BlueHat IL

Ezra Caltum & Anders Fogh

Life and death of a hardware security bug

Life and death of a hardware security bug

Preventing, finding, mitigating CPU bugs

BlueHat IL

Disclaimer

```
0x66 0x0F 0x1F 0x84 0x00 0x00 0x00 0x00 0x00
```

This is a complex, out of order talk. The activities described in this talk, didn't necessarily took place in the order we are presenting it, rather in the order we speculate will be easier to understand.

We are going to branch in and out of different topics, while trying to cover as much material as possible.

We predict that we won't be able to answer all your questions. There are certain topics we don't have the privilege to talk here, and we don't want to be at fault of disclosing privileged or proprietary information.

.INTRODUCTION

0x9a 0xcd

BlueHat IL

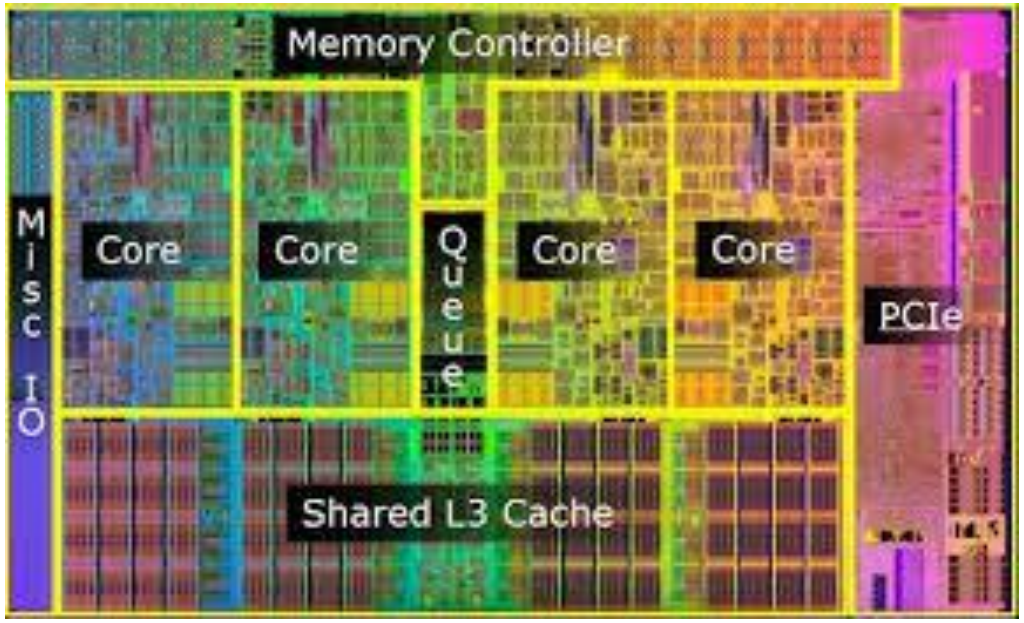
CPU

High level:

- CPUs are now adays a part of a System On a Chip (SoC)
- CPUs are hardware, expected to support an ISA (the specification approved between developers and CPU makers on the Architecture Definition), with a support level.
- Deprecation is difficult
- When the specification is broken, an erratum is born.

uArch

- Core
  - Computation unit
- SoC
  - Support systems
    - Power management
    - Memory
    - PCI
    - IO
    - And lots more



What is a bug

Functional bug/Wrong Result - Erratum

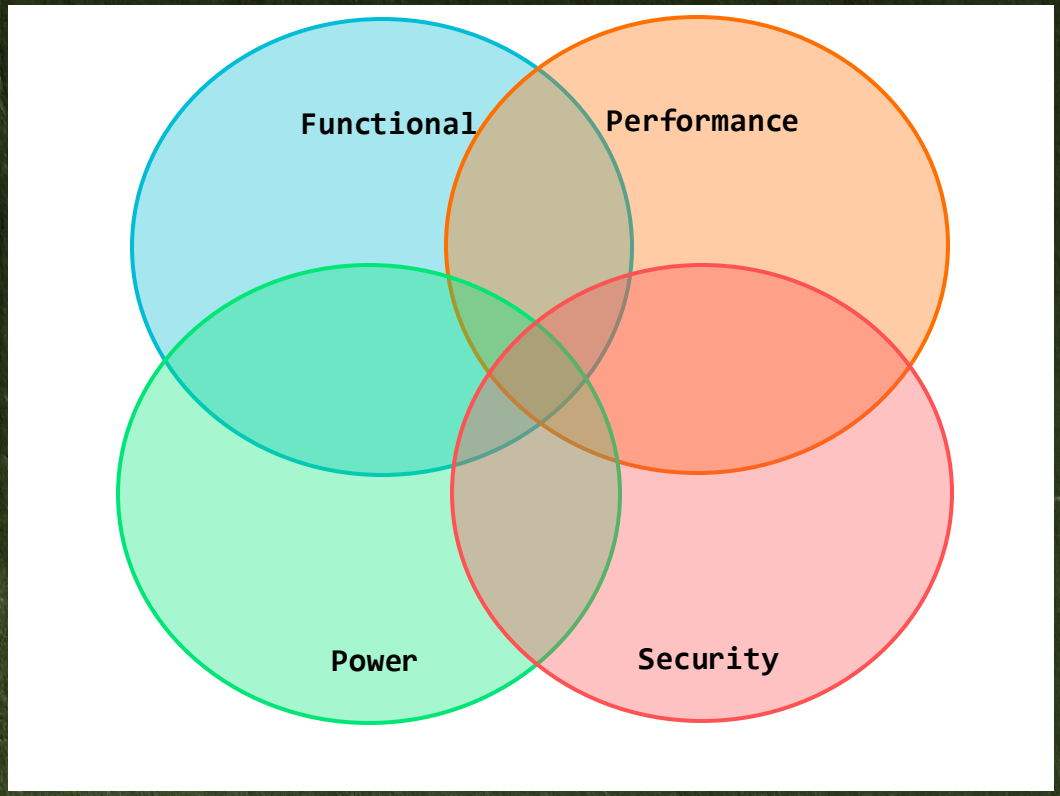
Performance bug - Too slow

Power bug – Unexpected power consumption

Security issues – CVE

Out of scope

What is the scope?



How we prevent bugs?

Internal findings before tape-in

- Fuzzers/Test Generators.
- Custom Static Analysis tools.
- “Asserts”.
- Coverage Testing.
- “Creativity”
- Formal
- Hardware “traces” - What happened at gate level.

https://www.intel.com/content/www/us/en/content-details/915022/2026-intel-platform-security-report.html

Fuzzers, hardware, reality

We use fuzzers on CPUs

- They are useful to find shallow variants.

However, it's not as useful as for software

- Reason 1: The state space is huge
  - Imagine fuzzing a 64bit multiplication: For each sand corn on earth you'd have a billion trillion input pairs.
  - It's a small thing for formal verification and perfect coverage.
- Reason 2: Hardware (silicon) is fast, but simulation is slow
  - Hardware (silicon) doesn't provide much data for understanding
  - Simulation (pre-silicon) does provide all the data you need*, but it is the slowest method.

Frequent pitfall: We build a fuzzer for a reduced state space. Only to find out you can brute force the entirety.

RET

0xCB

.Plundervolt

0x9a 0xcd

BlueHat IL

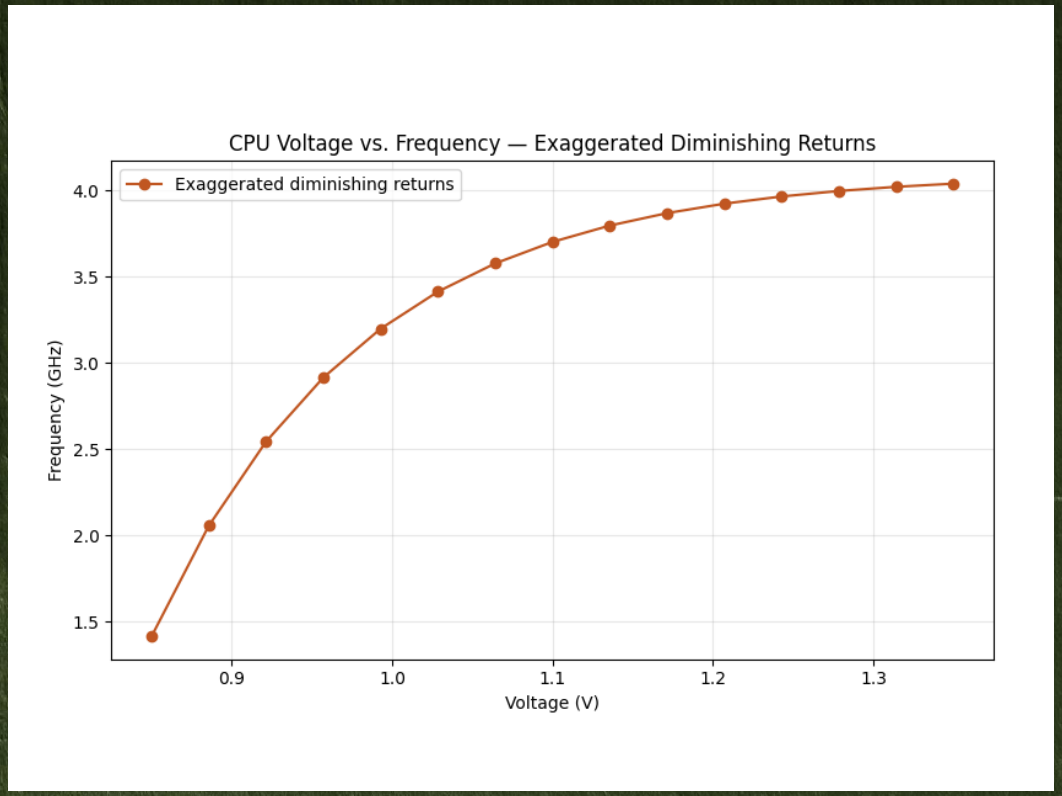
Background

Critical path: The longest sequence of gates a CPU is designed to run in a half cycle.

Two main factors determines critical path:

- Frequency
- Voltage

The diagram shows possible sensisble critical path for a choice of frequency and voltage



BlueHat IL

June 2019

BlueHat IL

Plundervolt – The report.

During June 2019, Intel PSIRT received an interesting report.

Intel had a Voltage knob in an MSR on the CPU for over clocking

- Increase the voltage => increase the frequency
- Decrease the voltage => decrease power consumption/temperature

Decreasing the voltage, without adjusting frequency can result in the critical path being overshot and bit flips could occur.

Under certain circumstances (like cryptography) this bit flips can lead us to leak confidential information.

- If you want to learn more about this, watch their talk.

Not a bug – used by gamers & overclockers

The knob is controlled by Kernel/VMM

- However: SGX doesn't trust VMM/Kernel

.SGX_INTRO

0x9a 0xcd

BlueHat IL

Intel SGX

Intel Software Guard Extensions (SGX) is a set of instruction codes implementing trusted execution environment that are built into some Intel central processing units (CPUs). They allow user-level and operating system code to define protected private regions of memory, called enclaves. This enclaves shield activity from malicious actors, including those that have full ring0 privileges.*

SGX has multiple components – which are not the focus of this talk.

* Doesn't apply to Physical attacks

ret

0xCB

Plundervolt - Triage

We had to reproduce this report… Not only for the reported assembly instructions, but for all the instructions.

Further, researchers reported that different inputs might produce different bit flips, depending on the value.

With voltage outside spec, every mistake can crash the system.

Lab systems are usually air gapped, and connection is done with jumpboxes.

The solution:

Reverse engineer smart “power strips” to build a heartbeat on the machines that automatically resumes from last executed command.

Plundervolt – how to fix?

How to mitigate?

- Simple – for “semi” future projects, make some checks that verify the status of this MSR for SGX.
- For longer term support – We introduced UVP on Gen 12 processors and newer (https://www.intel.com/content/www/us/en/support/articles/000094219/processors.html)

But what about what is already deployed?

.Mitigations

0x9a 0xcd

BlueHat IL

Quick Introduction to “mitigating vulnerabilities”

Hardware engineering is complex. Further – when you ship hardware, you want to be resilient. Disabling a feature in production is not like removing the piece of software. If the HW doesn’t work , it doesn’t work.

So we are “chickens” – really we are.

That’s why we create “chicken bits” so we can “chicken out” and disable the feature.

Not all features can be disabled.

Chicken bits

* No animals were harmed in the creation of you favorite CPU.

BlueHat IL

uCode

Microcode is the most used firmware for mitigations (but we have other tools in the arsenal)
Microcode lives in the MSROM and is injected in the front end of the pipeline
To be able to patch, we also have an MSRAM
We can rewrite existing flows
This real estate is some of the most expensive real estate in the world.

uCode limitations

We are limited to existing uOps

There are space limitations

- Mitigations are expensive in terms of on chip resources.

It often comes with trade offs

You can only do so much in uCode

- Despite rumours that a uCode patch is capable of doing magic

What is a uCode mitigation constraint

Mitigation can impact “hot” code (performance impact).

The size of the mitigation is over the projected project limits.

uCode not always has control over “microarchitectural” hardware for the flow.

The bug is on the SoC. In this cases the uCode might be able to act as a glue, but only as long as there are SoC “chicken bits”.

“WITHIN Hardware, a firmware(ucode in this case) can only do so much”

uCode delivery/loading

We provide a couple of modes to do uCode update – some of the are performed at reset, others are at BIOS, and yet another category can be applied in runtime (OS).

MSR 0x79, takes as an argument the address where the microcode is stored.

Operating systems vendors also provide this patch as part of the update process.

We provide a repository with the latest microcode versions, if you would like to update by yourself.

ret

0xCB

Plundervolt - mitigation

“After carefully reviewing the CPU voltage setting modification, Intel is mitigating the issue in two parts, a BIOS patch to disable the overclocking mailbox interface configuration. Secondly, a microcode update will be released that reflects the mailbox enablement status as part of SGX TCB [Trusted Computing Base] attestation. The Intel Attestation Service (IAS) and the Platform Certificate Retrieval Service will be updated with new keys in due course. The IAS users will receive a ‘CONFIGURATION NEEDED’ message from platforms that do not disable the overclocking mailbox interface.”

Plundervolt – disclosure/coordination

We received the report on June 2019.

The researchers kept updating us on progress, which led us to better understand the impact.

- Working with the community has proven to be a win-win scenario for Intel.

We disclosed on October 2019.

Hardware disclosure is hard.

- BIOS vendors.
- Operating System Vendors.
- SGX Customer.
- Multiple Research teams disclosed at the same time.
- General Community.

Acknowledgments

This talk, and all the work we do would be impossible without the support from all our Intel Colleagues.

The “Problem Response Team” – Ilya, Ittai x 2, Omer, Reuben, Neer, Chen, Yair, Gilad, Yoav x 2, Assaf

The external researchers who reported the bug:

Kit Murdock, David Oswald, Flavio Garcia, Navjivan Pal, Jo Van Bulck, Frank Piessens, Daniel Gruss, Zijo Kenjar, Tommaso Frassetto, Ahmed-Reza Sadeghi,David Gens, Michael Franz, Gang Qu, Yongqiang Lyu, Dongsheng Wang, Pengfei Qiu.

Conclusion.

Hardware is fun.

Hardware is hard.

We invite you to learn more about the topic. If you have any questions, feel free to contact us.

X: @aCaltum, @anders_fogh

If you need to disclose a bug to Intel, please don’t send it to us, and rather do it through our official channel – secure@intel.com