SHARING THE LOAD:
BUILDING A BETTER HW COMMUNITY

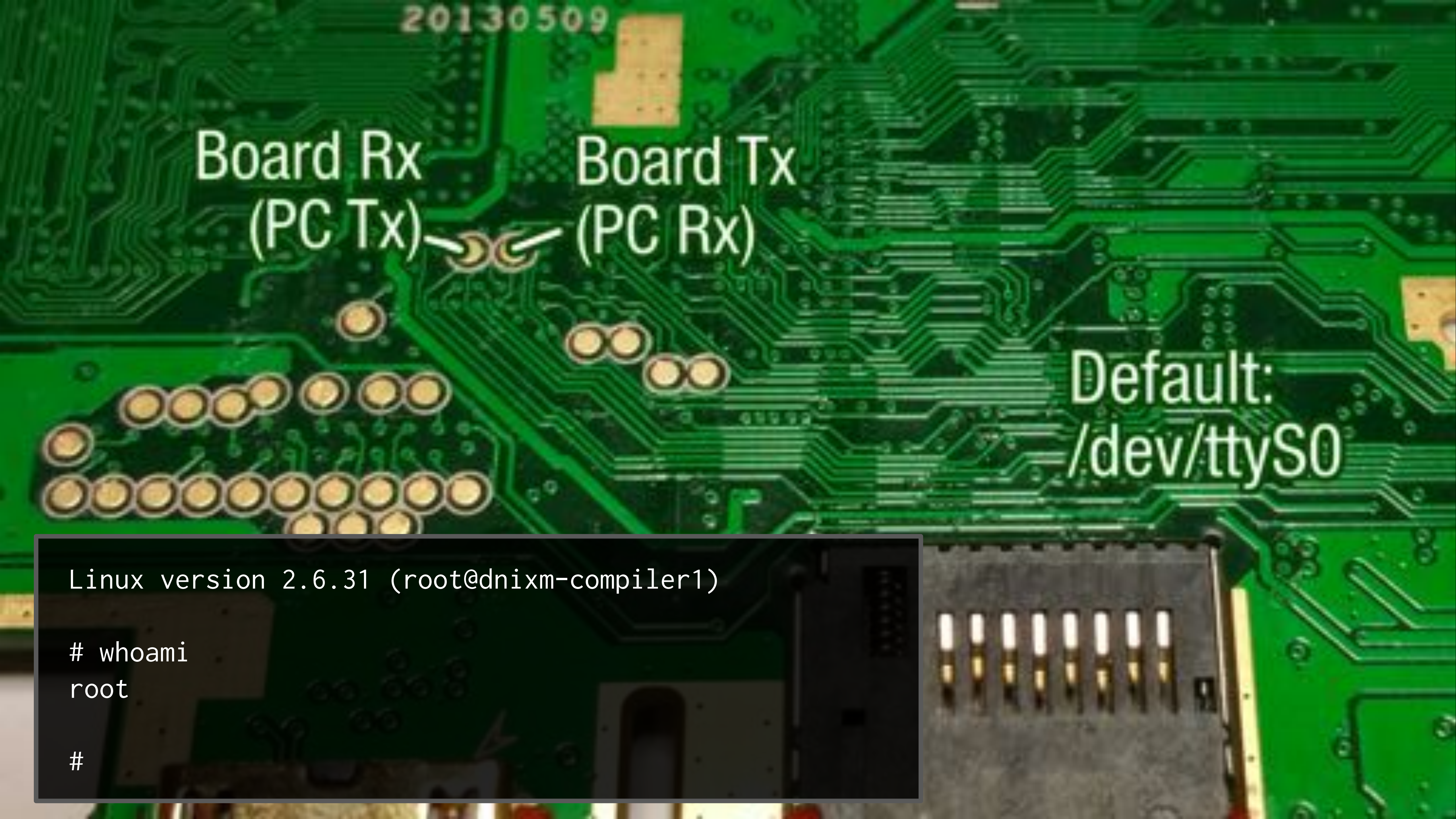
HARDWEAR.IO OPENING KEYNOTE

KATE TEMKIN /@ktemkin

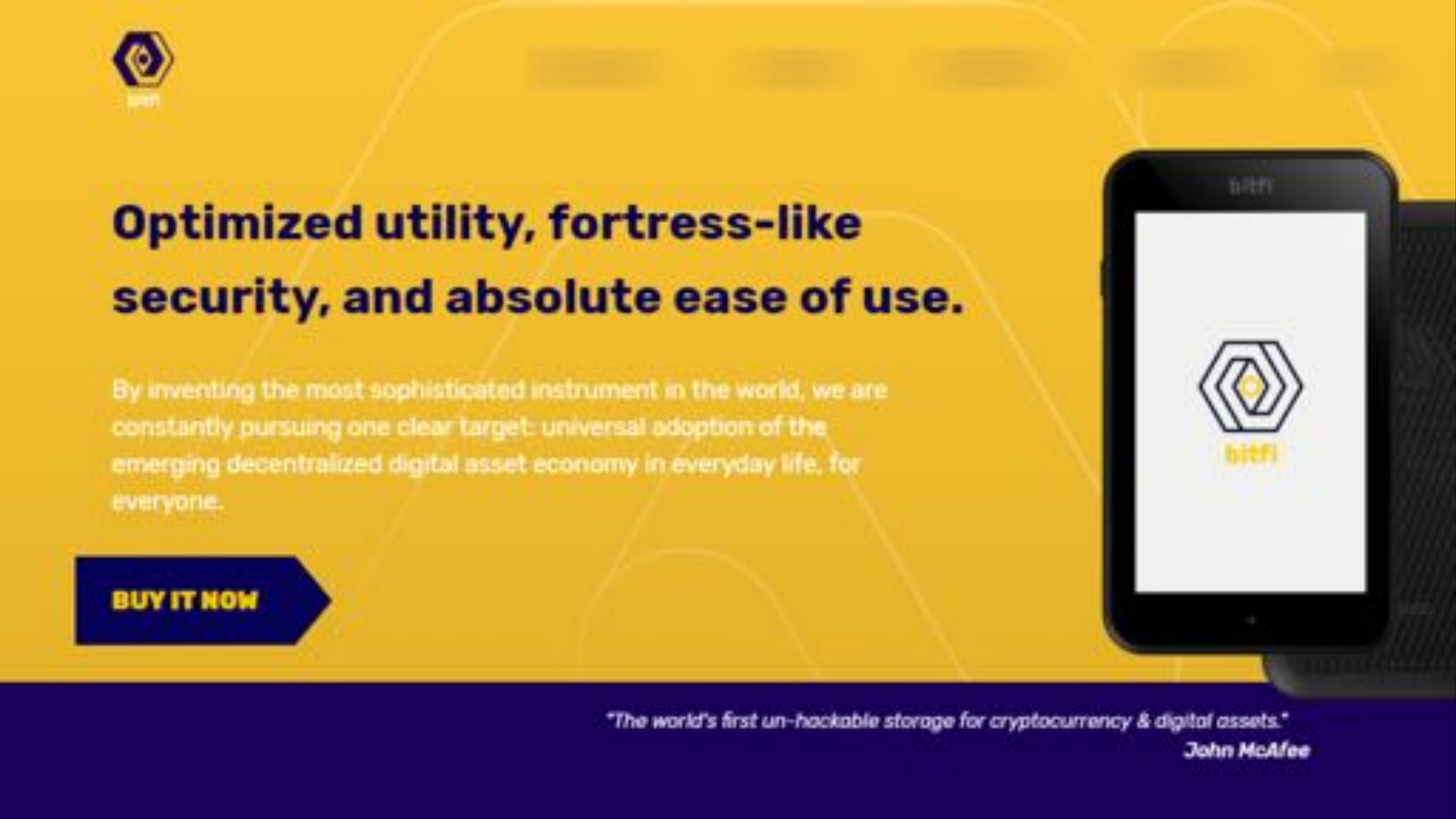
hardwear.io

THE STATE OF HARDWARE SECURITY…

KINDA SUCKS.



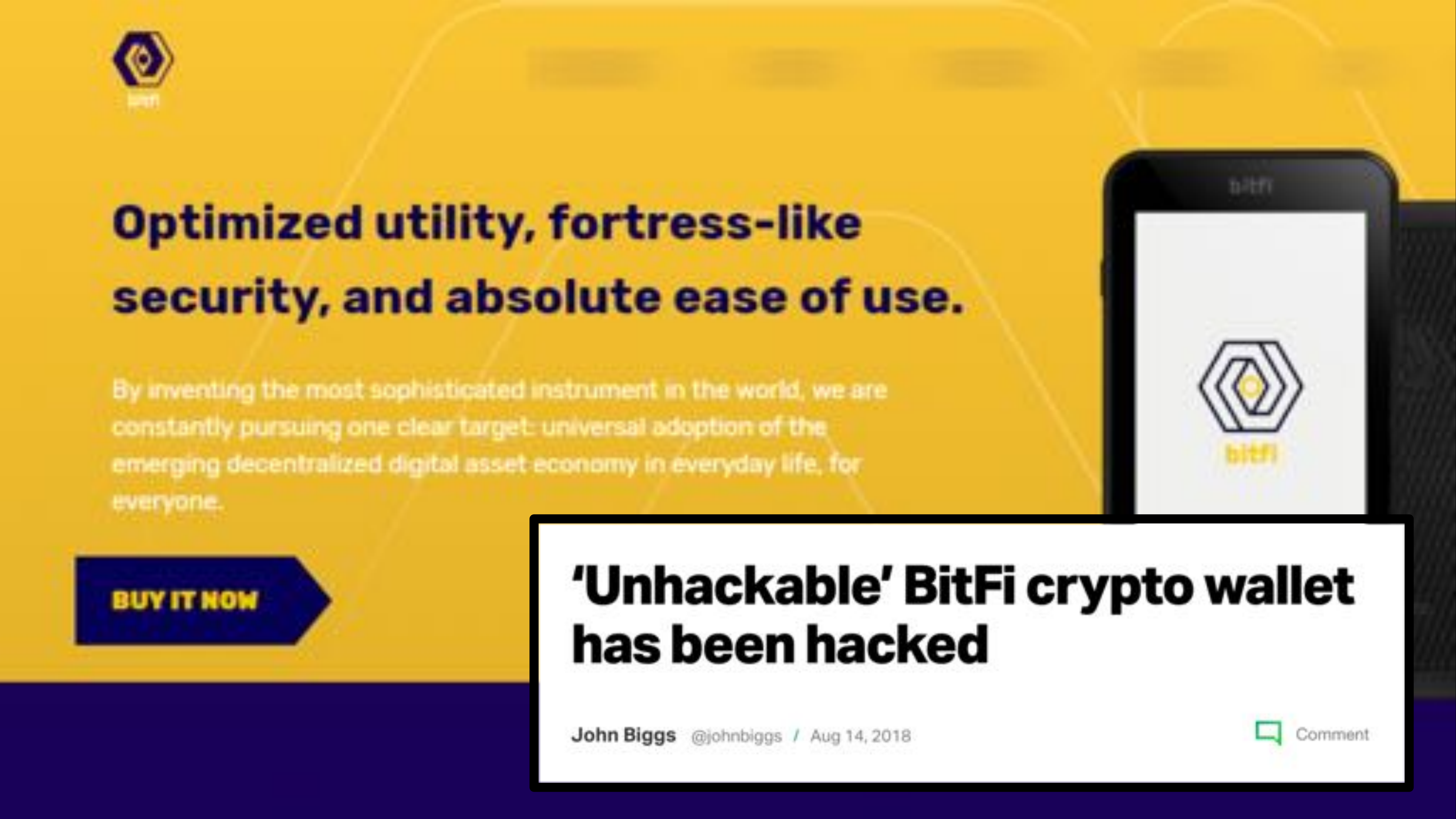
```
Linux version 2.6.31 (root@dnixm-compiler1)

# whoami
root

#
```



Optimized utility, fortress-like security, and absolute ease of use.

By inventing the most sophisticated instrument in the world, we are constantly pursuing one clear target: universal adoption of the emerging decentralized digital asset economy in everyday life, for everyone.

"The world's first un-hackable storage for cryptocurrency & digital assets."
John McAfee



Optimized utility, fortress-like security, and absolute ease of use.

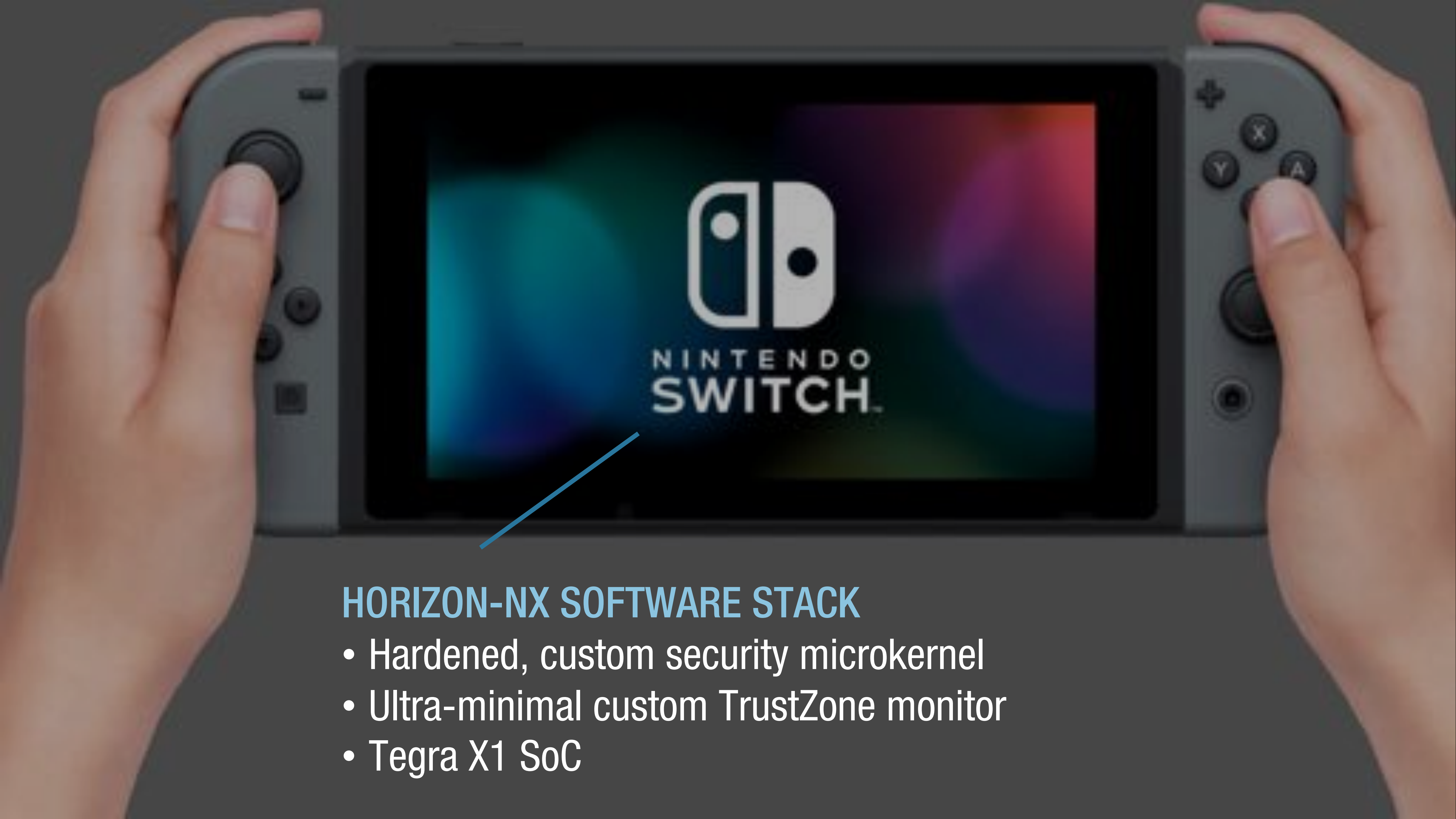
By inventing the most sophisticated instrument in the world, we are constantly pursuing one clear target: universal adoption of the emerging decentralized digital asset economy in everyday life, for everyone.

BUY IT NOW

‘Unhackable’ BitFi crypto wallet has been hacked

John Biggs @johnbiggs / Aug 14, 2018

Comment

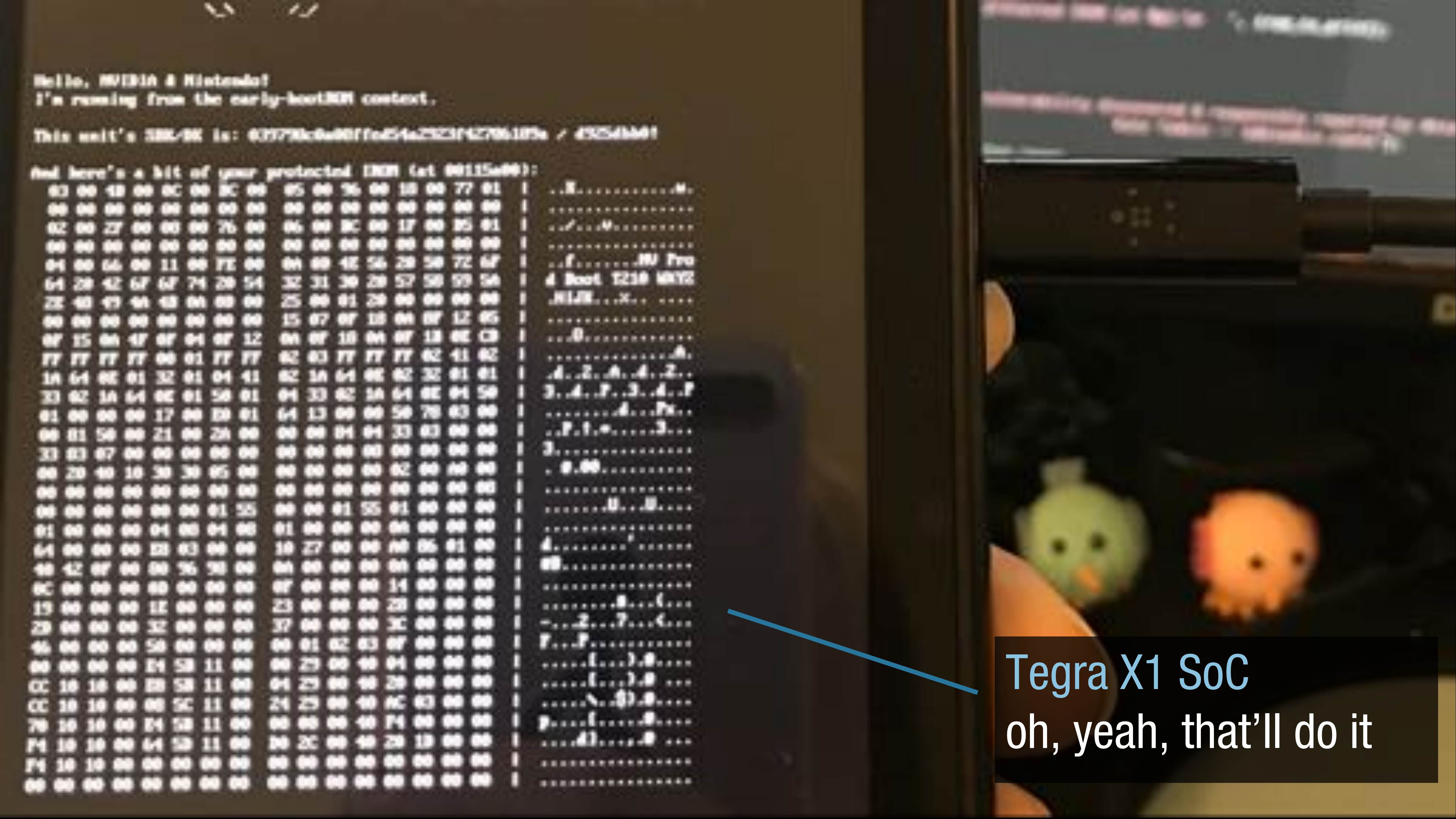


HORIZON-NX SOFTWARE STACK

- Hardened, custom security microkernel
- Ultra-minimal custom TrustZone monitor
- Tegra X1 SoC

HORIZON-NX SOFTWARE STACK

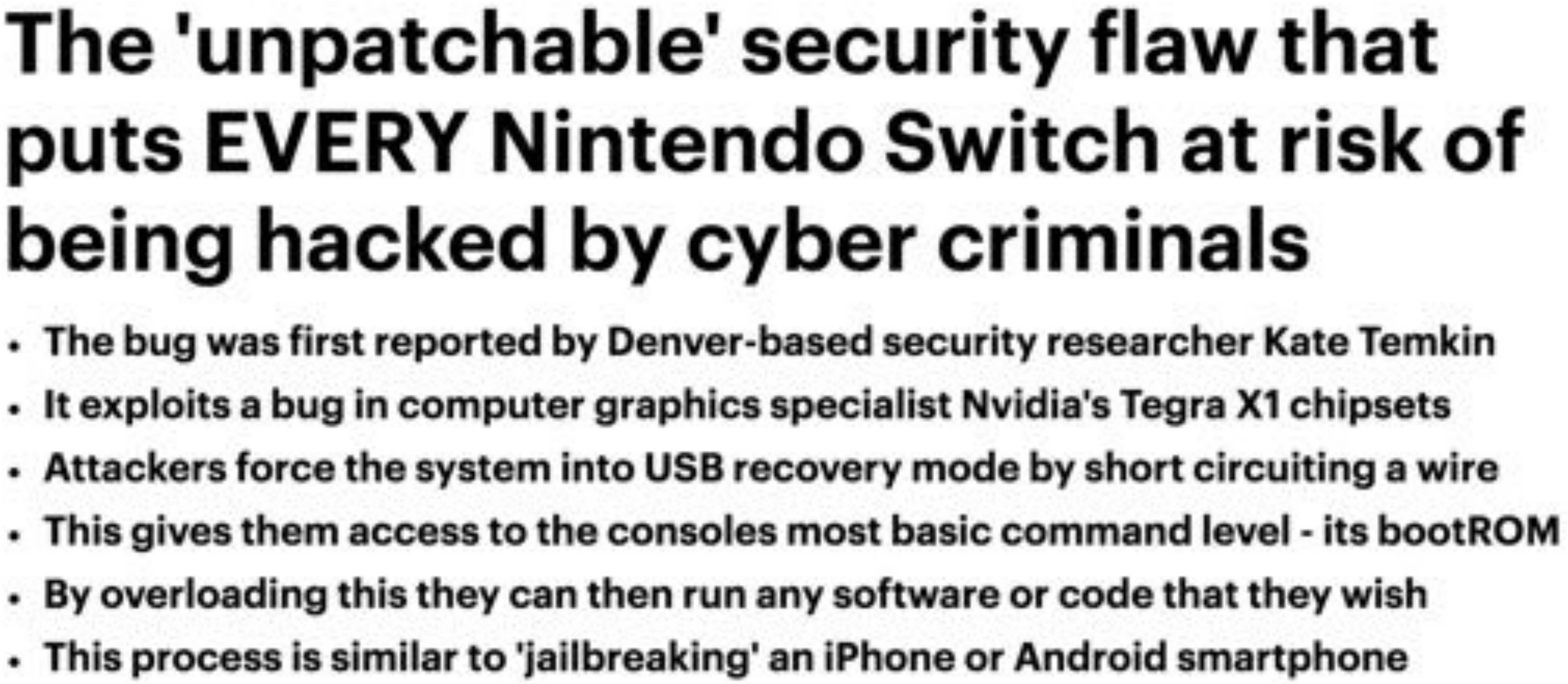
- Hardened, custom security microkernel
- Ultra-minimal custom TrustZone monitor
- Tegra X1 SoC



```
Hello, NVIDIA & Nintendo!
I'm running from the early-bootROM context.

This unit's SBK/DK is: 03f9790c0a00ffed54a25923f427061f9a / 45254bb401

And here's a bit of your protected IROM (at 00115e00):
  03 00 48 00 0C 00 BC 00  05 00 96 00 10 00 77 01  |  ..H...........w.
  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |  ................
  02 00 2F 00 00 00 76 00  06 00 BC 00 17 00 B5 01  |  ../...v.........
  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |  ................
  04 00 66 00 11 00 FE 00  0A 00 4E 56 20 50 72 6F  |  ..f.......NV Pro
  64 20 42 6F 6F 74 20 54  32 31 30 20 57 58 59 5A  |  d Boot T210 WXYZ
  2E 48 49 4A 4B 0A 00 00  25 00 01 20 00 00 00 00  |  .HIJK...%.. ....
  00 00 00 00 00 00 00 00  15 07 0F 10 0A 0F 12 05  |  ................
  0F 15 0A 47 0F 04 0F 12  0A 0F 10 0A 0F 13 0E C0  |  ...G............
  FF FF FF FF 00 01 FF FF  02 03 FF FF FF 02 41 02  |  ..............A.
  1A 64 0E 01 32 01 04 41  02 1A 64 0E 02 32 01 01  |  .d..2..A..d..2..
  33 02 1A 64 0E 01 50 01  04 33 02 1A 64 0E 04 50  |  3..d..P..3..d..P
  01 00 00 00 17 00 E0 01  64 13 00 00 50 78 03 00  |  ........d...Px..
  00 01 50 00 21 00 2A 00  00 00 04 04 33 03 00 00  |  ..P.!.*.....3...
  33 03 07 00 00 00 00 00  00 00 00 00 00 00 00 00  |  3...............
  00 20 40 10 30 30 05 00  00 00 00 00 02 00 A0 00  |  . @.00..........
  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |  ................
  00 00 00 00 00 00 01 55  00 00 01 55 01 00 00 00  |  .......U...U....
  01 00 00 00 04 00 04 00  01 00 00 00 0A 00 00 00  |  ................
  64 00 00 00 E8 03 00 00  10 27 00 00 A0 86 01 00  |  d........'......
  40 42 0F 00 00 96 98 00  0A 00 00 00 0A 00 00 00  |  @B..............
  0C 00 00 00 00 00 00 00  0F 00 00 00 14 00 00 00  |  ................
  19 00 00 00 1E 00 00 00  23 00 00 00 28 00 00 00  |  ........#...(...
  2D 00 00 00 32 00 00 00  37 00 00 00 3C 00 00 00  |  -...2...7...<...
  46 00 00 00 50 00 00 00  00 01 02 03 0F 00 00 00  |  F...P...........
  00 00 00 00 E4 5B 11 00  00 29 00 40 04 00 00 00  |  .....[...).@....
  CC 10 10 00 E8 5B 11 00  04 29 00 40 20 00 00 00  |  .....[...).@ ...
  CC 10 10 00 00 5C 11 00  24 29 00 40 AC 03 00 00  |  .....\..$).@....
  70 10 10 00 E4 5B 11 00  00 00 00 40 F4 00 00 00  |  p....[.....@....
  F4 10 10 00 64 5B 11 00  D0 2C 00 40 20 10 00 00  |  ....d[...,.@ ...
  F4 10 10 00 00 00 00 00  00 00 00 00 00 00 00 00  |  ................
  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |  ................
```

Tegra X1 SoC

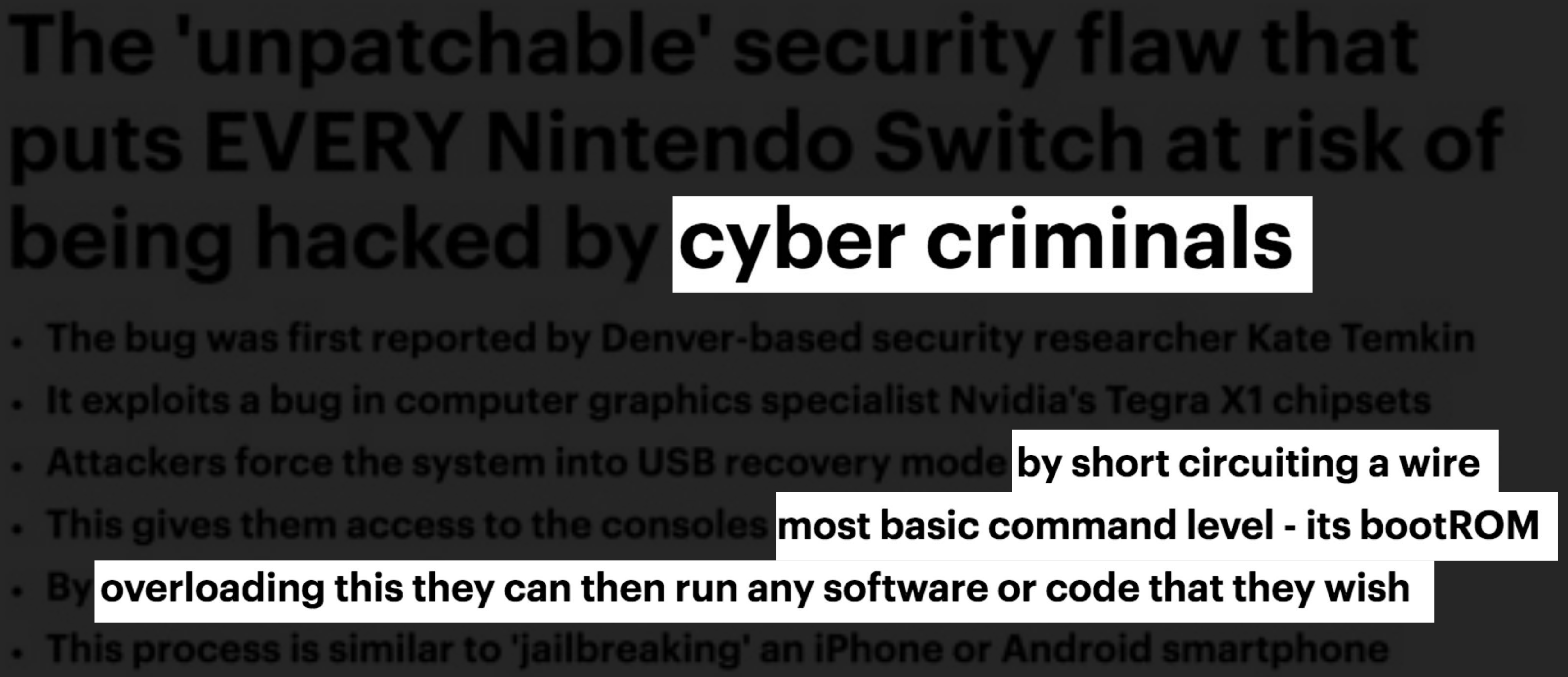
oh, yeah, that'll do it



The 'unpatchable' security flaw that puts EVERY Nintendo Switch at risk of being hacked by cyber criminals

- The bug was first reported by Denver-based security researcher Kate Temkin
- It exploits a bug in computer graphics specialist Nvidia's Tegra X1 chipsets
- Attackers force the system into USB recovery mode by short circuiting a wire
- This gives them access to the consoles most basic command level - its bootROM
- By overloading this they can then run any software or code that they wish
- This process is similar to 'jailbreaking' an iPhone or Android smartphone

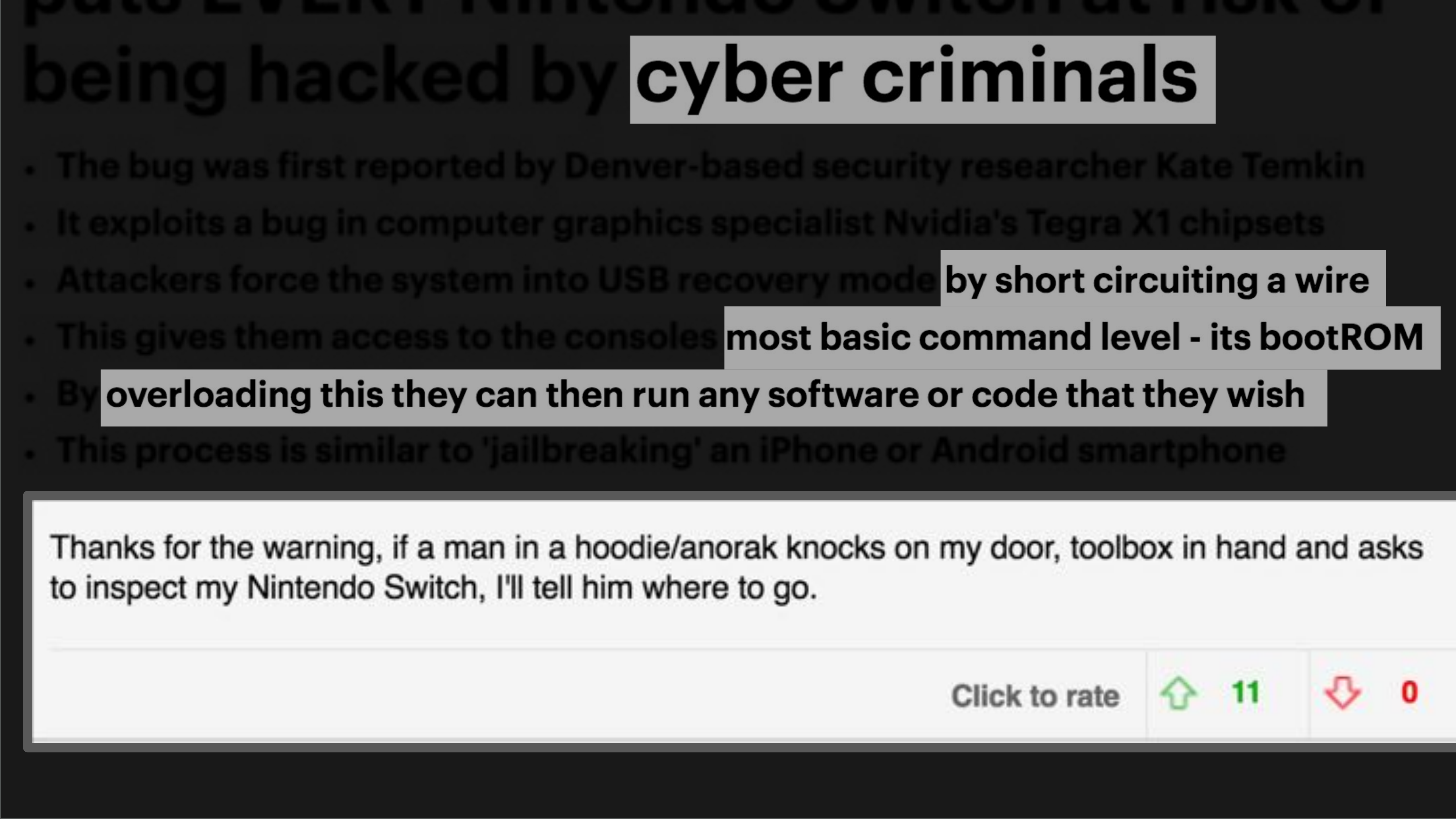
MailOnline



The 'unpatchable' security flaw that puts EVERY Nintendo Switch at risk of being hacked by cyber criminals

- The bug was first reported by Denver-based security researcher Kate Temkin
- It exploits a bug in computer graphics specialist Nvidia's Tegra X1 chipsets
- Attackers force the system into USB recovery mode by short circuiting a wire
- This gives them access to the consoles most basic command level - its bootROM
- By overloading this they can then run any software or code that they wish
- This process is similar to 'jailbreaking' an iPhone or Android smartphone

MailOnline



puts EVERY Nintendo Switch at risk of being hacked by cyber criminals

- The bug was first reported by Denver-based security researcher Kate Temkin
- It exploits a bug in computer graphics specialist Nvidia's Tegra X1 chipsets
- Attackers force the system into USB recovery mode by short circuiting a wire
- This gives them access to the consoles most basic command level - its bootROM
- By overloading this they can then run any software or code that they wish
- This process is similar to 'jailbreaking' an iPhone or Android smartphone

Thanks for the warning, if a man in a hoodie/anorak knocks on my door, toolbox in hand and asks to inspect my Nintendo Switch, I'll tell him where to go.

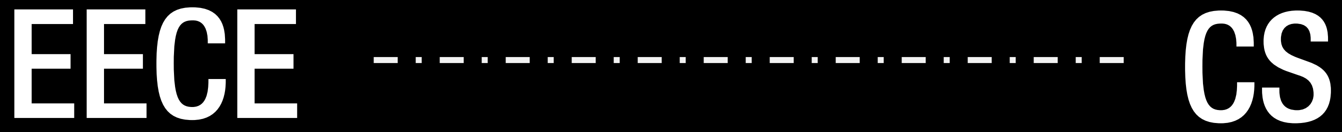
SO,

HOW DID WE WIND UP HERE?

SO, WHO *REALLY* ARE YOU?

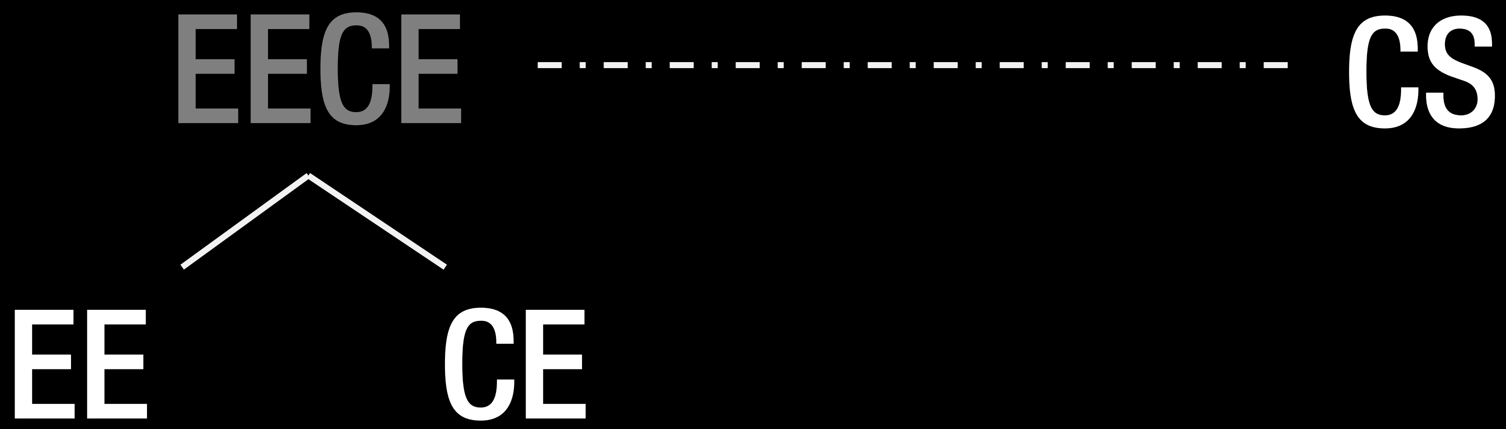
Katherine/Kate Temkin (@ktemkin):

- founder, Insomnia Security
- Nintendo Switch inspector
- glitch witch & open-source-tool-builder
- educational (reverse) engineer
- occasional engineering streamer



EECE

CS

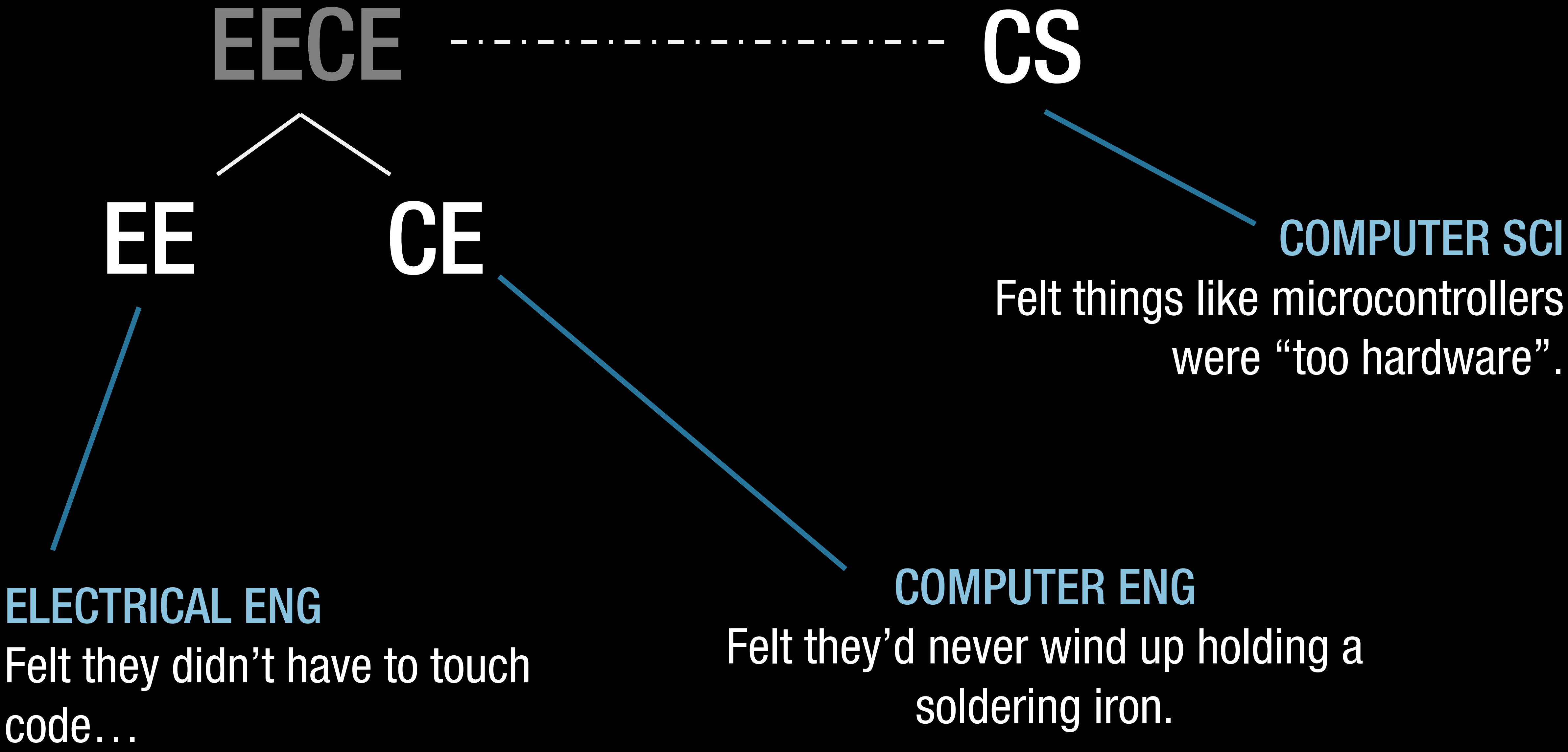


EECE

CS

EE

CE



EECE

CS

EE

CE

COMPUTER SCI

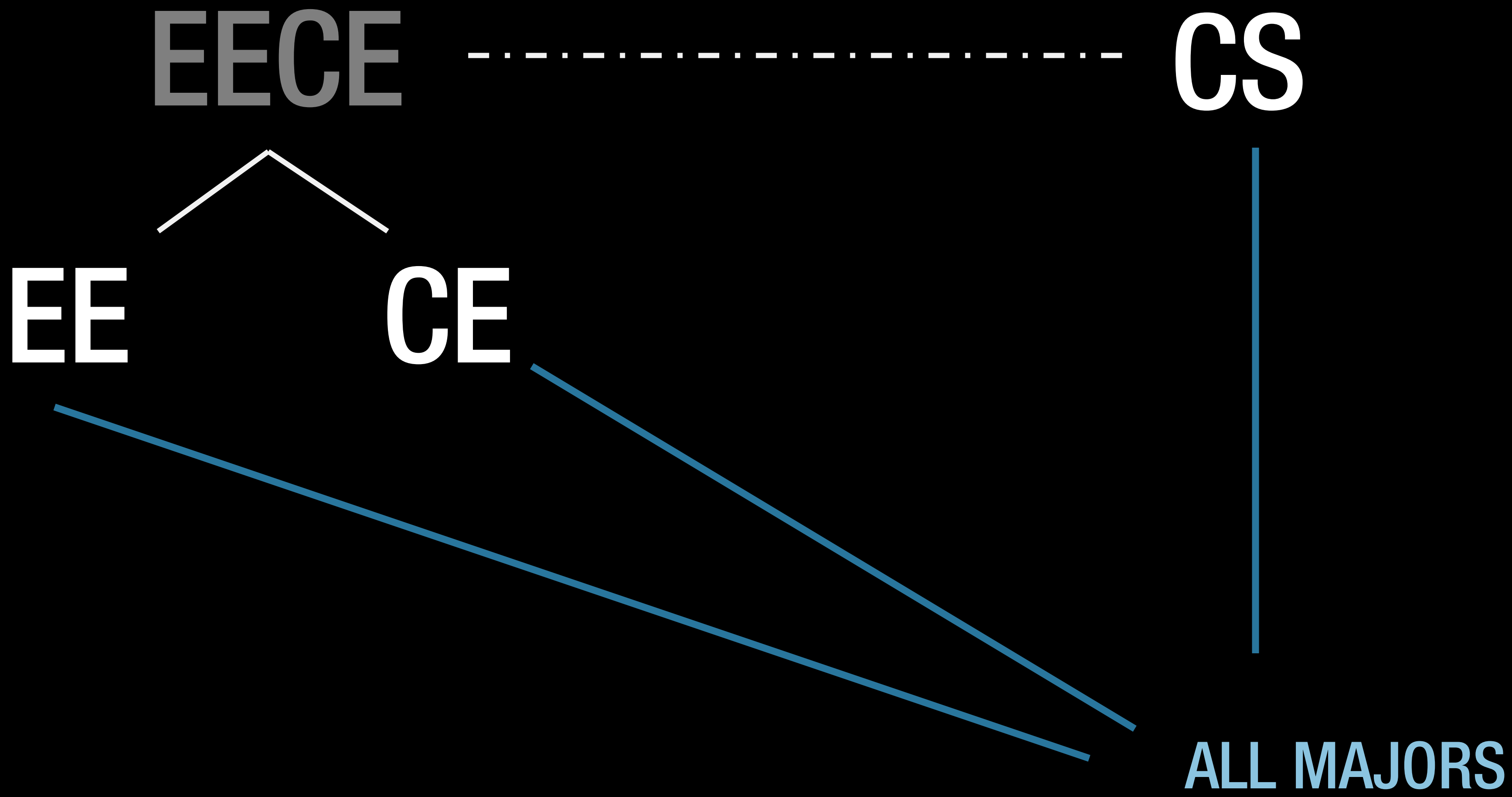
Felt things like microcontrollers were “too hardware”.

ELECTRICAL ENG

Felt they didn’t have to touch code…

COMPUTER ENG

Felt they’d never wind up holding a soldering iron.



EECE

CS

EE

CE

ALL MAJORS

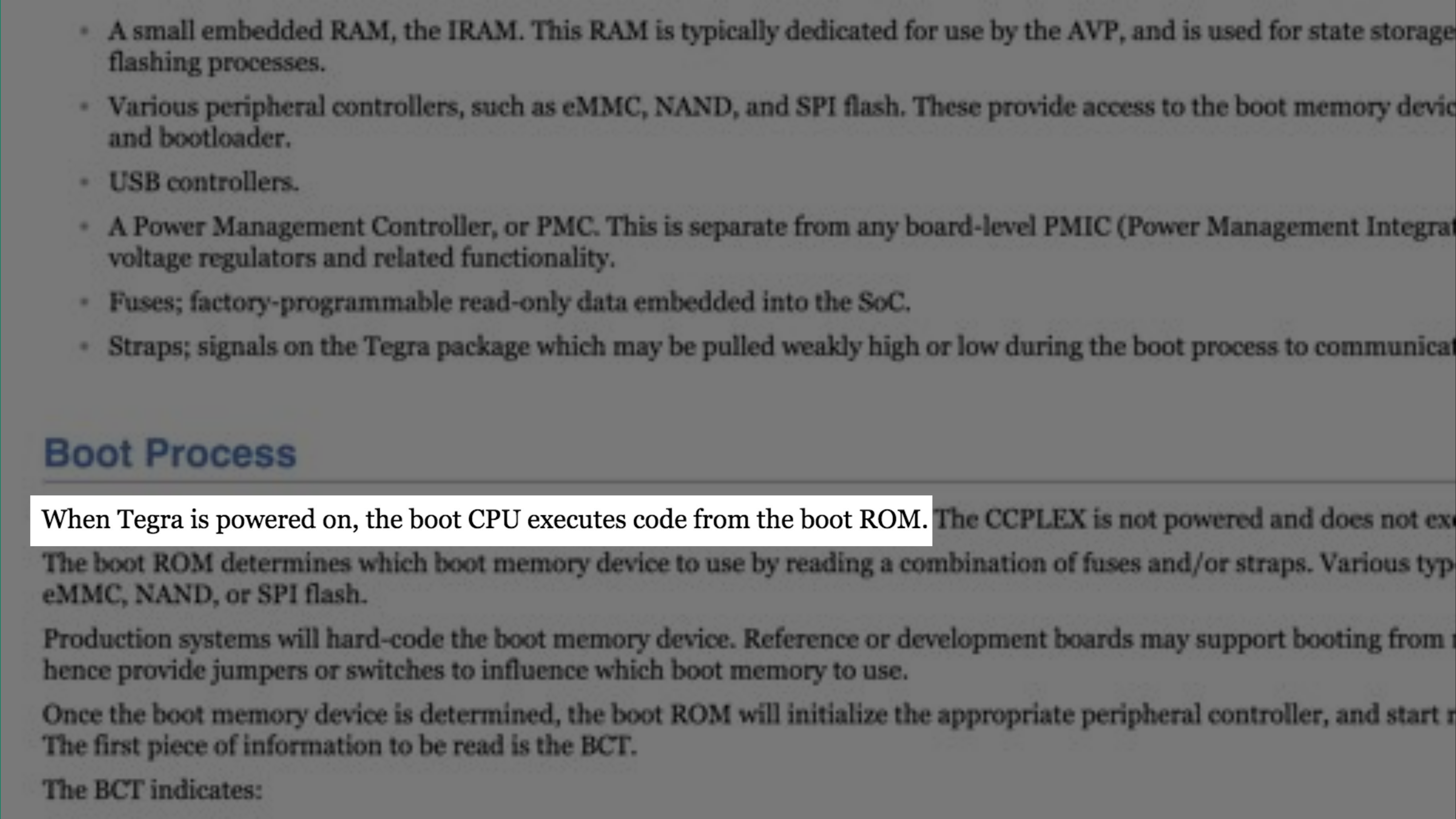
Tended to think they were there to *apply* techniques created by ‘heroic inventors’.

FAST FORWARD N YEARS

NINTENDO
SWITCH
X
Y
A

HORIZON-NX SOFTWARE STACK

- Extremely difficult to attack on the software front
- But that software is built – and runs – atop **hardware.**



- A small embedded RAM, the IRAM. This RAM is typically dedicated for use by the AVP, and is used for state storage
  flashing processes.
- Various peripheral controllers, such as eMMC, NAND, and SPI flash. These provide access to the boot memory devic
  and bootloader.
- USB controllers.
- A Power Management Controller, or PMC. This is separate from any board-level PMIC (Power Management Integrat
  voltage regulators and related functionality.
- Fuses; factory-programmable read-only data embedded into the SoC.
- Straps; signals on the Tegra package which may be pulled weakly high or low during the boot process to communicat

Boot Process

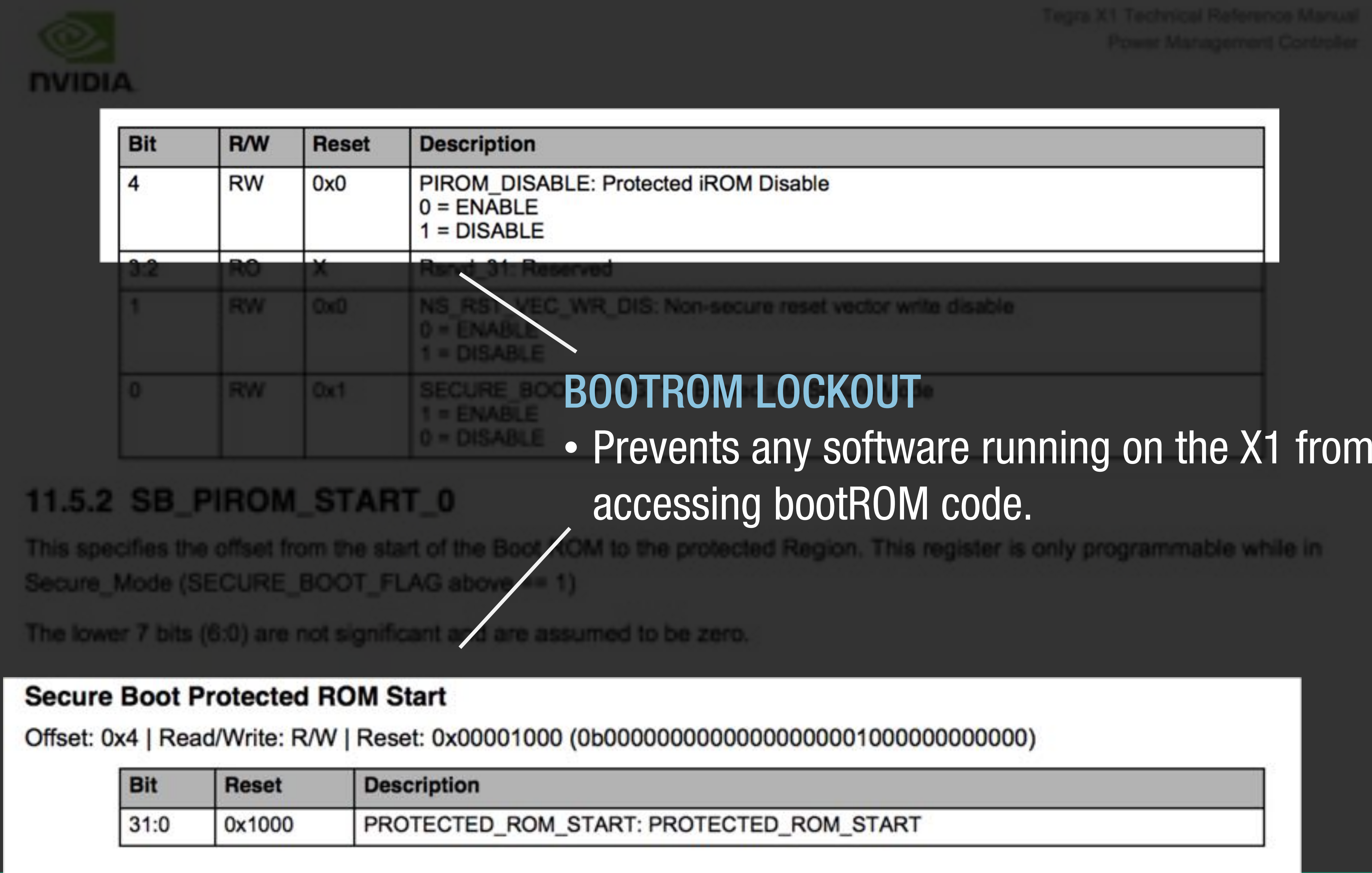
When Tegra is powered on, the boot CPU executes code from the boot ROM. The CCPLEX is not powered and does not ex

The boot ROM determines which boot memory device to use by reading a combination of fuses and/or straps. Various typ
eMMC, NAND, or SPI flash.

Production systems will hard-code the boot memory device. Reference or development boards may support booting from
hence provide jumpers or switches to influence which boot memory to use.

Once the boot memory device is determined, the boot ROM will initialize the appropriate peripheral controller, and start
The first piece of information to be read is the BCT.

The BCT indicates:



Bit	R/W	Reset	Description
4	RW	0x0	PIROM_DISABLE: Protected iROM Disable 0 = ENABLE 1 = DISABLE
3:2	RO	X	Rsvd_31: Reserved
1	RW	0x0	NS_RST_VEC_WR_DIS: Non-secure reset vector write disable 0 = ENABLE 1 = DISABLE
0	RW	0x1	SECURE_BOO[illegible]de 1 = ENABLE 0 = DISABLE

11.5.2 SB_PIROM_START_0

This specifies the offset from the start of the Boot ROM to the protected Region. This register is only programmable while in Secure_Mode (SECURE_BOOT_FLAG above == 1)

The lower 7 bits (6:0) are not significant and are assumed to be zero.

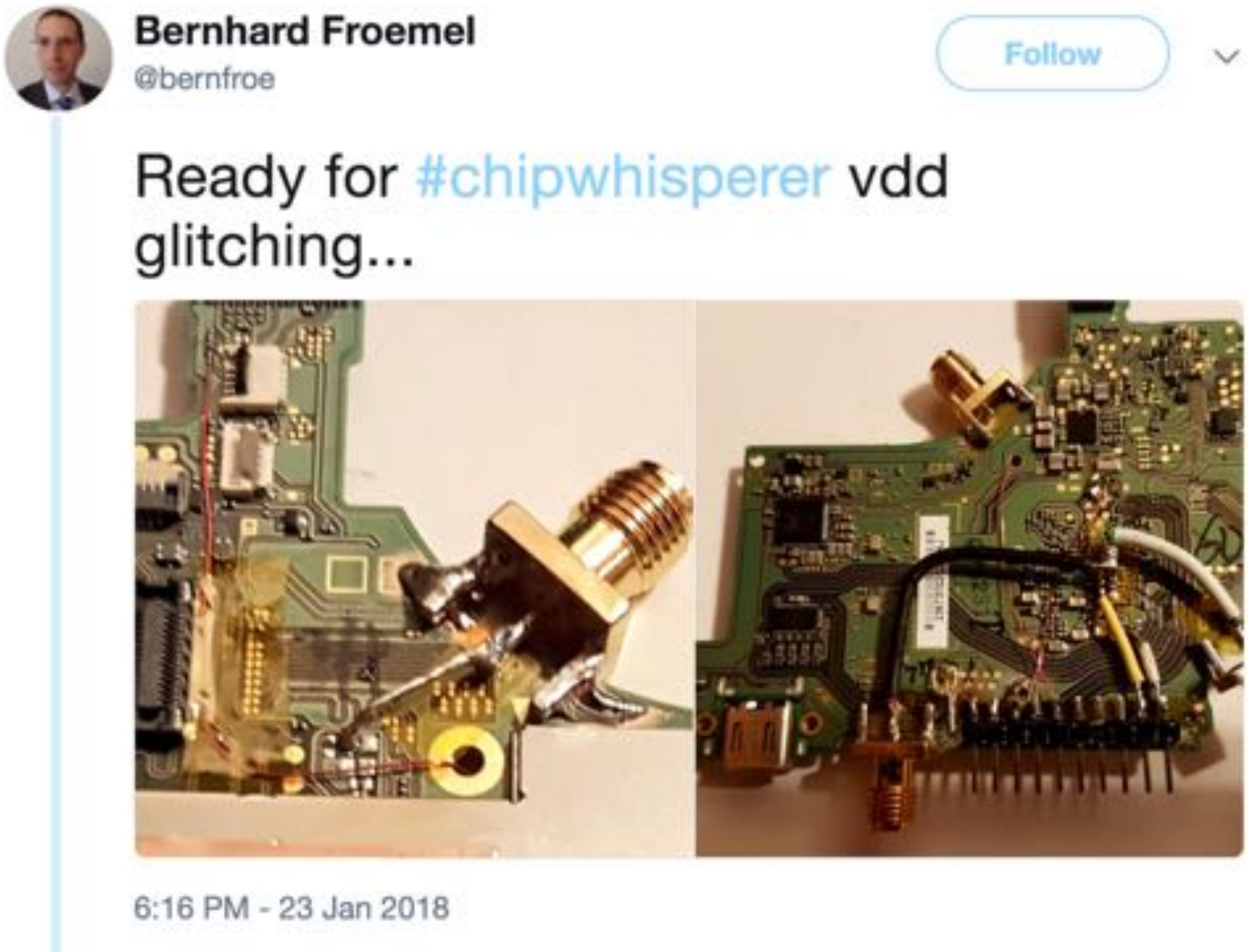
Secure Boot Protected ROM Start

Offset: 0x4 | Read/Write: R/W | Reset: 0x00001000 (0b00000000000000000001000000000000)

Bit	Reset	Description
31:0	0x1000	PROTECTED_ROM_START: PROTECTED_ROM_START

BOOTROM LOCKOUT

- Prevents any software running on the X1 from accessing bootROM code.

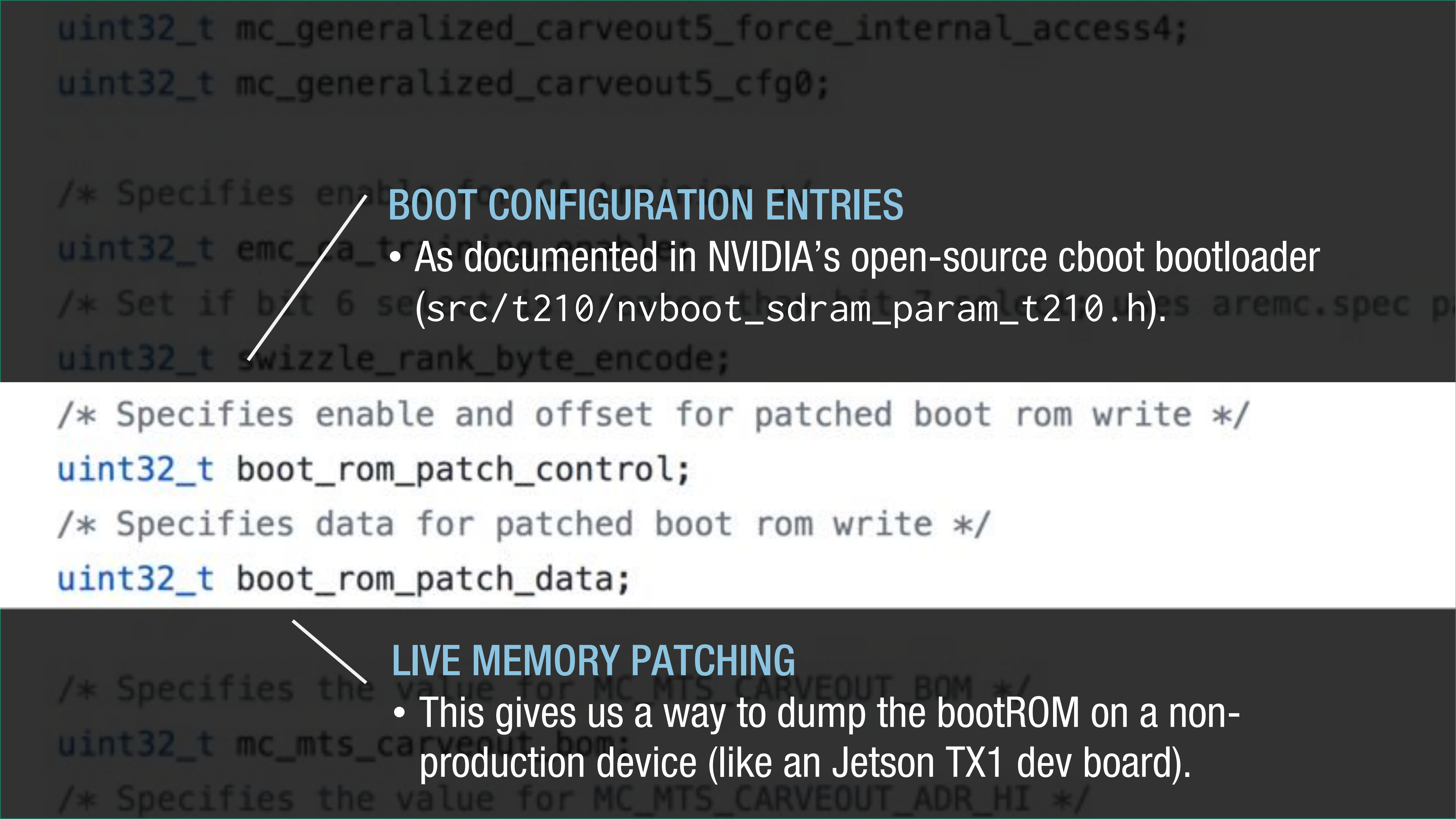


Bernhard Froemel

@bernfroe

Ready for #chipwhisperer vdd glitching...

6:16 PM - 23 Jan 2018



```
uint32_t mc_generalized_carveout5_force_internal_access4;
uint32_t mc_generalized_carveout5_cfg0;

/* Specifies enable for CA training */
uint32_t emc_ca_training_enable;
/* Set if bit 6 select is greater than bit 7 select; uses aremc.spec p
uint32_t swizzle_rank_byte_encode;
/* Specifies enable and offset for patched boot rom write */
uint32_t boot_rom_patch_control;
/* Specifies data for patched boot rom write */
uint32_t boot_rom_patch_data;

/* Specifies the value for MC_MTS_CARVEOUT_BOM */
uint32_t mc_mts_carveout_bom;
/* Specifies the value for MC_MTS_CARVEOUT_ADR_HI */
```

BOOT CONFIGURATION ENTRIES

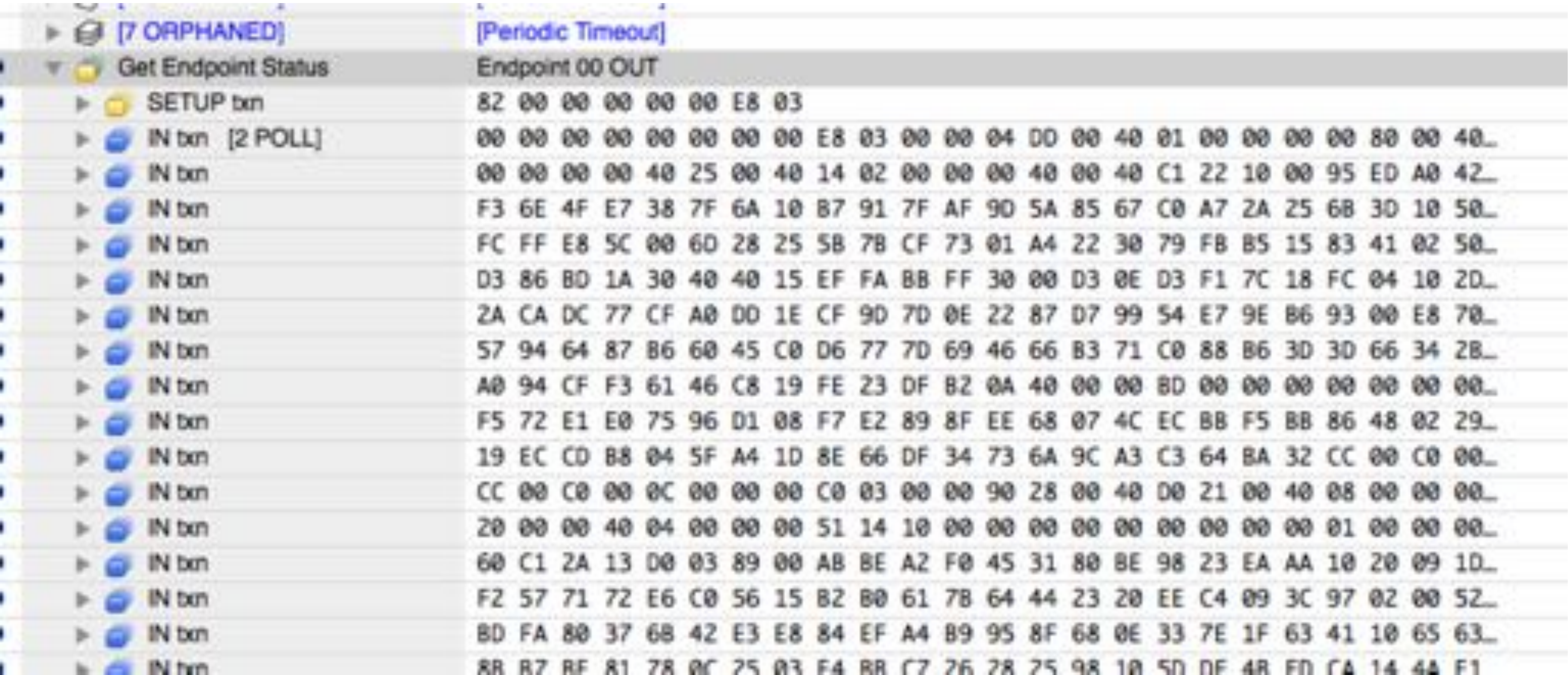
- As documented in NVIDIA’s open-source cboot bootloader (`src/t210/nvboot_sdram_param_t210.h`).

LIVE MEMORY PATCHING

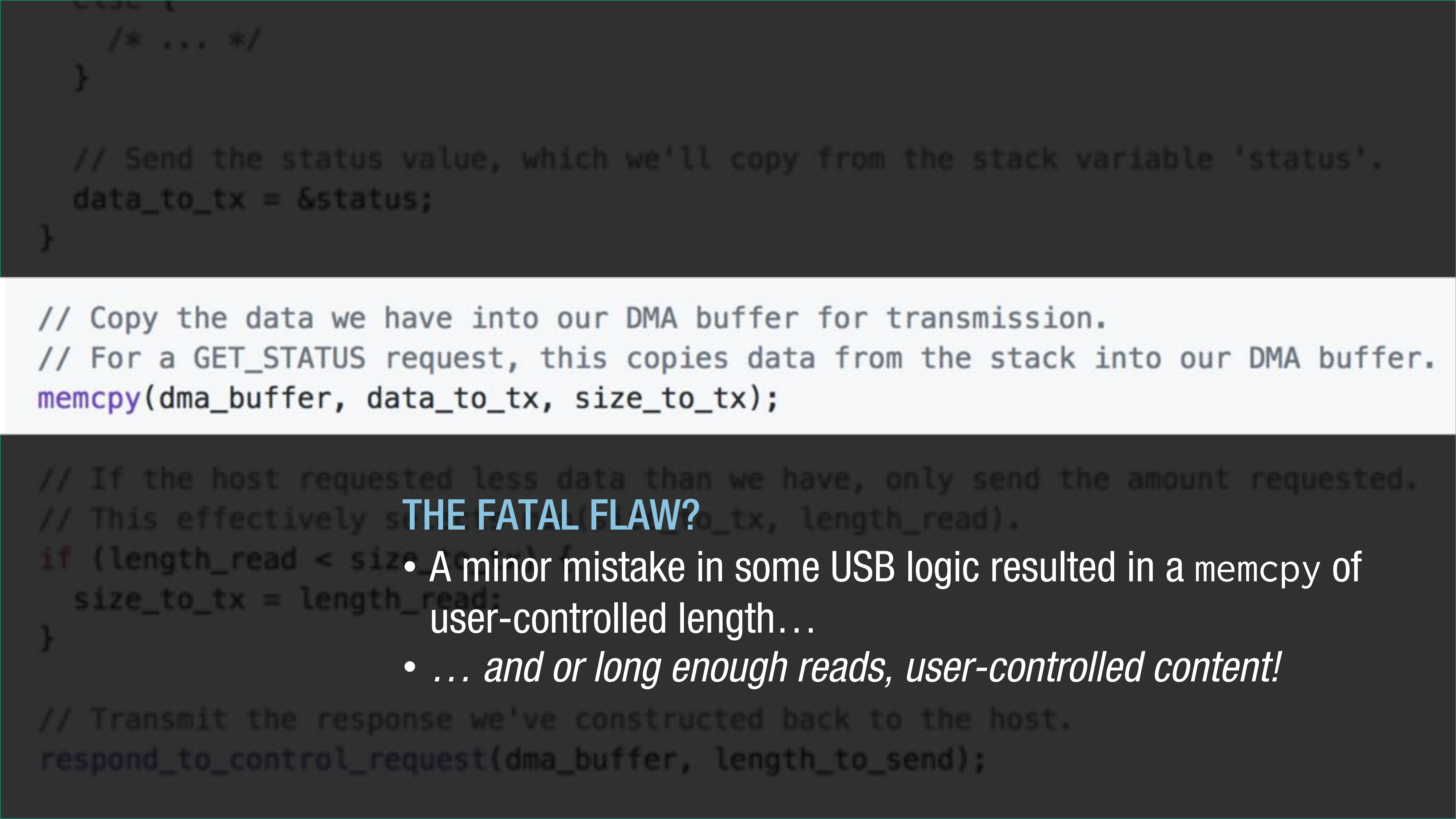
- This gives us a way to dump the bootROM on a non-production device (like an Jetson TX1 dev board).

```
ctrl_transfer(STANDARD_REQUEST_DEVICE_TO_HOST_TO_ENDPOINT,
        GET_STATUS, 0, 0, 4096)
```

USB tools at:
https://github.com/ktemkin/Facedancer



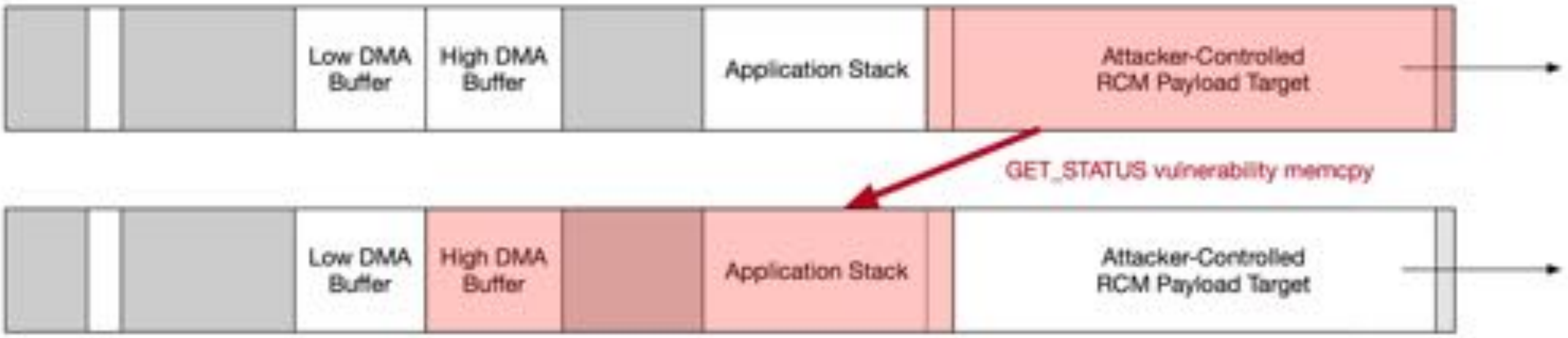
```
[7 ORPHANED]           [Periodic Timeout]
Get Endpoint Status    Endpoint 00 OUT
  SETUP txn            82 00 00 00 00 00 E8 03
  IN txn  [2 POLL]     00 00 00 00 00 00 00 00 E8 03 00 00 04 DD 00 40 01 00 00 00 00 80 00 40…
  IN txn               00 00 00 00 40 25 00 40 14 02 00 00 00 40 00 40 C1 22 10 00 95 ED A0 42…
  IN txn               F3 6E 4F E7 38 7F 6A 10 B7 91 7F AF 9D 5A 85 67 C0 A7 2A 25 6B 3D 10 50…
  IN txn               FC FF E8 5C 00 6D 28 25 5B 7B CF 73 01 A4 22 30 79 FB B5 15 83 41 02 50…
  IN txn               D3 86 BD 1A 30 40 40 15 EF FA BB FF 30 00 D3 0E D3 F1 7C 18 FC 04 10 2D…
  IN txn               2A CA DC 77 CF A0 DD 1E CF 9D 7D 0E 22 87 D7 99 54 E7 9E B6 93 00 E8 70…
  IN txn               57 94 64 87 B6 60 45 C0 D6 77 7D 69 46 66 B3 71 C0 88 B6 3D 3D 66 34 2B…
  IN txn               A0 94 CF F3 61 46 C8 19 FE 23 DF B2 0A 40 00 00 BD 00 00 00 00 00 00 00…
  IN txn               F5 72 E1 E0 75 96 D1 08 F7 E2 89 8F EE 68 07 4C EC BB F5 BB 86 48 02 29…
  IN txn               19 EC CD B8 04 5F A4 1D 8E 66 DF 34 73 6A 9C A3 C3 64 BA 32 CC 00 C0 00…
  IN txn               CC 00 C0 00 0C 00 00 00 C0 03 00 00 90 28 00 40 D0 21 00 40 08 00 00 00…
  IN txn               20 00 00 40 04 00 00 00 51 14 10 00 00 00 00 00 00 00 00 00 01 00 00 00…
  IN txn               60 C1 2A 13 D0 03 89 00 AB BE A2 F0 45 31 80 BE 98 23 EA AA 10 20 09 1D…
  IN txn               F2 57 71 72 E6 C0 56 15 B2 B0 61 7B 64 44 23 20 EE C4 09 3C 97 02 00 52…
  IN txn               BD FA 80 37 6B 42 E3 E8 84 EF A4 B9 95 8F 68 0E 33 7E 1F 63 41 10 65 63…
  IN txn               8B B7 BF 81 78 0C 25 03 F4 BB C7 26 28 25 98 10 5D DF 4B FD CA 14 4A F1
```



```
  else {
    /* ... */
  }

  // Send the status value, which we'll copy from the stack variable 'status'.
  data_to_tx = &status;
}

// Copy the data we have into our DMA buffer for transmission.
// For a GET_STATUS request, this copies data from the stack into our DMA buffer.
memcpy(dma_buffer, data_to_tx, size_to_tx);

// If the host requested less data than we have, only send the amount requested.
// This effectively sets size_to_tx = min(size_to_tx, length_read).
if (length_read < size_to_tx) {
  size_to_tx = length_read;
}

// Transmit the response we've constructed back to the host.
respond_to_control_request(dma_buffer, length_to_send);
```

THE FATAL FLAW?

- A minor mistake in some USB logic resulted in a `memcpy` of user-controlled length…
- *… and or long enough reads, user-controlled content!*


Low DMA Buffer
High DMA Buffer
Application Stack
Attacker-Controlled RCM Payload Target
GET_STATUS vulnerability memcpy
Low DMA Buffer
High DMA Buffer
Application Stack
Attacker-Controlled RCM Payload Target

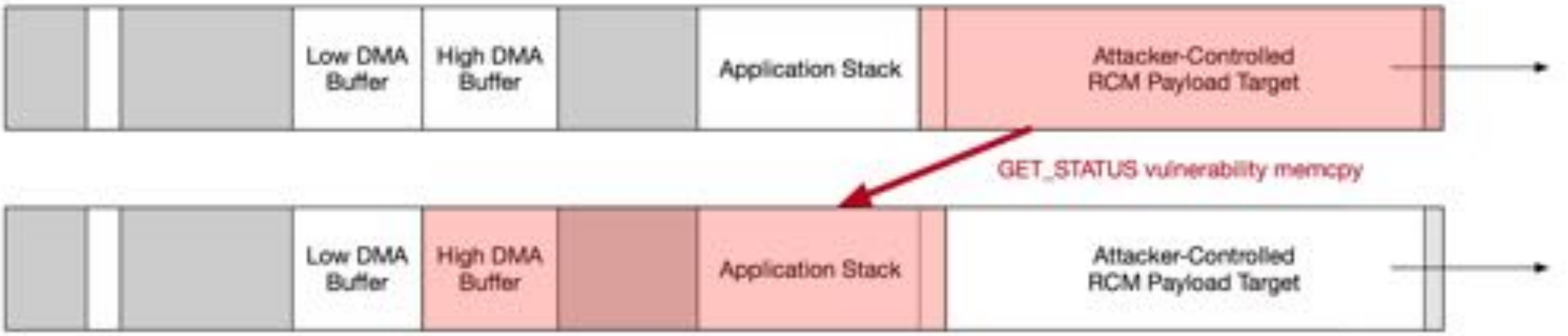
CVE-2018-6242 (“Fusée Gelée” / ”shofEL2”)

- Easy to **apply** locally, easy to **discover**, and **simple in mechanism.**
- Completely compromises all **root-of-trust technology** on relevant processors.
  - *Wait, which processors?*

TEGRA PROCESSOR SERIES

- Tegra APX: affected
- Tegra 2: affected
- Tegra 3: affected
- Tegra 4: affected
- Tegra K1: affected
- Tegra X1: affected
- Tegra X2: not affected

phew



Low DMA Buffer
High DMA Buffer
Application Stack
Attacker-Controlled RCM Payload Target
GET_STATUS vulnerability memcpy
Low DMA Buffer
High DMA Buffer
Application Stack
Attacker-Controlled RCM Payload Target

CVE-2018-6242 (“Fusée Gelée” / ”shofEL2”)

- Easy to **apply** locally, easy to **discover**, and **simple in mechanism.**
- Completely compromises all **root-of-trust technology** on **most Tegras.**

So: how the heck did this stick around for so long?

OKAY:
SO WHAT DO WE DO NOW?

WELL, THEY DON'T CALL IT EASY-WARE.

— @securelyfitz

Show hardware hacking as **approachable**, rather than as **deep wizardry**.

WELL, THEY DON’T CALL IT EASY-WARE.

— @securelyfitz

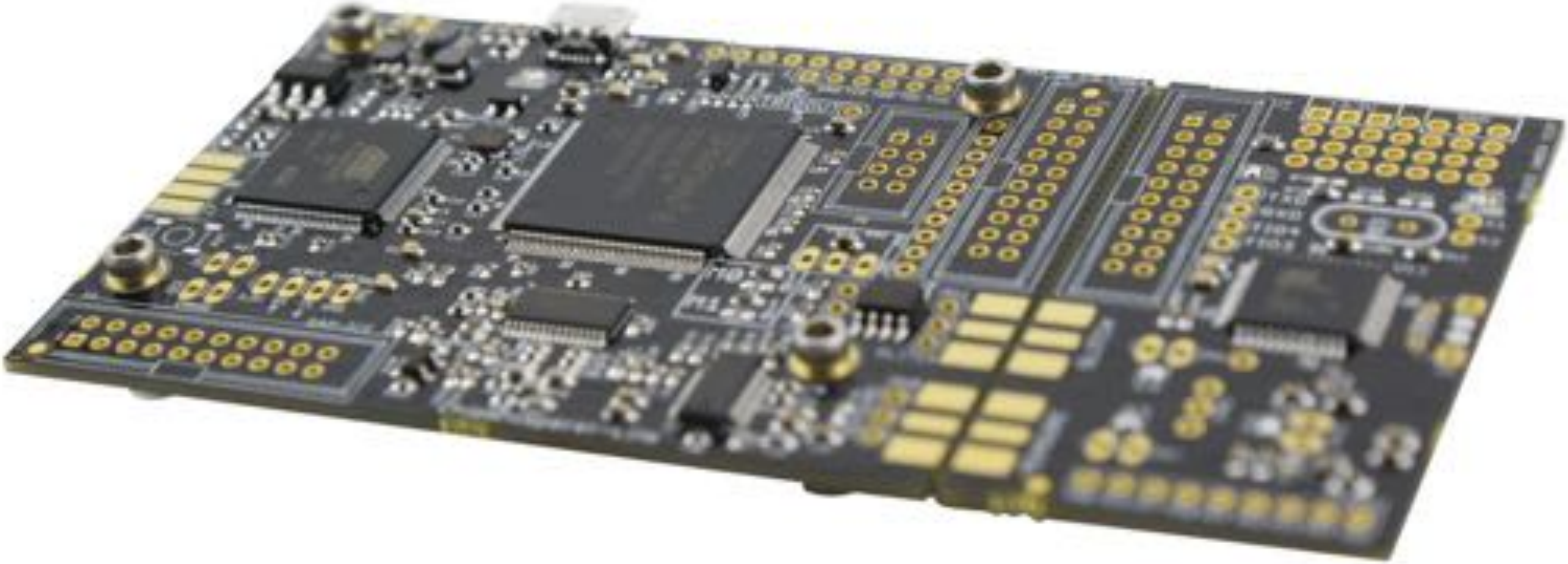
Fill in the **artificial divide** between hardware and software engineers.

SOMETIMES, WE HAVE TO KILL OUR HEROES.

Celebrate those who **lift others up**.
Fewer **rockstars**, more **teachers**.

OPEN A DOOR,
TEAR DOWN A BARRIER.

Produce more **entry-level materials**, and
build **more open, inexpensive tools**.



CHIPWHISPERER LITE GLITCHING & SIDE-CHANNEL BOARD

https://newae.com/tools/chipwhisperer/

https://github.com/newaetech/chipwhisperer

open source hardware

OPEN A DOOR, TEAR DOWN A BARRIER.

Don’t let educational spaces develop **additional barriers.**

DON’T TOLERATE
RACISM / SEXISM / ABLEISM / *PHOBIA
IN YOUR COMMUNITIES.

OPEN A DOOR, TEAR DOWN A BARRIER.

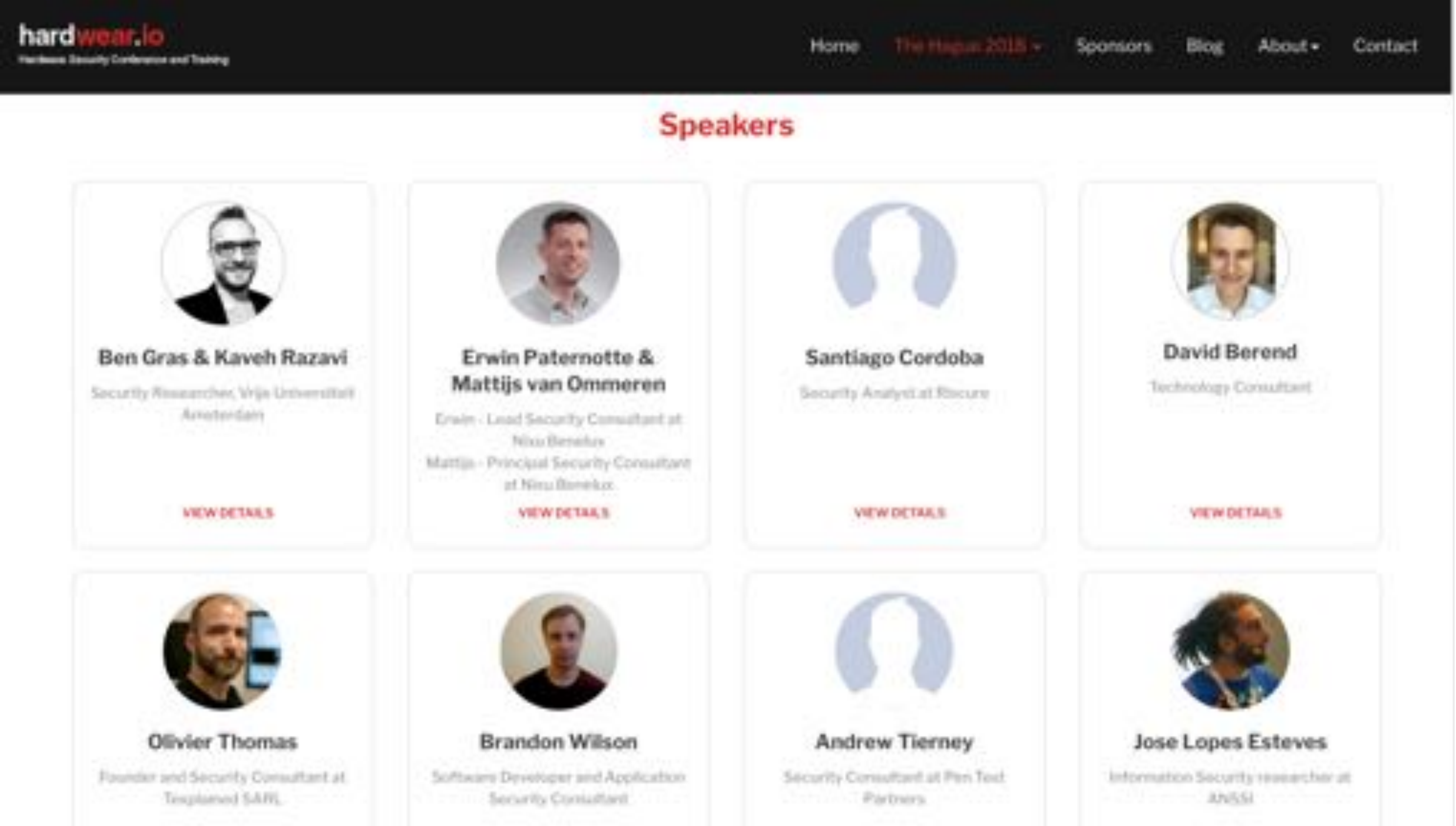
Don’t let educational spaces develop **additional barriers.**

AND FOR GOODNESS SAKE, STOP HIDING MY STUFF.

Vendors: hardware isn't just an implementation detail.

ONE MORE THING:

SO, WHY BRING THIS UP NOW?



Speakers

Ben Gras & Kaveh Razavi

Security Researcher, Vrije Universiteit Amsterdam

Erwin Paternotte & Mattijs van Ommeren

Erwin - Lead Security Consultant at Nixu Benelux

Mattijs - Principal Security Consultant at Nixu Benelux

Santiago Cordoba

Security Analyst at Riscure

David Berend

Technology Consultant

Olivier Thomas

Founder and Security Consultant at Texplained SARL

Brandon Wilson

Software Developer and Application Security Consultant

Andrew Tierney

Security Consultant at Pen Test Partners

Jose Lopes Esteves

Information Security researcher at ANSSI

QUESTIONS?

THANKS FOR LISTENING!

IMAGE CREDITS

- slide 6: nintendo switch icon by Sweet Farm from the Noun Project