed TLS

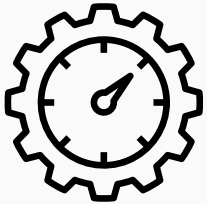


Performance



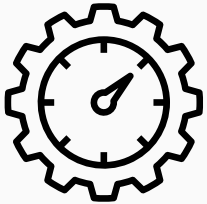
- Time per key bit increases linearly based on the index

Performance



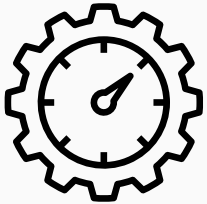
- Time per key bit increases linearly based on the index
- 3h 31m for a **512 bit**

Performance



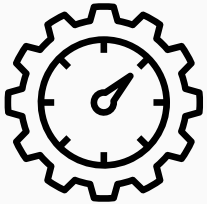
- Time per key bit increases linearly based on the index
- 3h 31m for a **512 bit**
  - 52 minutes for finding target instruction

Performance

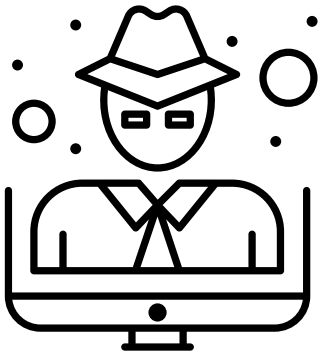


- Time per key bit increases linearly based on the index
- 3h 31m for a **512 bit**
  - 52 minutes for finding target instruction
- Record 3 samples per key bit

Performance

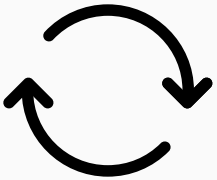


- Time per key bit increases linearly based on the index
- 3h 31m for a **512 bit**
  - 52 minutes for finding target instruction
- Record 3 samples per key bit
  - This could be extend to a single trace attack



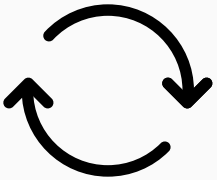
Crypto Attacks from User Space

Repeat Repeat Repeat



- **Difficult** to measure parts without SGX-step

Repeat Repeat Repeat



- **Difficult** to measure parts without SGX-step
- **Can** measure over the **overall execution**

Correlation Power Analysis

- Building a power consumption **model** of the device:

Correlation Power Analysis

- Building a power consumption **model** of the device:

Hamming Weight

Number of bits set

Correlation Power Analysis

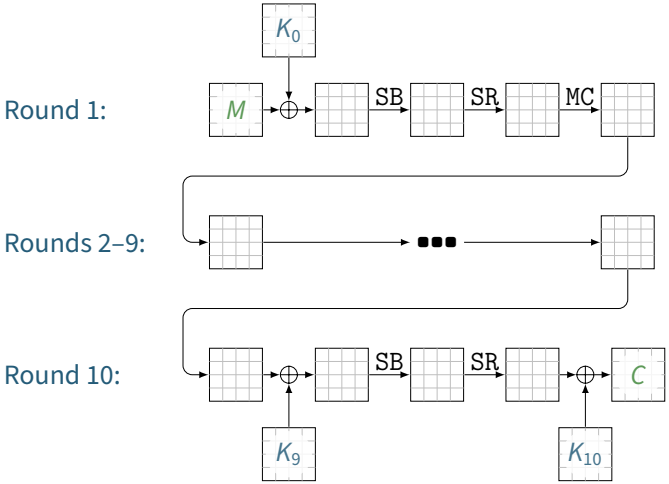
- Building a power consumption **model** of the device:

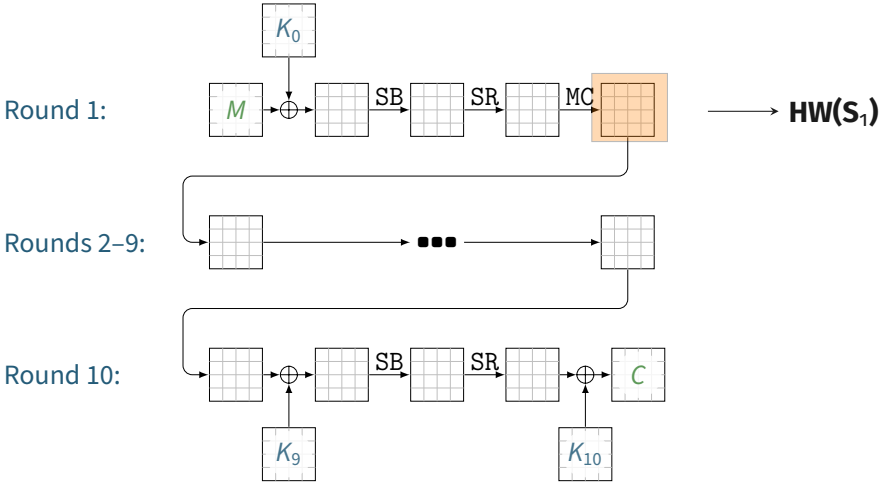
Hamming Weight

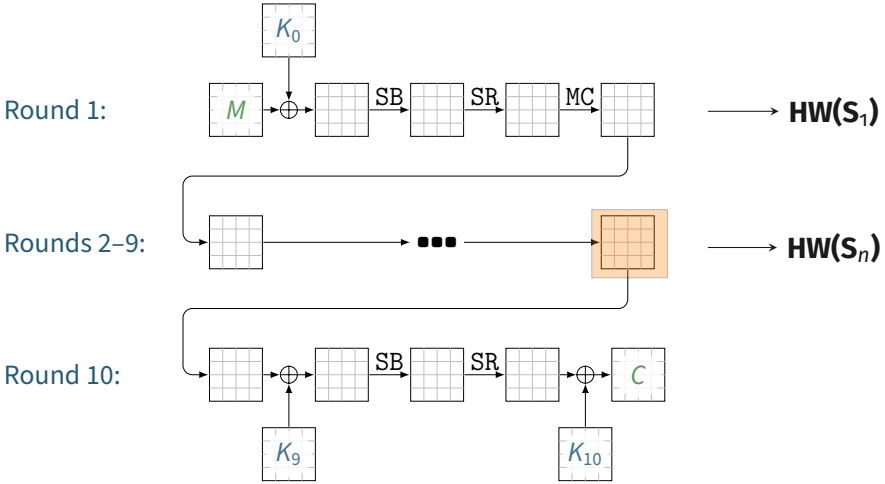
Number of bits set

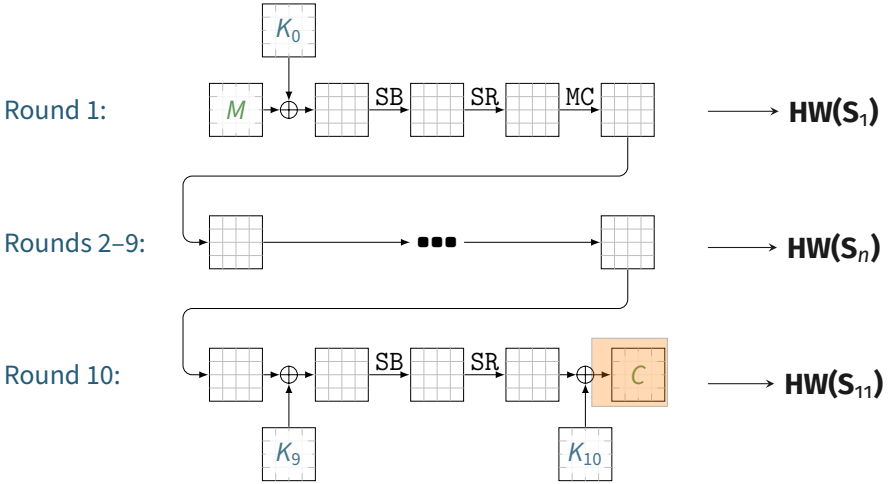
Hamming Distance

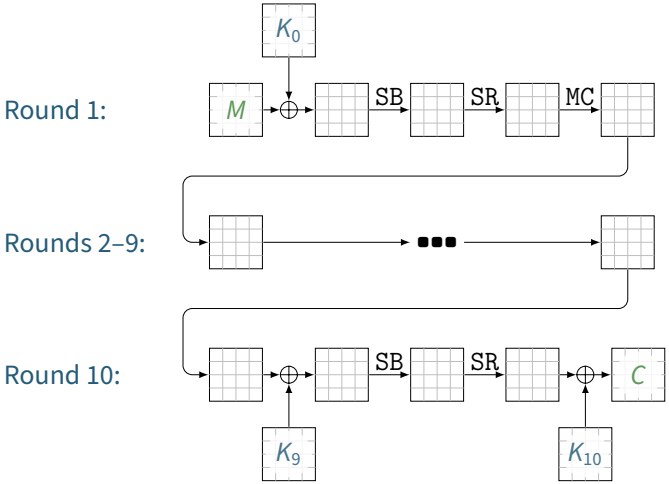
Bits flipping between operations

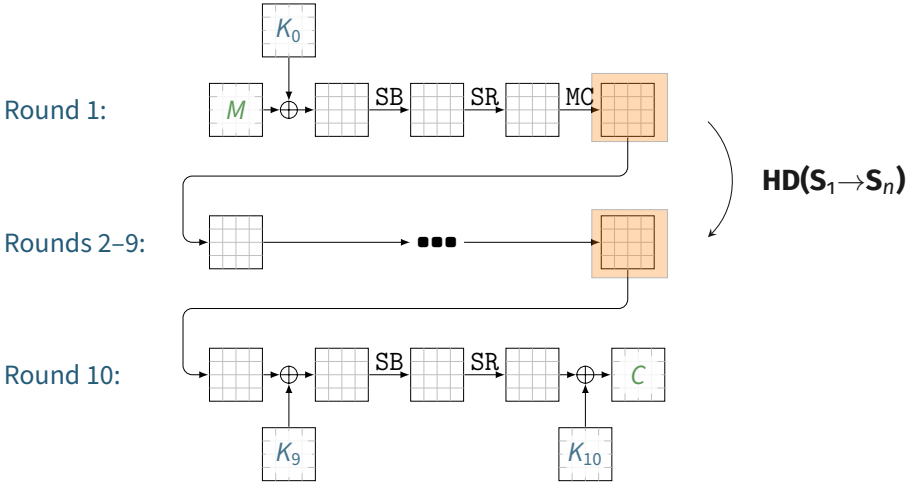

Round 1:
K_0
M
SB
SR
MC
Rounds 2–9:
Round 10:
SB
SR
C
K_9
K_{10}

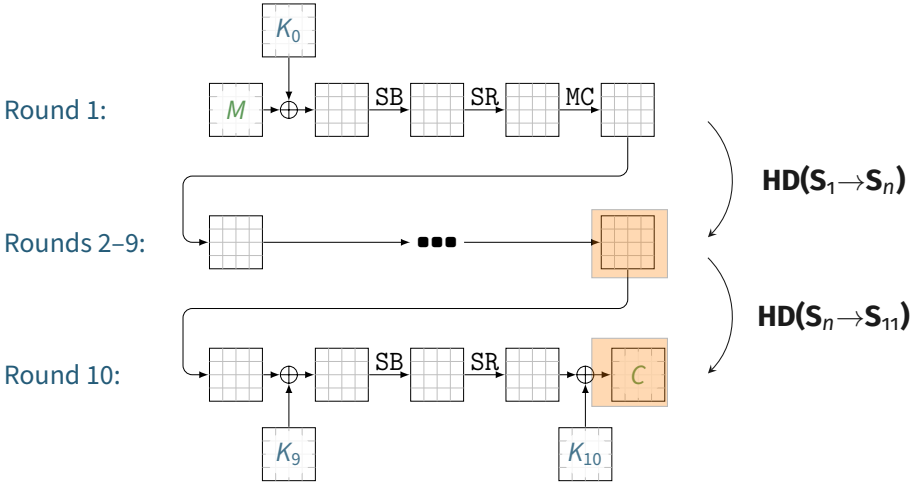

Round 1:
K_0
M
SB
SR
MC
HW(S_1)
Rounds 2–9:
Round 10:
SB
SR
C
K_9
K_{10}


Round 1:
K_0
M
SB
SR
MC
HW(S_1)
Rounds 2–9:
HW(S_n)
Round 10:
SB
SR
C
K_9
K_{10}


Round 1:
K_0
M
SB
SR
MC
HW(S_1)
Rounds 2–9:
HW(S_n)
Round 10:
SB
SR
C
HW(S_{11})
K_9
K_{10}


Round 1:
K_0
M
SB
SR
MC
Rounds 2–9:
Round 10:
SB
SR
C
K_9
K_{10}


Round 1:
K_0
M
SB
SR
MC
Rounds 2–9:
HD($S_1 \rightarrow S_n$)
Round 10:
SB
SR
C
K_9
K_{10}


Round 1:
K_0
M
SB
SR
MC
Rounds 2–9:
$\mathbf{HD(S_1 \to S_n)}$
Round 10:
SB
SR
C
K_9
K_{10}
$\mathbf{HD(S_n \to S_{11})}$

Correlation Power Analysis

- **AES-NI**: Side-channel resilient instruction-set extension
- Target **AES-NI** in a scenario where we can trigger encryption/decryption of many blocks
  - Disk encryption/decryption
  - TLS
  - (Un)sealing SGX enclave state

Correlation Power Analysis

- We control the plain text

Correlation Power Analysis

- We control the plain text
- We observe the cipher text

Correlation Power Analysis

- We control the plain text
- We observe the cipher text
- We measure the energy consumption over many operations

Correlation Power Analysis

- We control the plain text
- We observe the cipher text
- We measure the energy consumption over many operations
- We guess the key

Correlation Power Analysis

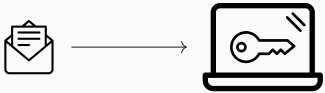
- We control the plain text
- We observe the cipher text
- We measure the energy consumption over many operations
- We guess the key

- With our model and all possible values, **where** is the **correlation** the highest?

CPA Attack

CPA Attack

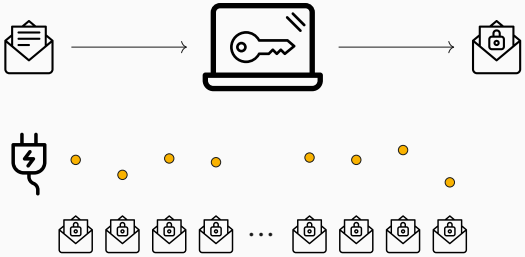
CPA Attack



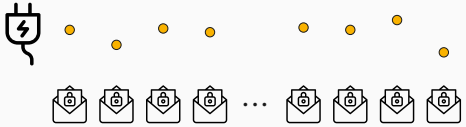
CPA Attack

CPA Attack

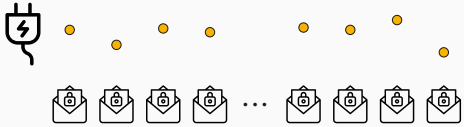
CPA Attack



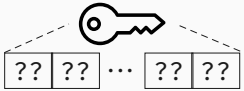
CPA Attack

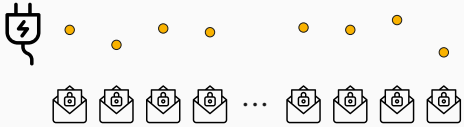


CPA Attack

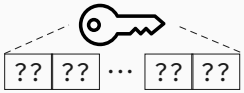


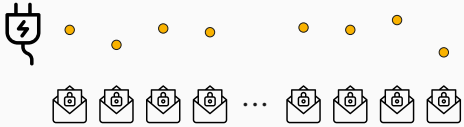
CPA Attack



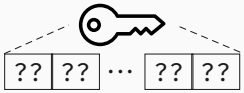


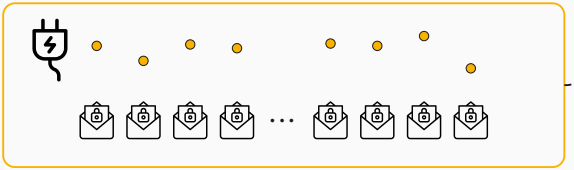
CPA Attack



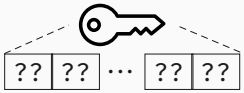


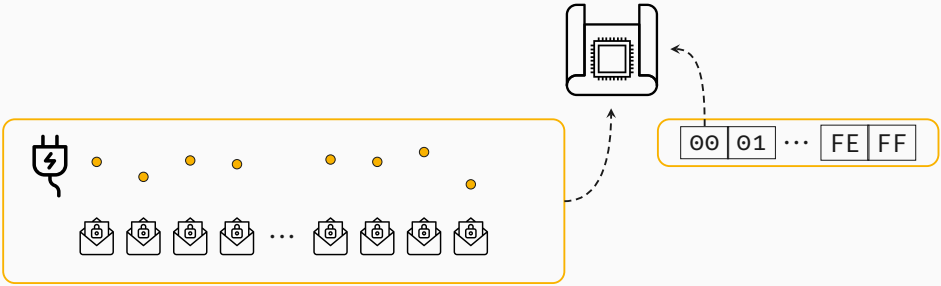
CPA Attack



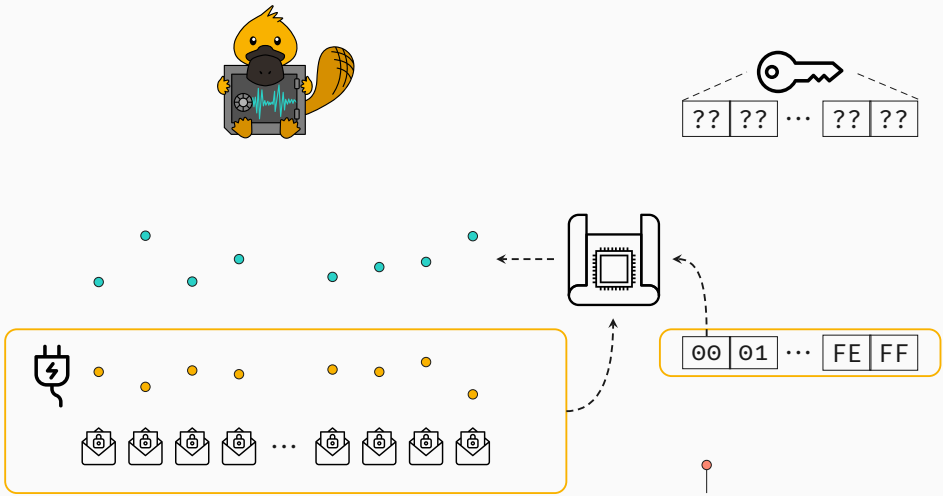


CPA Attack

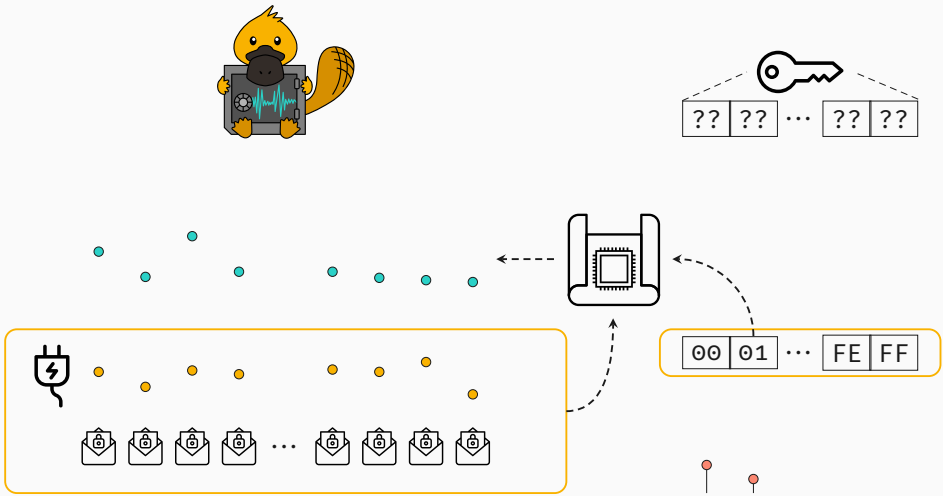




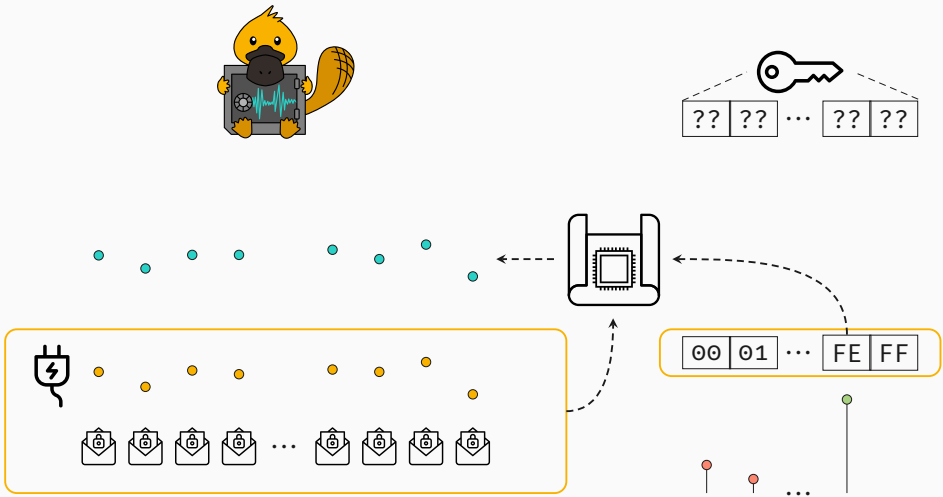
CPA Attack



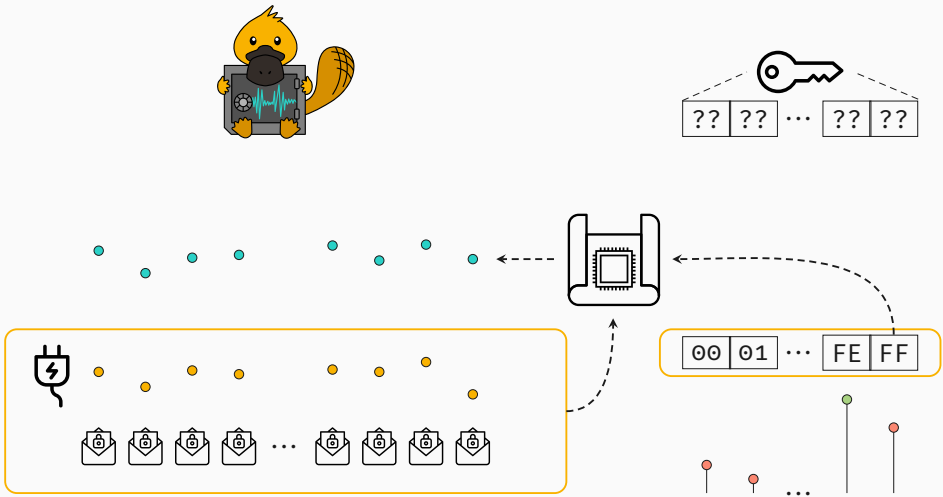
CPA Attack



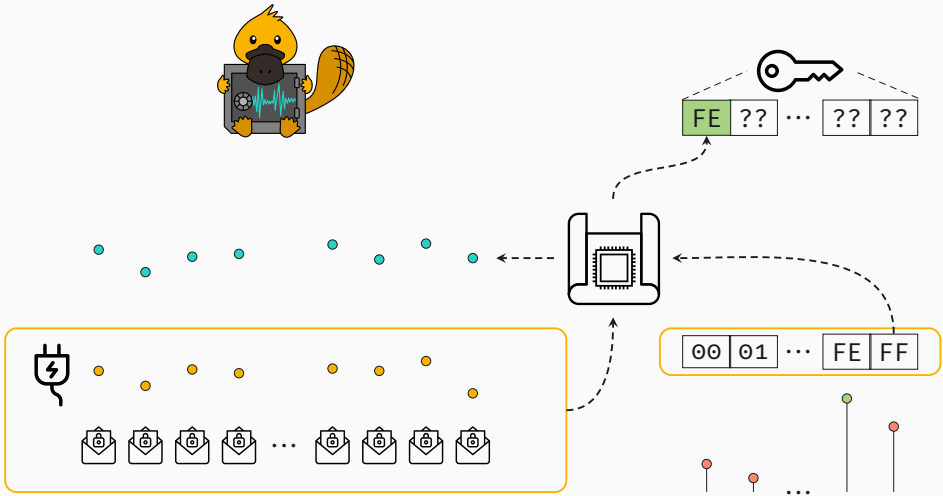
CPA Attack



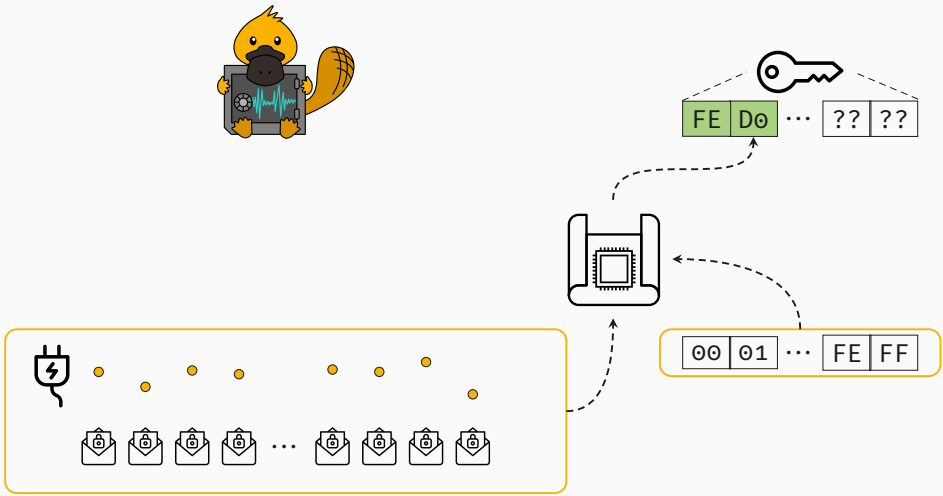
CPA Attack



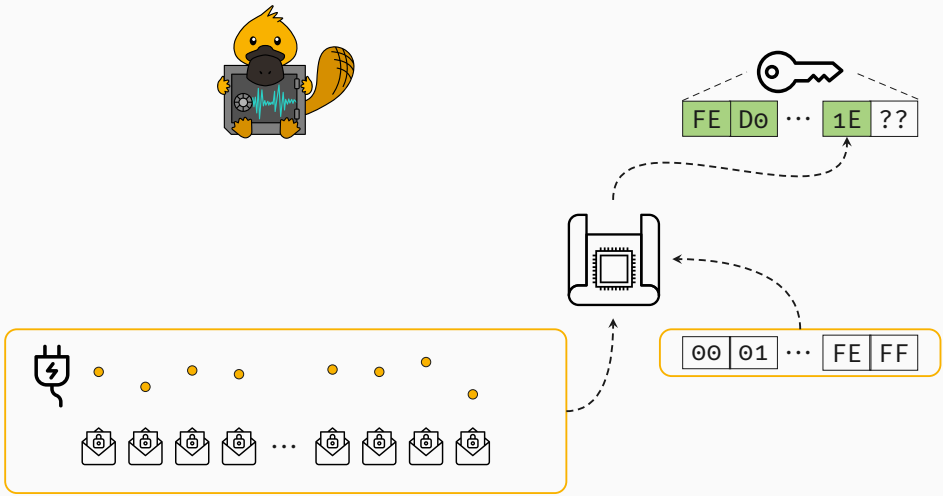
CPA Attack



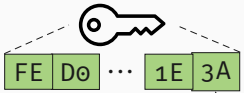
CPA Attack

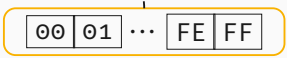


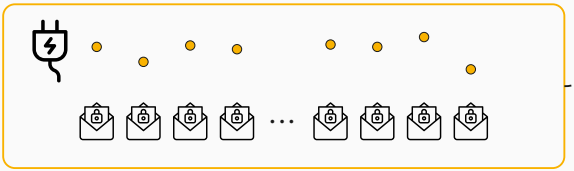
CPA Attack



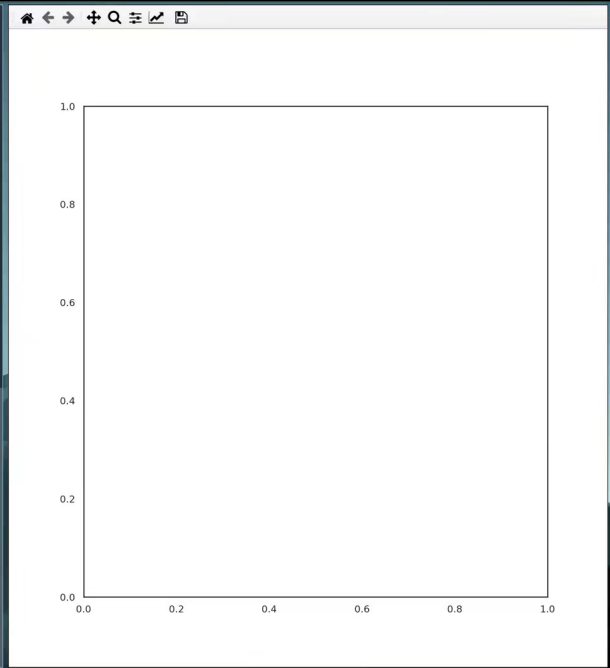
CPA Attack







```
mlq@dreadnought ~/platypus-aesni % ./cpa -f . -c 2000000 -m 4 -n
```



Other CPU Vendors

- AMD affected as well

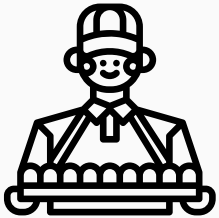
Other CPU Vendors

- AMD affected as well
- Never heard back after disclosure

Other CPU Vendors

- AMD affected as well
- Never heard back after disclosure
- Similar Linux patch as Intel

Other CPU Vendors



Other CPUs also have interfaces for measuring power

- Some ARM development boards (odroid XU+E, SAML11)
- NVIDIA Jetson TX2
- IBM POWER9
- Marvell ThunderX2
- Ampere Altra
- Hygon Dhyana CPU family

Countermeasures

Countermeasures

- Remove the unprivileged access to the RAPL MSRs

Countermeasures

- Remove the unprivileged access to the RAPL MSRs
- **1 Line Patch** for the Linux Kernel

Countermeasures

- Threat model of SGX allows a **compromised operating system**

Countermeasures

- Threat model of SGX allows a **compromised operating system**
  - Operating system patch does not help

Countermeasures

- Threat model of SGX allows a **compromised operating system**
  - Operating system patch does not help
- **Microcode updates** are necessary

Countermeasures

- Threat model of SGX allows a **compromised operating system**
  - Operating system patch does not help
- **Microcode updates** are necessary
  - Fallback to a model of the energy consumption

Countermeasures

- Threat model of SGX allows a **compromised operating system**
  - Operating system patch does not help
- **Microcode updates** are necessary
  - Fallback to a model of the energy consumption
  - Does **not allow** to distinguish data/operands any more

Countermeasures

- Threat model of SGX allows a **compromised operating system**
  - Operating system patch does not help
- **Microcode updates** are necessary
  - Fallback to a model of the energy consumption
  - Does **not allow** to distinguish data/operands any more
  - **Constant-time implementations** are necessary

Takeaways

- Power side-channel attacks can be exploited **from software** on modern CPUs

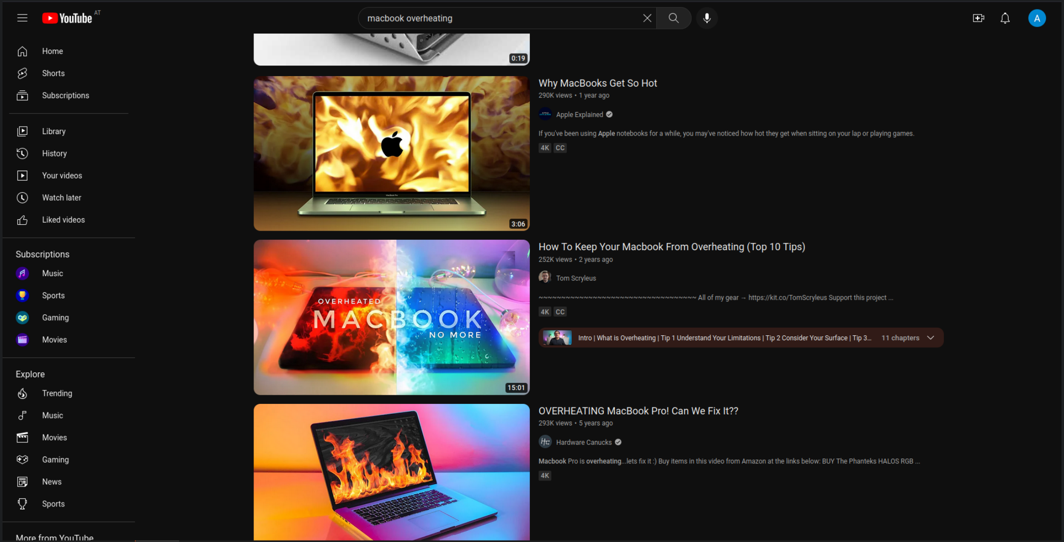
Takeaways

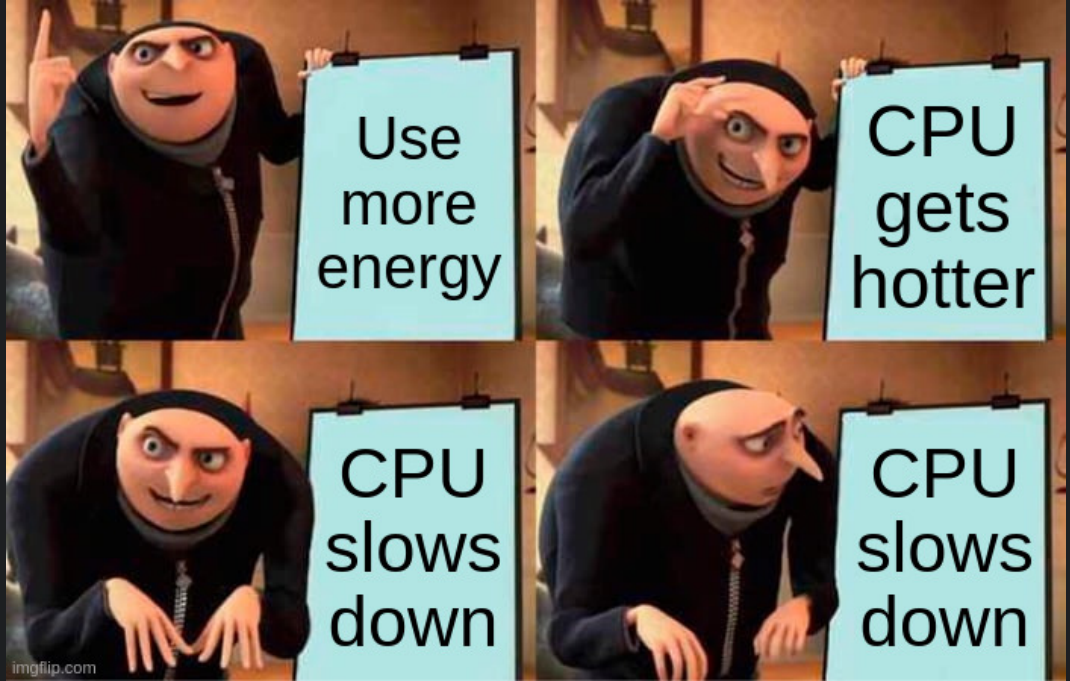
- Power side-channel attacks can be exploited **from software** on modern CPUs
- Threat model of Intel SGX requires more complex mitigations

Takeaways

- Power side-channel attacks can be exploited **from software** on modern CPUs
- Threat model of Intel SGX requires more complex mitigations
- Other CPU manufacturers provide similar interfaces

The End?


YouTube AT
macbook overheating
Home
Shorts
Subscriptions
Library
History
Your videos
Watch later
Liked videos
Subscriptions
Music
Sports
Gaming
Movies
Explore
Trending
Music
Movies
Gaming
News
Sports
More from YouTube
0:19
Why MacBooks Get So Hot
290K views • 1 year ago
Apple Explained
If you've been using Apple notebooks for a while, you may've noticed how hot they get when sitting on your lap or playing games.
4K CC
3:06
How To Keep Your Macbook From Overheating (Top 10 Tips)
252K views • 2 years ago
Tom Scryleus
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ All of my gear → https://kit.co/TomScryleus Support this project ...
4K CC
Intro | What is Overheating | Tip 1 Understand Your Limitations | Tip 2 Consider Your Surface | Tip 3...
11 chapters
OVERHEATED
MACBOOK
NO MORE
15:01
OVERHEATING MacBook Pro! Can We Fix It??
293K views • 5 years ago
Hardware Canucks
Macbook Pro is overheating...lets fix it :) Buy items in this video from Amazon at the links below: BUY The Phanteks HALOS RGB ...
4K
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

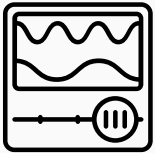

Use more energy
CPU gets hotter
CPU slows down
CPU slows down
imgflip.com

Remember?

- CPU power management is complex
- In order to save power, you can …

Shut down resources

Reduce voltage



Reduce frequency

Hertzbleed

- The Hertzbleed attack from Wang et al. shows:
- If more **energy** is used

Hertzbleed

- The Hertzbleed attack from Wang et al. shows:
- If more **energy** is used
- The CPU gets **hotter**

Hertzbleed

- The Hertzbleed attack from Wang et al. shows:
- If more **energy** is used
- The CPU gets **hotter**
- Until the frequency is no longer sustainable

Hertzbleed

- The Hertzbleed attack from Wang et al. shows:
- If more **energy** is used
- The CPU gets **hotter**
- Until the frequency is no longer sustainable

→ The runtime of the executed code slows down

Hertzbleed

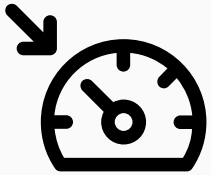
- The Hertzbleed attack from Wang et al. shows:
- If more **energy** is used
- The CPU gets **hotter**
- Until the frequency is no longer sustainable

→ The runtime of the executed code slows down

- Measure with fixed clock, e.g., `rdtsc`

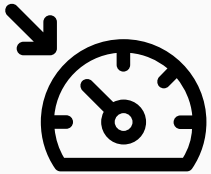

WAITING FOR THE CPU TO OVERHEAD
imgflip.com

How to Throttle the CPU



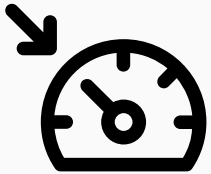
- RAPL provides energy **limits**

How to Throttle the CPU



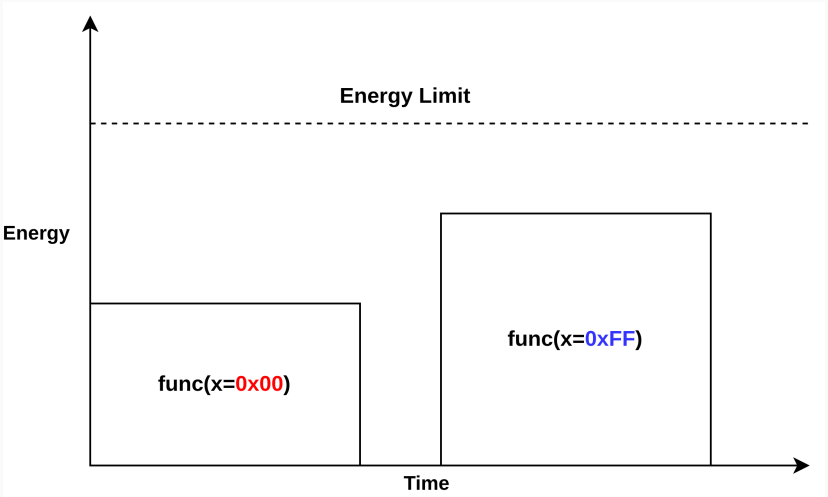
- RAPL provides energy **limits**
  - If exhausted CPU throttles the frequency
- Run **Stress** on the system

How to Throttle the CPU

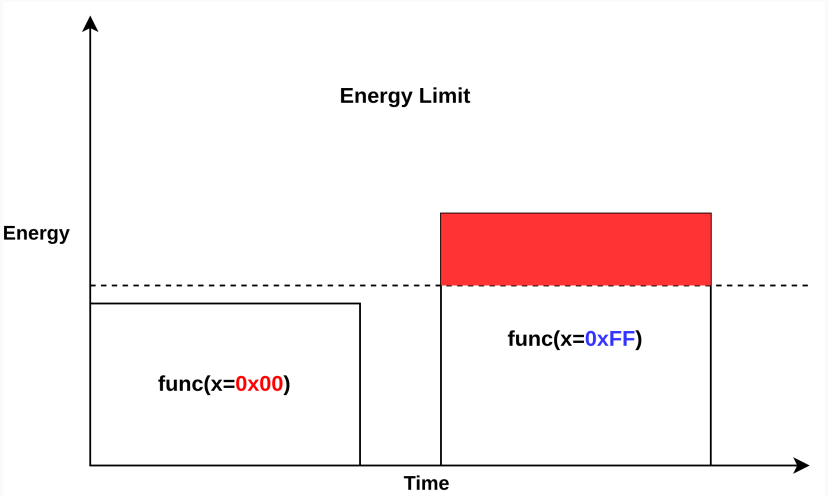


- RAPL provides energy **limits**
  - If exhausted CPU throttles the frequency
- Run **Stress** on the system
  - CPUs start throttling when using many threads

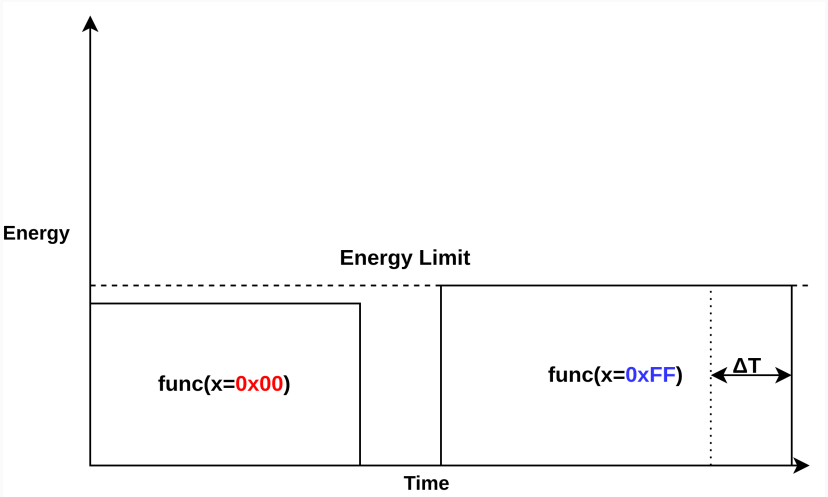
Converting Energy Differences



Convert Energy Differences



Convert Energy Differences




NYPD
POWER
TIME
imgflip.com

What can we do with this?

Covert Channel

- Hidden communication channel

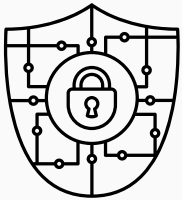
Covert Channel

- Hidden communication channel
- No power interface required

Covert Channel

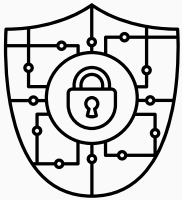
- Hidden communication channel
- No power interface required
- Time/Frequency measurements proxy power interface

Attacking Crypto



- AES Correlation Power Analysis
  - Measure execution time of AES encryptions

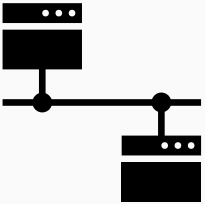
Attacking Crypto



- AES Correlation Power Analysis
  - Measure execution time of AES encryptions

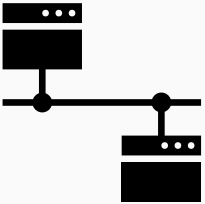
  $\rightarrow$ Apply CPA technique to recover key

Remote Attack



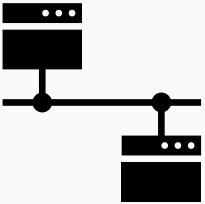
- Remote attacker requests service from server

Remote Attack



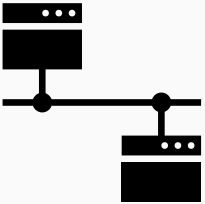
- Remote attacker requests service from server
  - Cryptographic operation, i.e. encryption, signature

Remote Attack



- Remote attacker requests service from server
  - Cryptographic operation, i.e. encryption, signature
- Server computes respose using secret

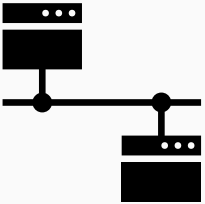
Remote Attack



- Remote attacker requests service from server
  - Cryptographic operation, i.e. encryption, signature
- Server computes respose using secret

→ **Hertzbleed-effect** influences response times

Remote Attack

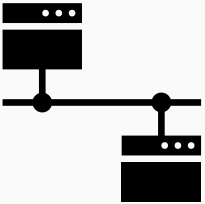


- Remote attacker requests service from server
  - Cryptographic operation, i.e. encryption, signature
- Server computes respose using secret

→ **Hertzbleed-effect** influences response times

- **Calculations using secret** influences server CPU frequency

Remote Attack



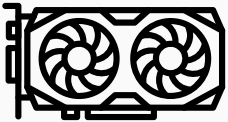
- Remote attacker requests service from server
  - Cryptographic operation, i.e. encryption, signature
- Server computes respose using secret

→ **Hertzbleed-effect** influences response times

- **Calculations using secret** influences server CPU frequency

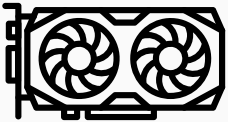
→ Attacker recovers secret using collected timings

GPU Throttling



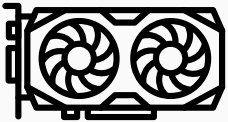
- **Integrated** GPUs share power limits with the CPU

GPU Throttling



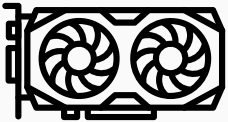
- **Integrated** GPUs share power limits with the CPU
  → **CPU throttling** indicates high GPU consumption

GPU Throttling



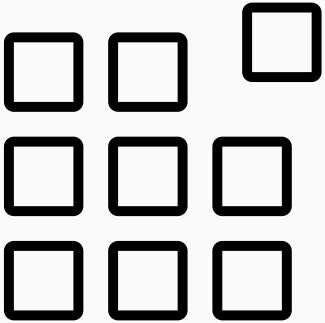
- **Integrated** GPUs share power limits with the CPU
  - → **CPU throttling** indicates high GPU consumption
- **Dedicated** GPUs have power limits too

GPU Throttling



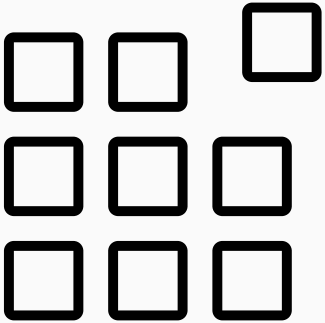
- **Integrated** GPUs share power limits with the CPU
  - → **CPU throttling** indicates high GPU consumption
- **Dedicated** GPUs have power limits too
  - → **Observable** by timing a GPU workload

Pixel Stealing



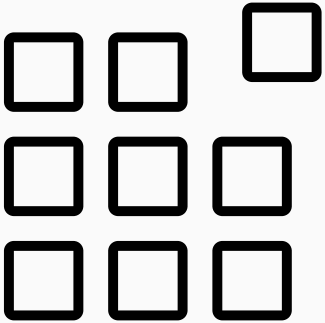
- What secrets are *“inside”* a GPU?

Pixel Stealing



- What secrets are *“inside”* a GPU?
  - GPU renders windows and screen

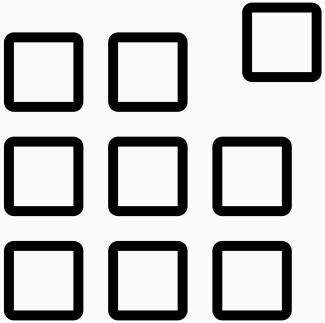
Pixel Stealing



- What secrets are *“inside”* a GPU?
  - GPU renders windows and screen

  → **Privacy** related information

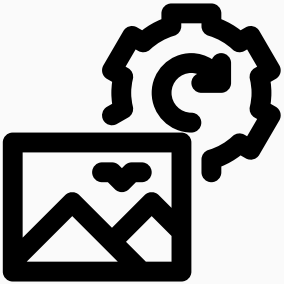
Pixel Stealing



- What secrets are *“inside”* a GPU?
  - GPU renders windows and screen

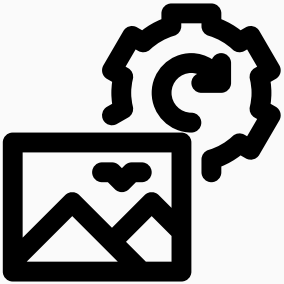
  $\rightarrow$ **Privacy** related information
- **Pixel** color represents the information

Pixel Stealing



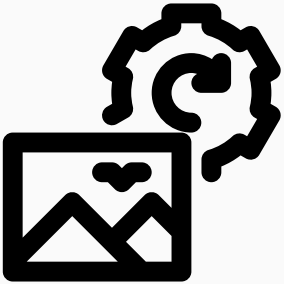
- **Post-processing** without revealing the pixels

Pixel Stealing



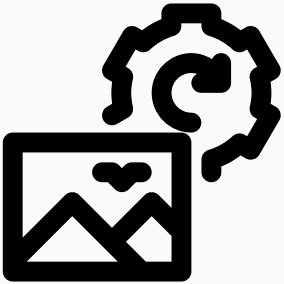
- **Post-processing** without revealing the pixels
- Pixel value is the **data operand**

Pixel Stealing



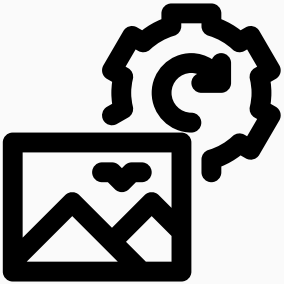
- **Post-processing** without revealing the pixels
- Pixel value is the **data operand**
- Distinguishable power consumption

Pixel Stealing



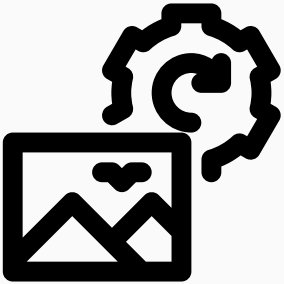
- **Post-processing** without revealing the pixels
- Pixel value is the **data operand**
- Distinguishable power consumption
  - **Bright** pixel $\rightarrow$ less power

Pixel Stealing



- **Post-processing** without revealing the pixels
- Pixel value is the **data operand**
- Distinguishable power consumption
  - **Bright** pixel $\rightarrow$ less power
  - **Dark** pixel $\rightarrow$ more power

Pixel Stealing

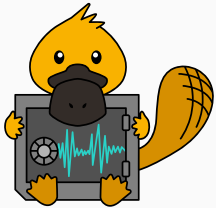


- **Post-processing** without revealing the pixels
- Pixel value is the **data operand**
- Distinguishable power consumption
  - **Bright** pixel → less power
  - **Dark** pixel → more power

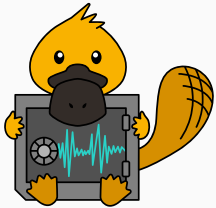
→ Measure timing and infer pixel value

How can we transform power side channels towards a broader scope?

Motivation

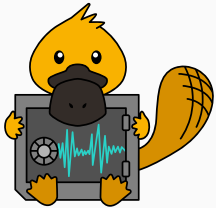


Motivation



Software-based Power Side Channels

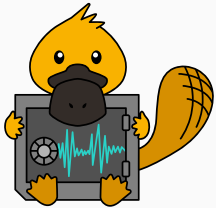
Motivation



Software-based Power Side Channels

- **Specific** targets: Algorithms

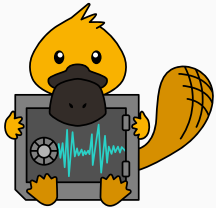
Motivation



Software-based Power Side Channels

- **Specific** targets: Algorithms
- Leak edge cases

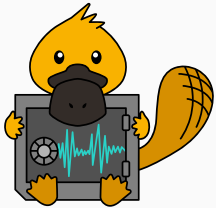
Motivation



Software-based Power Side Channels

- **Specific** targets: Algorithms
- Leak edge cases
- **Limited** to a side channel

Motivation

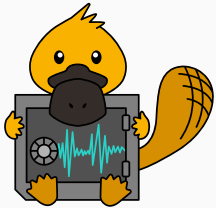


Software-based Power Side Channels

- **Specific** targets: Algorithms
- Leak edge cases
- **Limited** to a side channel

Transient Execution Attacks

Motivation



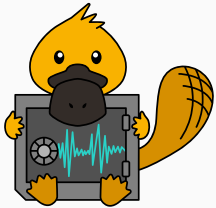
Software-based Power Side Channels

- **Specific** targets: Algorithms
- Leak edge cases
- **Limited** to a side channel

Transient Execution Attacks

- **Generic** targets: CPU components

Motivation



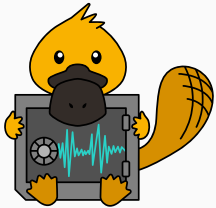
Software-based Power Side Channels

- **Specific** targets: Algorithms
- Leak edge cases
- **Limited** to a side channel

Transient Execution Attacks

- **Generic** targets: CPU components
- Leak arbitrary data

Motivation



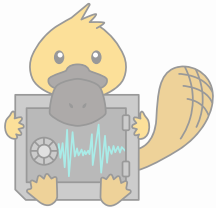
Software-based Power Side Channels

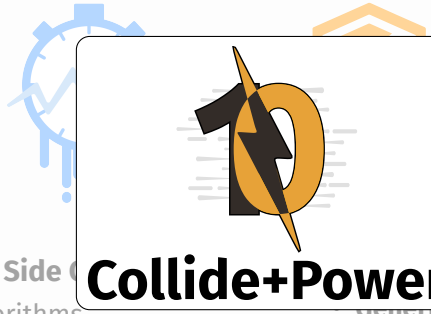
- **Specific** targets: Algorithms
- Leak edge cases
- **Limited** to a side channel

Transient Execution Attacks

- **Generic** targets: CPU components
- Leak arbitrary data
- **Agnostic** to side channels

Motivation





Software-based Power Side C

- **Specific** targets: Algorithms
- Leak edge cases
- **Limited** to a side channel

ecution Attacks

Generic targets: CPU components

- Leak arbitrary data
- **Agnostic** to side channels

Collide+Power

Collide+Power - Overview

- **Collide+Power** exploits leakage between:

Collide+Power - Overview

- **Collide+Power** exploits leakage between:
  - **Guess $\mathcal{G}$:** Attacker-controlled data

Collide+Power - Overview

- **Collide+Power** exploits leakage between:
  - **Guess** $\mathcal{G}$**:** Attacker-controlled data
  - **Value** $\mathcal{V}$**:** Victim secret data

Collide+Power - Overview

- **Collide+Power** exploits leakage between:
  - **Guess** $\mathcal{G}$**:** Attacker-controlled data
  - **Value** $\mathcal{V}$**:** Victim secret data

💡 Hamming distance: $\mathsf{hd}(\mathcal{G}, \mathcal{V})$

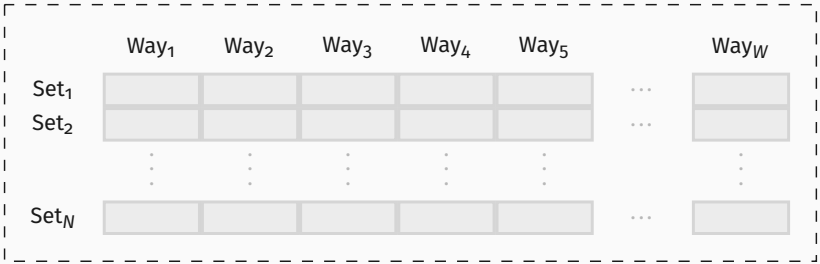
Collide+Power - Overview

- **Collide+Power** exploits leakage between:
  - **Guess** $\mathcal{G}$**:** Attacker-controlled data
  - **Value** $\mathcal{V}$**:** Victim secret data

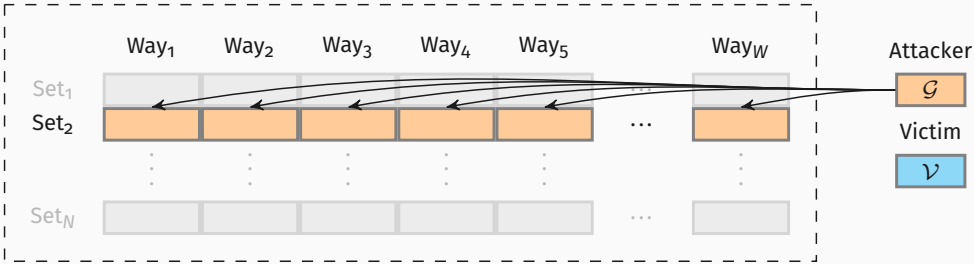
💡 Hamming distance: $\mathsf{hd}(\mathcal{G}, \mathcal{V})$

$\rightarrow$ How to exploit this limited information?

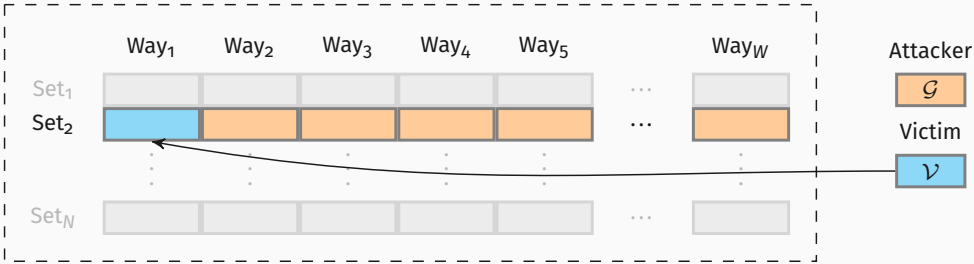
Collide+Power - Memory Subsystem



Collide+Power - Memory Subsystem



Collide+Power - Memory Subsystem



Power Leakage - Model Components

Power Leakage - Model Components

Hamming Weight: $\mathrm{hw}(x)$

Power Leakage - Model Components

Hamming Weight: $\mathrm{hw}(x)$

Number of set bits

Power Leakage - Model Components

Hamming Weight: $\mathrm{hw}(x)$

Number of set bits

$\mathrm{hw}(11_2) = 2$

Power Leakage - Model Components

Hamming Weight: $\mathrm{hw}(x)$

Number of set bits

$\mathrm{hw}(11_2) = 2$

Hamming Distance: $\mathrm{hd}(x, y)$

Power Leakage - Model Components

Hamming Weight: $\mathrm{hw}(x)$

Number of set bits

$\mathrm{hw}(11_2) = 2$

Hamming Distance: $\mathrm{hd}(x, y)$

Number of different bits

Power Leakage - Model Components

Hamming Weight: $\mathrm{hw}(x)$

Number of set bits

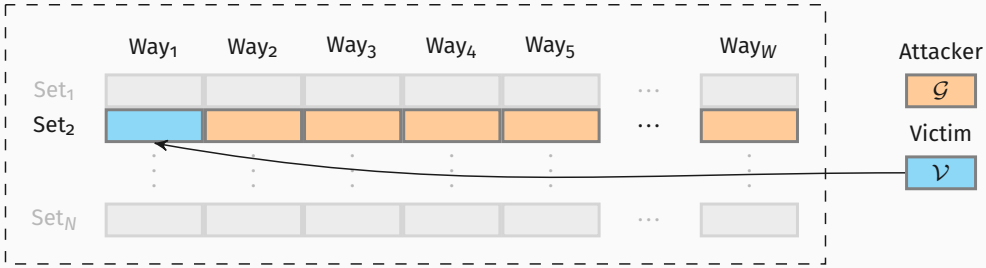
$\mathrm{hw}(11_2) = 2$

Hamming Distance: $\mathrm{hd}(x, y)$

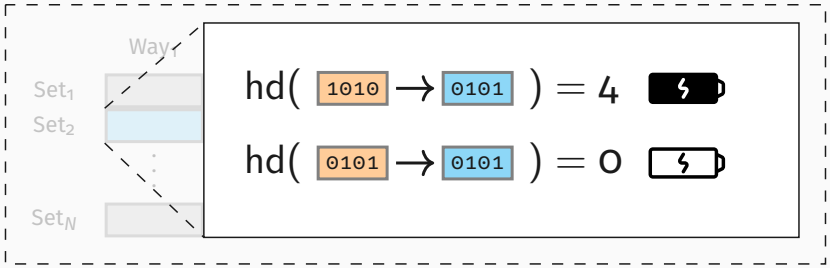
Number of different bits

$\mathrm{hd}(11_2, 01_2) = 1$

Collide+Power - Memory Subsystem



Collide+Power - Memory Subsystem



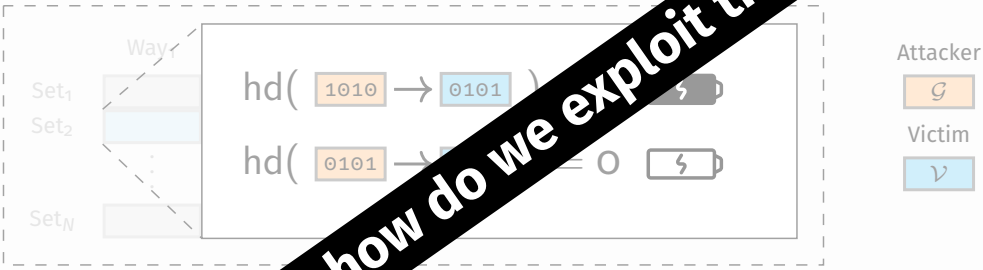
Attacker

$\mathcal{G}$

Victim

$\mathcal{V}$

Collide+Power - Memory Subsystem



But how do we exploit this?

Collide+Power - Intuition

$$\mathcal{P}(\mathcal{G}, \mathcal{V}) \approx \dots$$

Collide+Power - Intuition

$$\mathcal{P}(\mathcal{G}, \mathcal{V}) \approx \mathsf{hd}(\mathcal{G}, \mathcal{V})$$

Collide+Power - Intuition

$$\mathcal{P}(\mathcal{G}, \mathcal{V}) \approx \mathsf{hd}(\mathcal{G}, \mathcal{V})$$

Collide+Power - Intuition

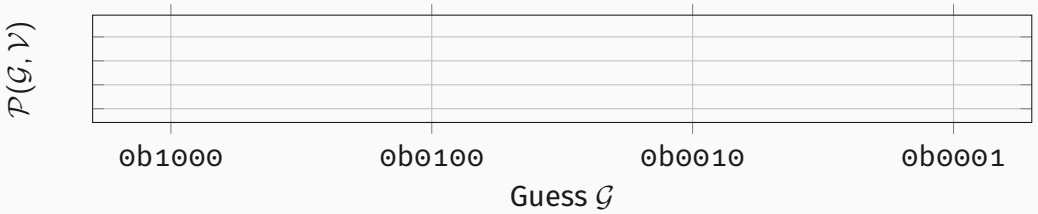
$$\underbrace{\mathcal{P}(\mathcal{G}, \mathcal{V})}_{\text{model}} \approx \mathsf{hd}(\mathcal{G}, \mathcal{V})$$

Collide+Power - Intuition

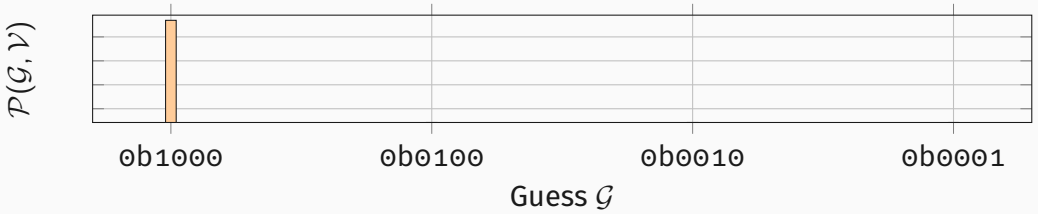
$$\underbrace{\mathcal{P}(\mathcal{G}, \mathcal{V})}_{\text{model}} \approx \underbrace{\mathsf{hd}(\mathcal{G}, \mathcal{V})}_{\text{signal}}$$

Collide+Power - Example

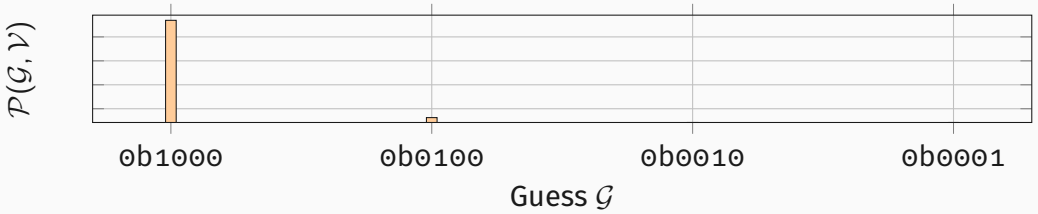
$$\mathcal{P}(\mathcal{G}, 0101_2) \approx \mathrm{hd}(\mathcal{G}, 0101_2)$$



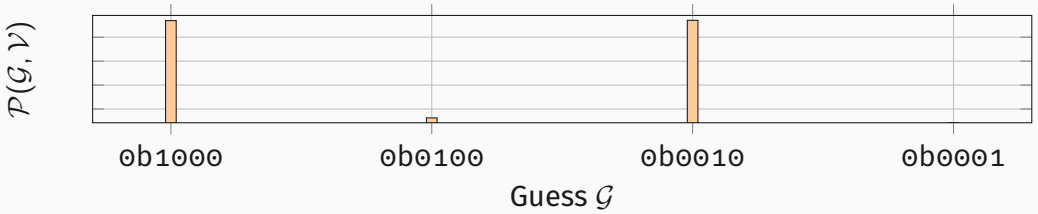
$$\mathcal{P}(1000_2, 0101_2) \approx \mathrm{hd}(\mathbf{1}000_2, \mathbf{0}101_2) = 3$$



$$\mathcal{P}(0100_2, 0101_2) \approx \mathrm{hd}(0\mathbf{1}00_2, 0\mathbf{1}01_2) = 1$$

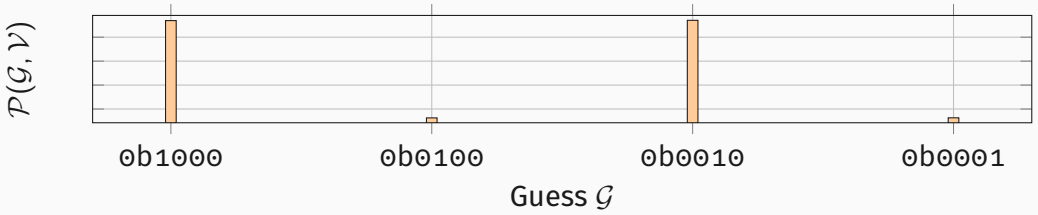


$$\mathcal{P}(0010_2, 0101_2) \approx \mathrm{hd}(00\mathbf{1}0_2, 01\mathbf{0}1_2) = 3$$



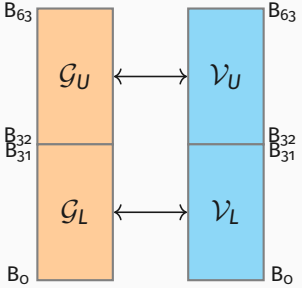
Collide+Power - Example

$$\mathcal{P}(0001_2, 0101_2) \approx \mathrm{hd}(000\mathbf{1}_2, 010\mathbf{1}_2) = 1$$



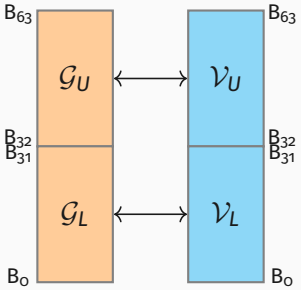
Leakage Analysis - Generalization

Aligned Leakage

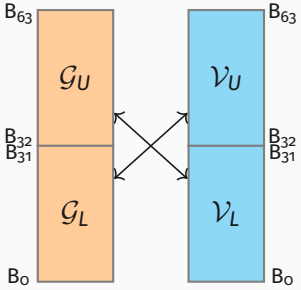


Leakage Analysis - Generalization

Aligned Leakage

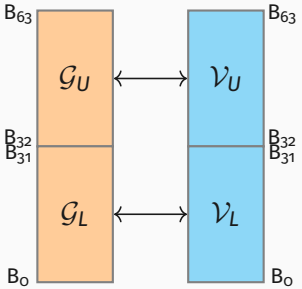


Cross Leakage

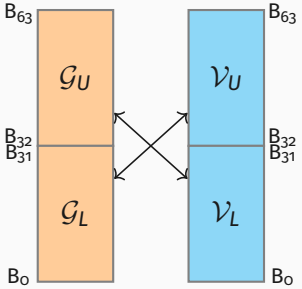


Leakage Analysis - Generalization

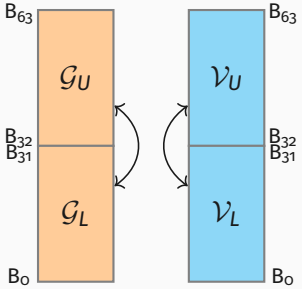
Aligned Leakage



Cross Leakage



Self Leakage



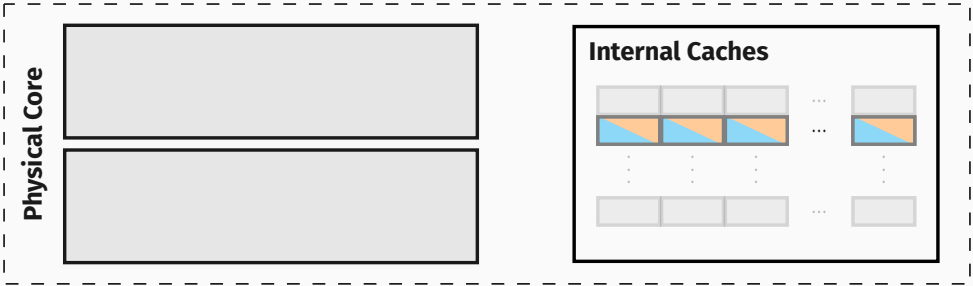
Leakage Analysis: Results

Inst.	Evict.	Effectiveness		Aligned Leakage		Cross Leakage		Self Leakage		Weights			
		$\hat{\rho}$	SNR_A	$\mathrm{hd}(v_L, g_L)$	$\mathrm{hd}(v_U, g_U)$	$\mathrm{hd}(v_L, g_U)$	$\mathrm{hd}(v_U, g_L)$	$\mathrm{hd}(v_L, v_U)$	$\mathrm{hd}(g_L, g_U)$	$\mathrm{hw}(v_L)$	$\mathrm{hw}(v_U)$	$\mathrm{hw}(g_L)$	$\mathrm{hw}(g_U)$
		$\cdot 1$	$\cdot 10^{-3}$	a_0 in μW	a_1 in μW	c_0 in μW	c_1 in μW	s_0 in μW	s_1 in μW	w_0 in μW	w_1 in μW	w_2 in μW	w_3 in μW
Load	None	0.311	72.004	544.5	4.2	1.1	0.5	0.0			0.0	362.6	0.0
	L1	0.907	7.873	598.3	278.8	0.0	0.0				0.0	6124.4	2696.9
	L1+L2	0.822	5.632	339.3	141.7	106.6					0.0	3750.7	1435.0
Prefetch	None	0.003	0.000	0.0					0.0	0.0	0.0	1.7	2.8
	L1	0.370	11.365	136				0.0	0.0	0.0	0.0	454.1	455.5
	L1+L2	0.300	5.294				43.0	0.0	0.0	0.0	0.0	334.0	332.5
Store	None	0.003	0.000		0.0	0.0	3.1	0.0	0.0	0.0	0.0	7.0	0.0
	L1	0.241	3.876	93.3	74.5	4.9	9.6	0.0	0.0	0.0	0.0	204.6	303.2
	L1+L2	0.450	6.457	133.7	169.0	84.7	86.2	0.0	0.0	0.0	0.0	347.1	1130.5

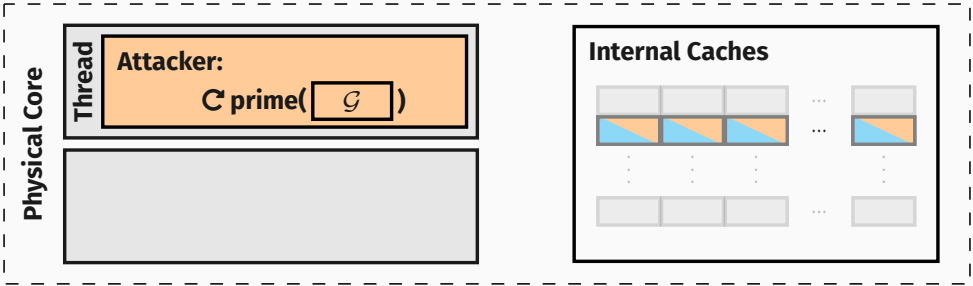
Do not start reading this!

Generic Attacks

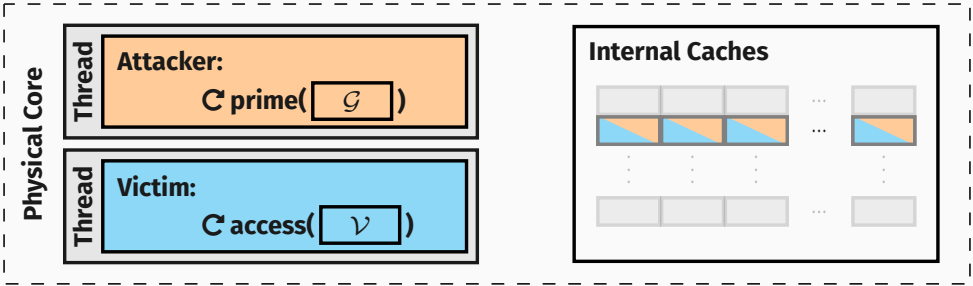
MDS-style Attack



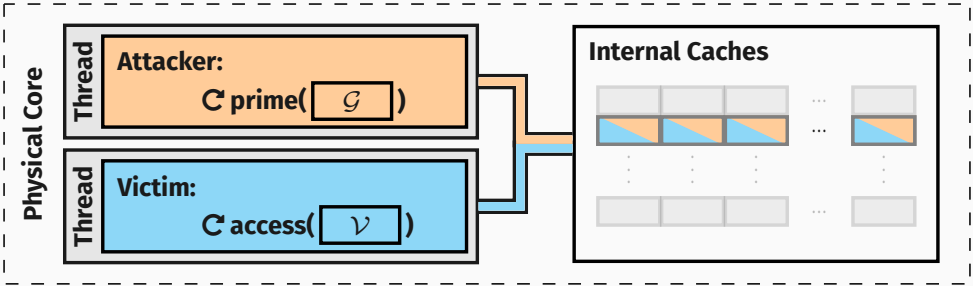
MDS-style Attack



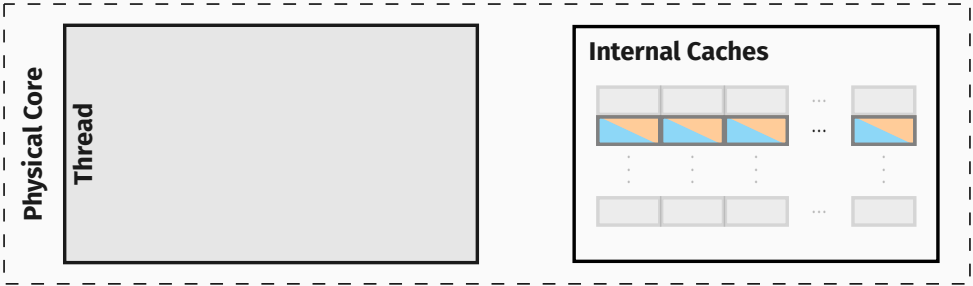
MDS-style Attack



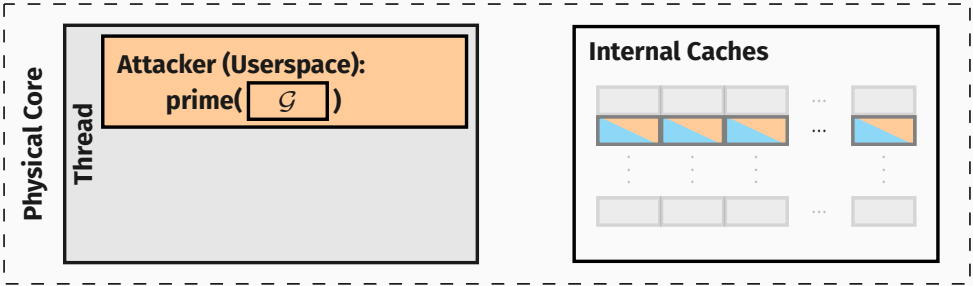
MDS-style Attack



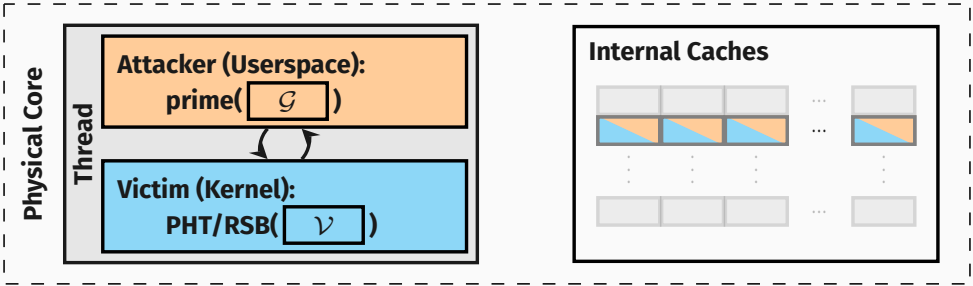
Meltdown-style Attack



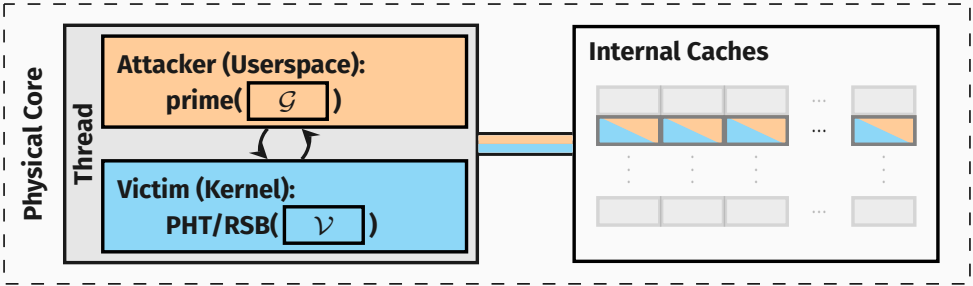
Meltdown-style Attack



Meltdown-style Attack



Meltdown-style Attack



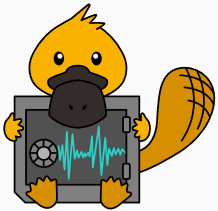
This must be slow?

NO!

It is EXTREMELY slow![1]

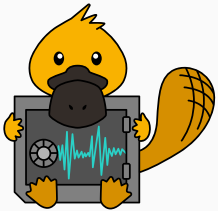
[1] With the current state-of-the-art.

Software-based Power Side Channels



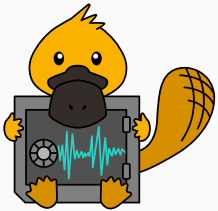
- **MDS-style:**
  4.82 bit/h

Software-based Power Side Channels



- **MDS-style:**
  4.82 bit/h
- **Meltdown-style (RSB):**
  0.84 bit/h

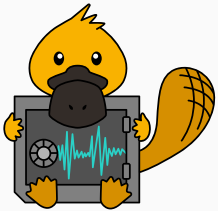
Software-based Power Side Channels



- **MDS-style:**
  4.82 bit/h
- **Meltdown-style (RSB):**
  0.84 bit/h

- **MDS-style:**
  0.065 to 0.68 bit/h

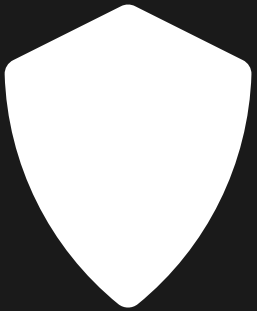
Software-based Power Side Channels



- **MDS-style:**
  4.82 bit/h
- **Meltdown-style (RSB):**
  0.84 bit/h

- **MDS-style:**
  0.065 to 0.68 bit/h
- **Meltdown-style estimate (PHT):**
  99.95 days/bit to 2.86 years/bit

Mitigations



Mitigations

- **Preventing data collisions:**

Mitigations

- **Preventing data collisions:**
  - **Redesign** of the complete shared data path

Mitigations

- **Preventing data collisions:**
  - **Redesign** of the complete shared data path
  - **Costly** to deploy

Mitigations

- **Preventing data collisions:**
  - **Redesign** of the complete shared data path
  - **Costly** to deploy
  - **Missed** components re-enable Collide+Power

Mitigations

- **Preventing observable power consumption:**
  - **Restricting** all direct power interfaces

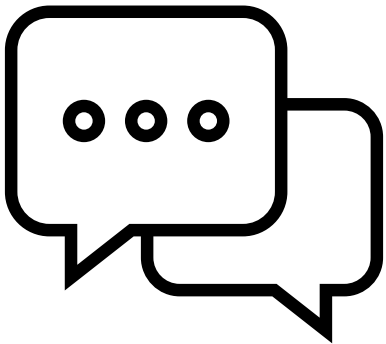
Mitigations

- **Preventing observable power consumption:**
  - **Restricting** all direct power interfaces
- **Mitigating** Hertzbleed is challenging
  - Thermal and power management is required

Mitigations

- **Preventing observable power consumption:**
  - **Restricting** all direct power interfaces
- **Mitigating** Hertzbleed is challenging
  - Thermal and power management is required

→ **Collide+Power** is slow but unmitigated on modern CPUs!



Questions?