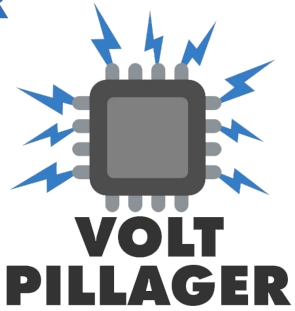


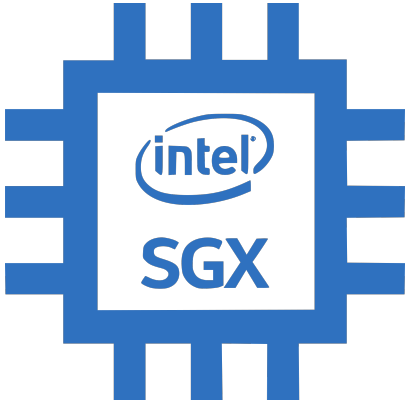
Plundering and Pillaging with Voltage: Software and Hardware-based Fault-injection Attacks against SGX.

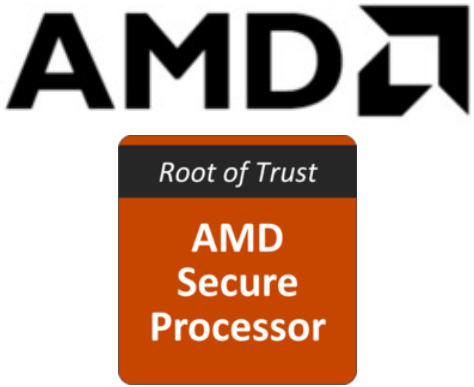
Flavio D. Garcia

Joint work with: Zitai Chen, Kit Murdock, Georgios Vasilakis, Edward Dean, David Oswald, Jo Van Bulck, Daniel Gruss, Frank Piessens

TEEs

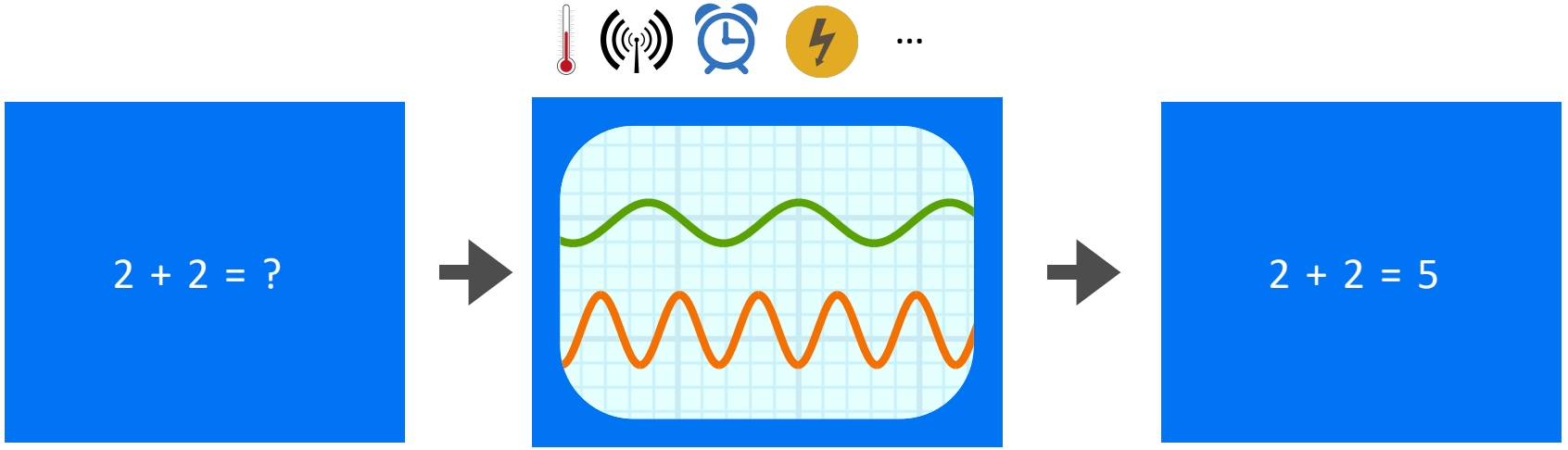

arm
TRUSTZONE

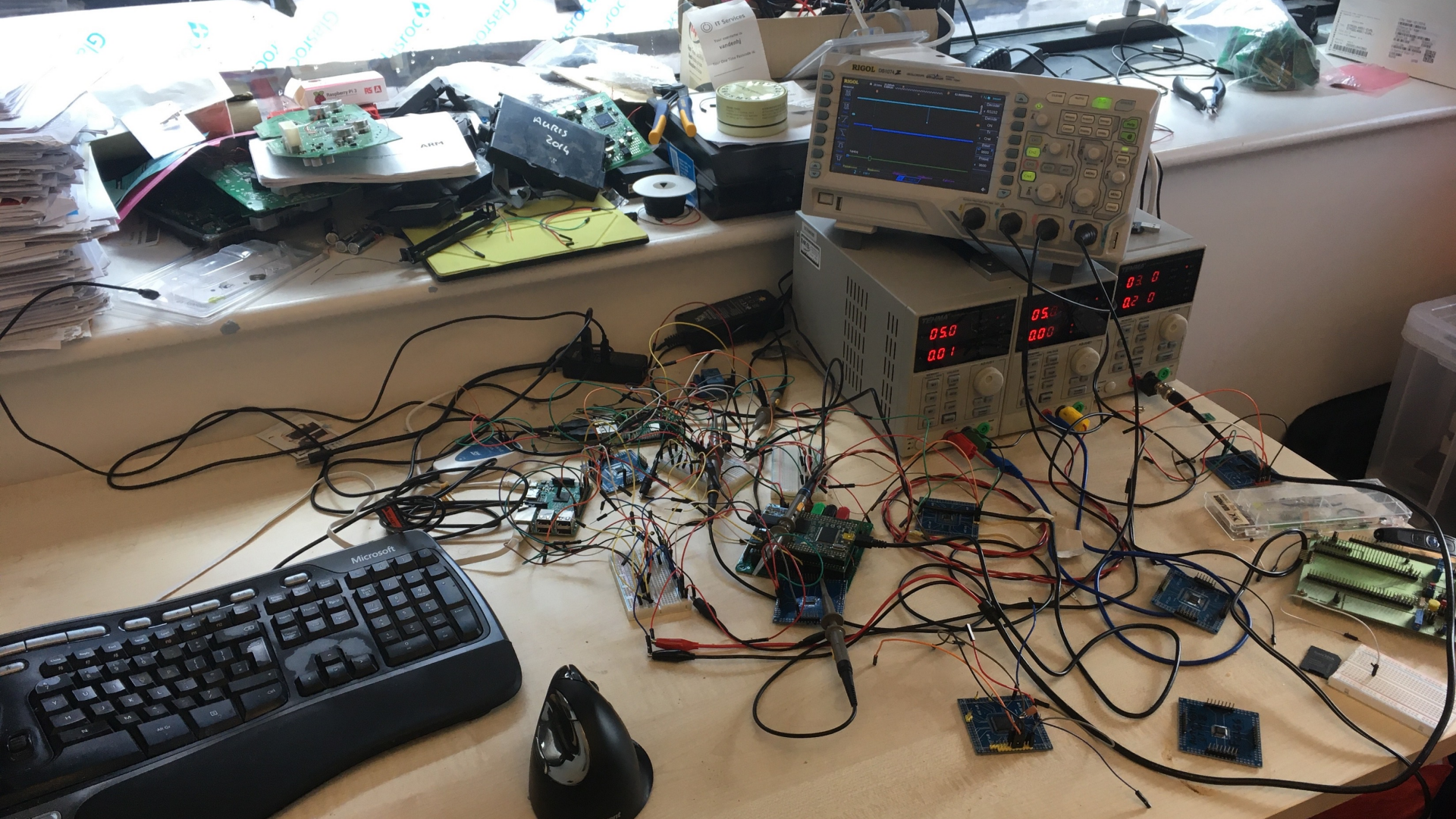

intel
SGX


AMD
Root of Trust
AMD
Secure
Processor


IBM Secure
Service Container

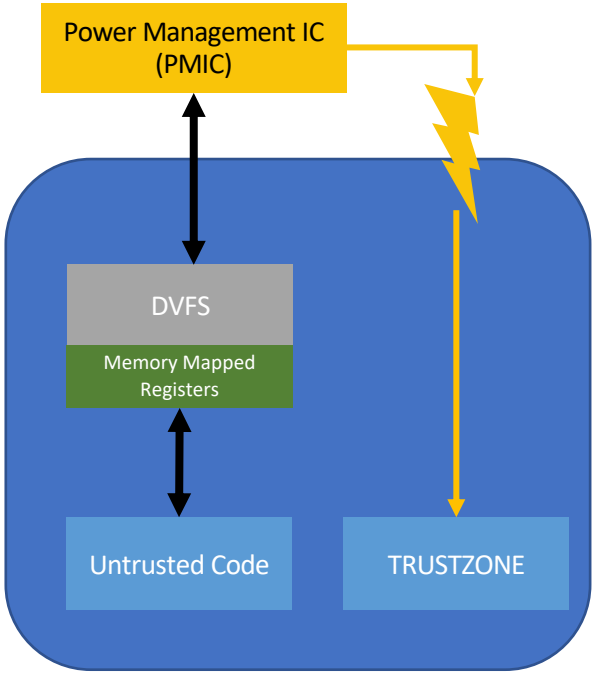
Fault Injection

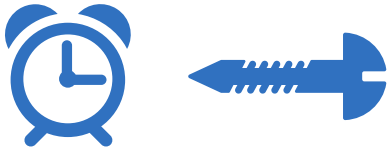

...
2 + 2 = ?
2 + 2 = 5


RIGOL
Microsoft
AURIS
2014
ARM

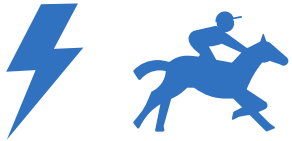
A new class of fault attacks

ARM SoC



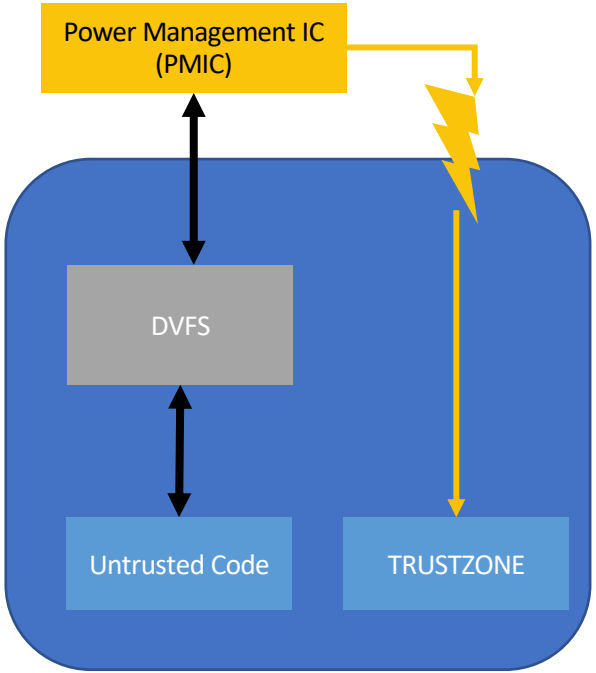


Adrian Tang et al. "CLKSCREW: exposing the perils of security-oblivious energy management"
In: USENIX Security Symposium. 2017

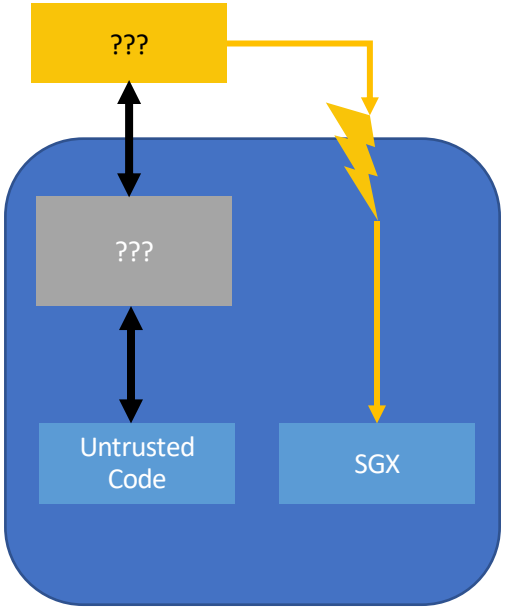


Pengfei Qiu et al. "VoltJockey: Breaching TrustZone by Software-Controlled Voltage Manipulation over Multi-core Frequencies"
In: CSS. 2019

ARM SoC



Intel



Overclocking

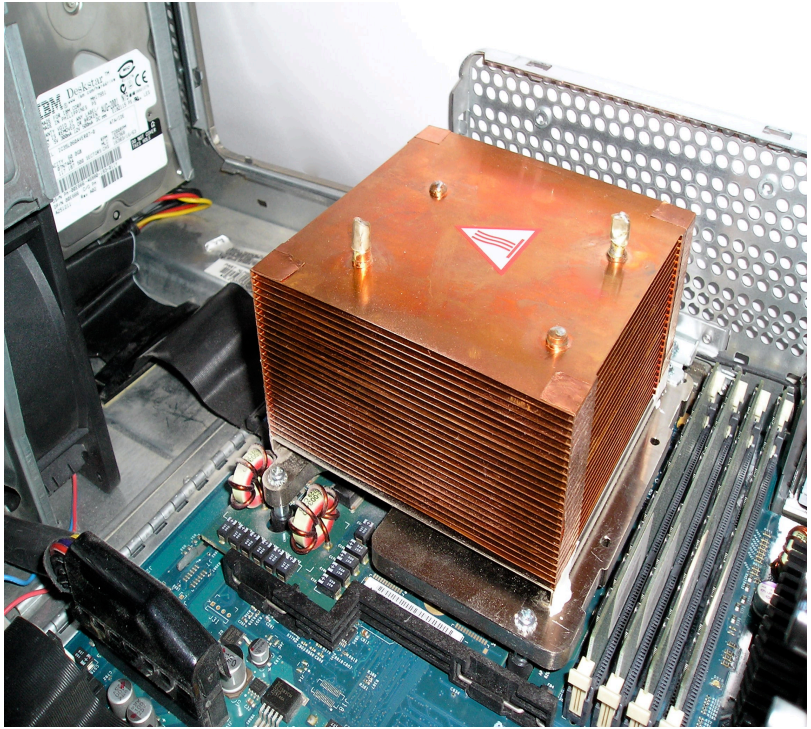


Image attribution: Charles Gaudette

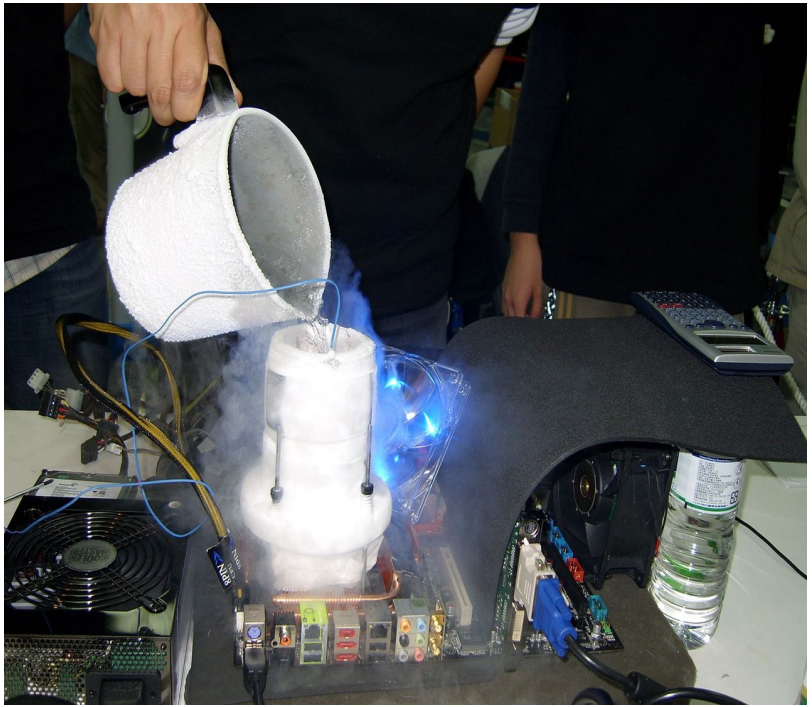
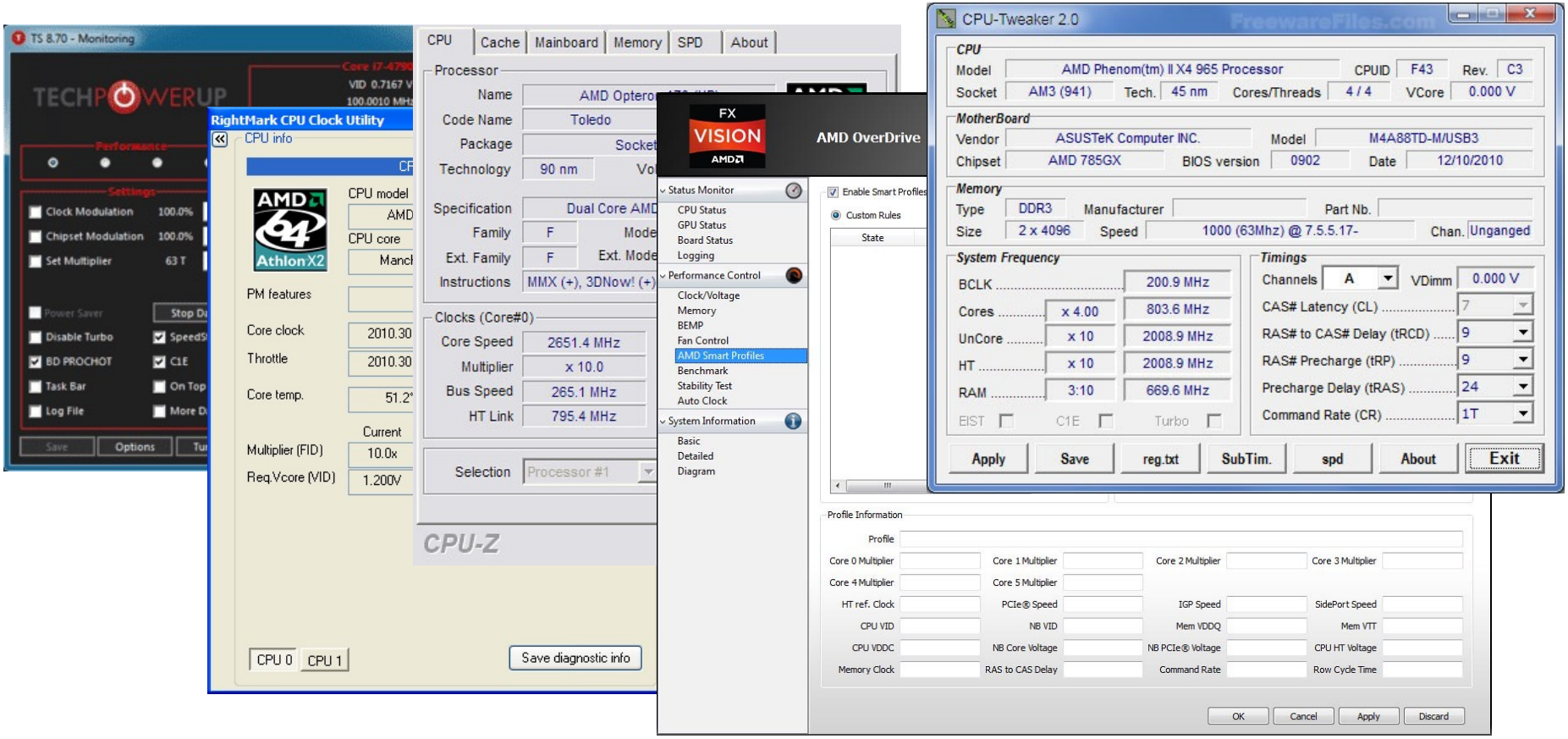


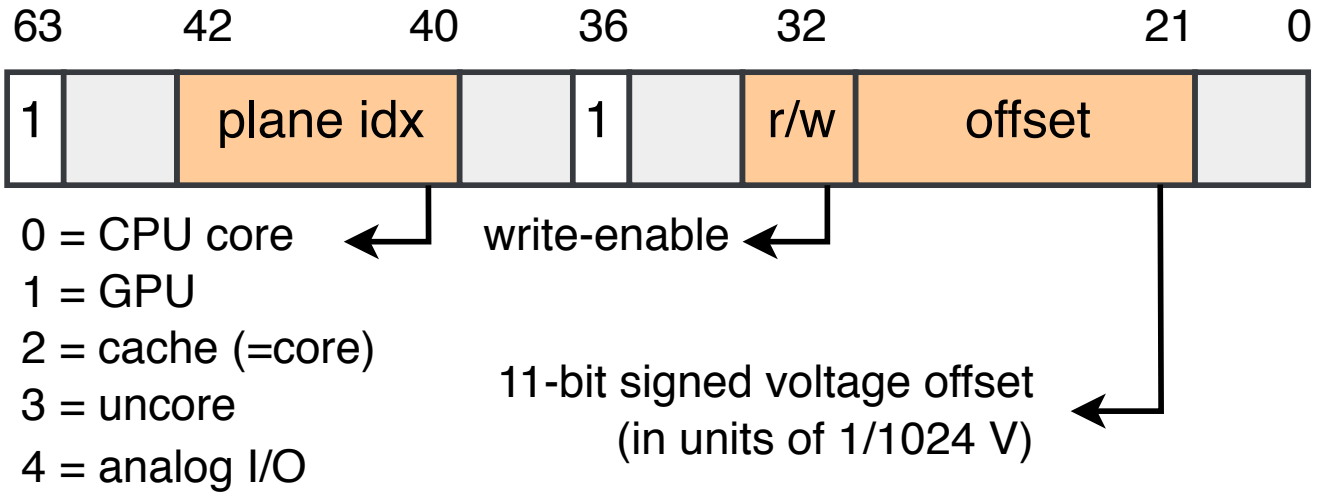
Image attribution: Rico Shen

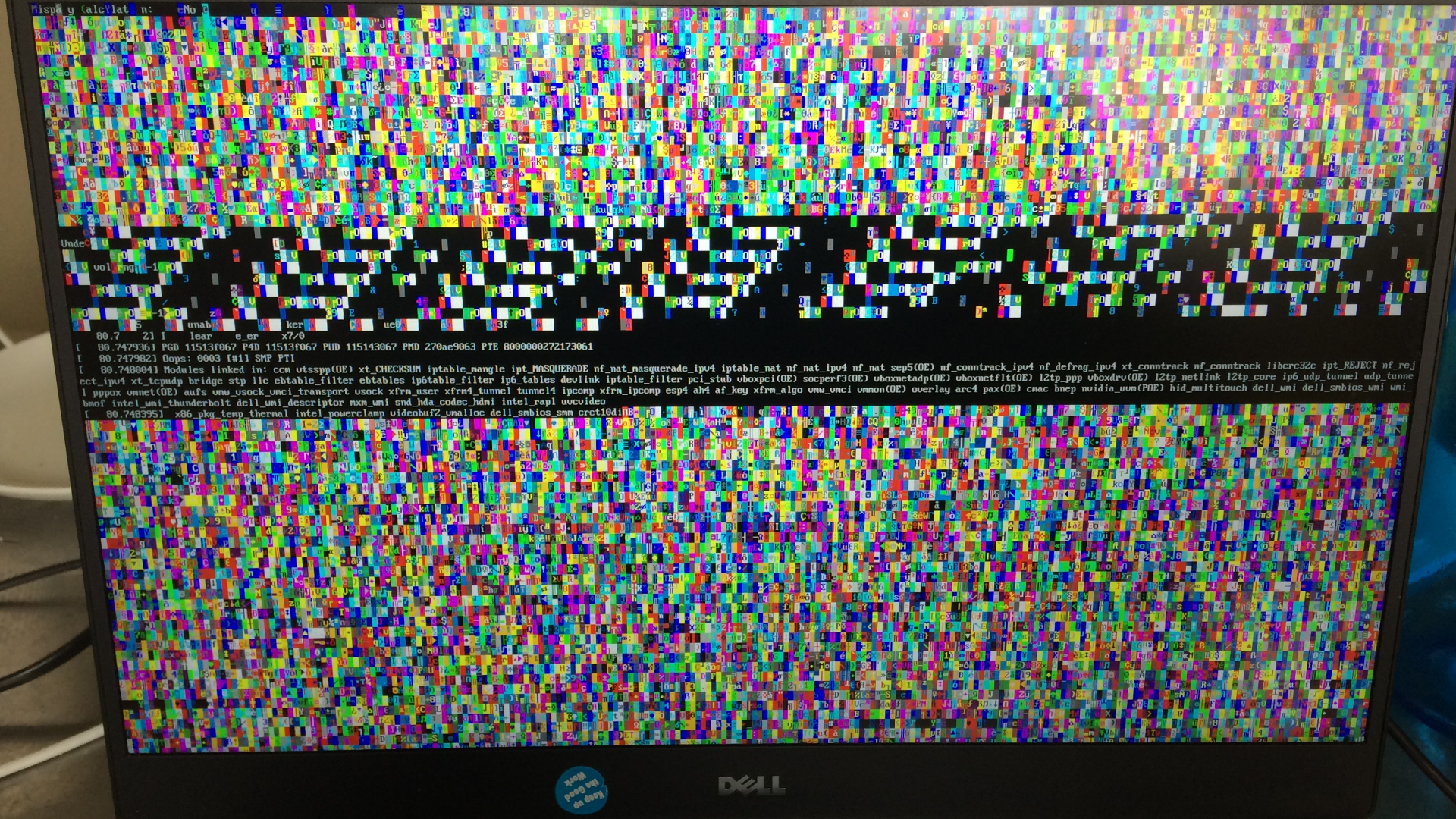


Plundering and Pillaging with Voltage: Software and Hardware-based Fault-injection Attacks against SGX

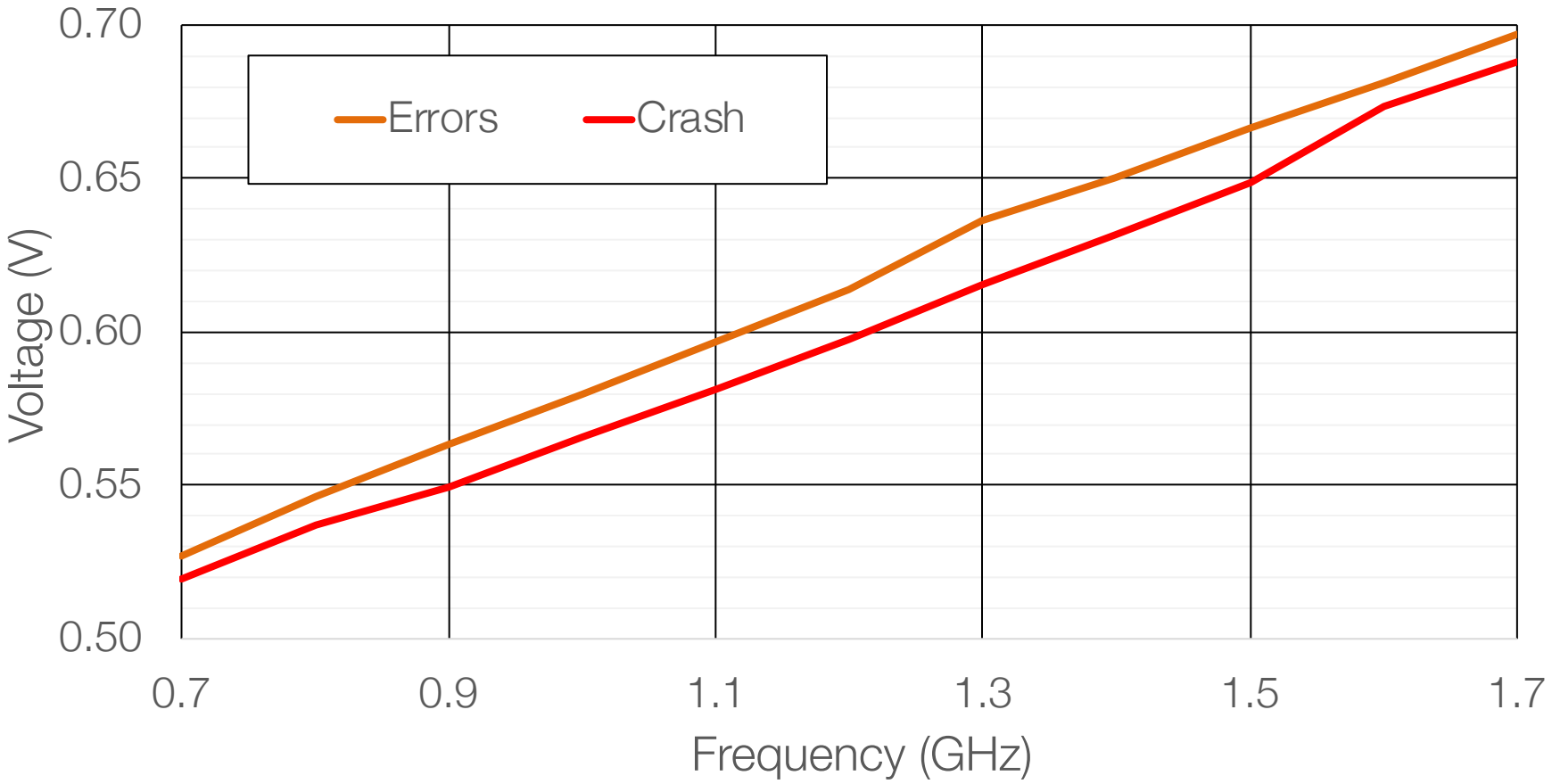
MSR 0x150

- Disable SVID
- Disable FIVR fault protection
- Disable FIVR efficiency management
- Set max clock ratio
- Set static voltage
- Set under/over voltage

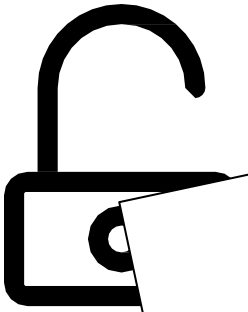

63
42
40
36
32
21
0
1
plane idx
1
r/w
offset
0 = CPU core
1 = GPU
2 = cache (=core)
3 = uncore
4 = analog I/O
write-enable
11-bit signed voltage offset
(in units of 1/1024 V)


80.7 2] I lear e_er x7/0
[80.747936] PGD 11513f067 P4D 11513f067 PUD 115143067 PMD 270ae9063 PTE 8000000272173061
[80.747982] Oops: 0003 [#1] SMP PTI
[80.748004] Modules linked in: ccm vtsspp(OE) xt_CHECKSUM iptable_mangle ipt_MASQUERADE nf_nat_masquerade_ipv4 iptable_nat nf_nat_ipv4 nf_nat sep5(OE) nf_conntrack_ipv4 nf_defrag_ipv4 xt_conntrack nf_conntrack libcrc32c ipt_REJECT nf_rej
ect_ipv4 xt_tcpudp bridge stp llc ebtable_filter ebtables ip6table_filter ip6_tables devlink iptable_filter pci_stub vboxpci(OE) socperf3(OE) vboxnetadp(OE) vboxnetflt(OE) l2tp_ppp vboxdrv(OE) l2tp_netlink l2tp_core ip6_udp_tunnel udp_tunne
l pppox vmnet(OE) aufs vmw_vsock_vmci_transport vsock xfrm_user xfrm4_tunnel tunnel4 ipcomp xfrm_ipcomp esp4 ah4 af_key xfrm_algo vmw_vmci vmmon(OE) overlay arc4 pax(OE) cmac bnep nvidia_uvm(POE) hid_multitouch dell_wmi dell_smbios_wmi wmi_
bmof intel_wmi_thunderbolt dell_wmi_descriptor mxm_wmi snd_hda_codec_hdmi intel_rapl uvcvideo
[80.748395] x86_pkg_temp_thermal intel_powerclamp videobuf2_vmalloc dell_smbios_smm crct10dif

Error and crash voltages – Intel(R) Core(TM) i3-7100U CPU



RSA-CRT



- Public Key Cryptography
- Untrusted channel
- Encryp

,p,q

Many RSA implementations use the Chinese Remainder Theorem optimisation

Lenstra Attack - 1996

- Requires a fault in one of the two exponentiations
- Improves upon the Bellcore attack
    - Only requires one faulty result

$$q = \gcd((x')^e - y, n) \quad p = \frac{n}{q}$$

```
// Start undervolting
uint8_t rsa_dec_ecall(int iterations)
{
    //Wait for first fault
    trigger_fault(iterations);

    //Actual decryption
    ippsRSA_Decrypt(ct,dec,pPrv,scratchBuffer);
}
// Stop undervolting
```

```
bagger> dog Enclave/encl
```

AES-NI

Instruction	Description
`AESENC`	Perform one round of an AES ener…
`AESENCLAST`	Perform the la…
`AESDEC`	
	…eration
	…AES Inverse Mix Columns
	Carryless multiply (CLMUL)

AES can be attacked if you get a fault in the 8th round.

When a single random byte fault is induced at the input of the eighth round, the AES key can be deduced.

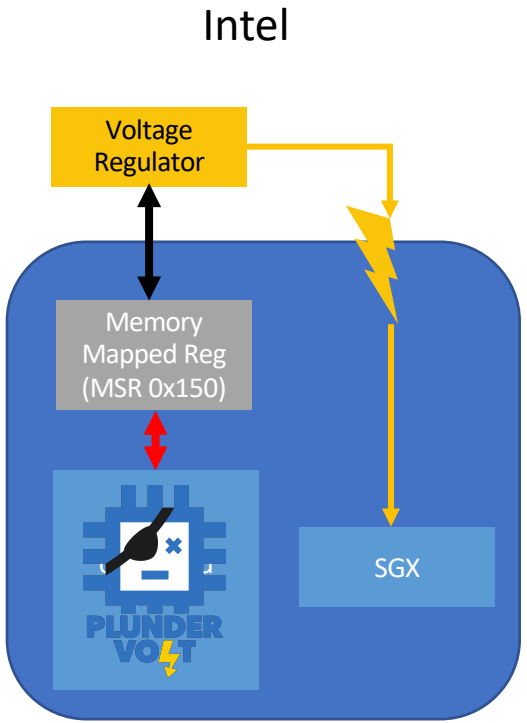
The computation complexity to recover 128 bit key is: $2^{32} + 256$ encryptions.

```
// Start undervolting

do
{
        plaintext= <randomlygenerated>;
        result1=aes128_encryption(plaintext);
        result2=aes128_encryption(plaintext);

} while(result1 == result2)

// Stop undervolting
```

```
bagger> sudo ./aes-encrypt 100000 -262
```

- Faulting Multiplication
- Faulting RSA in SGX
- Faulting AES-NI in SGX
- Memory Corruption

Kit Murdock et al. Plundervolt: Software-based Fault Injection Attacks against Intel SGX
In: 41st IEEE Symposium on Security and Privacy (S&P'20)



Intel’s response

Summary:

A potential security vulnerability in some Intel® Processors may allow escalation of privilege and/or information disclosure. Intel has released firmware updates to system manufacturers to mitigate this potential vulnerability

Vulnerability Details:

CVE-2019-11157

Description: Improper conditions check in voltage settings for some Intel(R) Processors may allow a privileged user to potentially enable escalation of privilege and/or information disclosure via local access.

CVSS Base Score: 7.9 High

CVSS Vector: CVSS:3.1/AV:L/AC:L/PR:H/UI:N/S:C/C:H/I:H/A:N

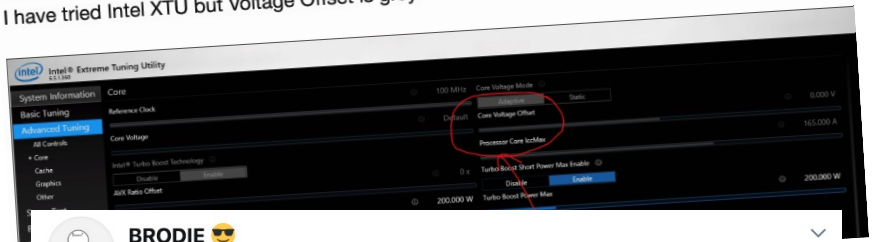
Recommendations:

Intel recommends that users of the above Intel® Processors update to the latest BIOS version provided by the system manufacturer that addresses these issues.

Intel is conducting an SGX TCB recovery. Refer to Intel® SGX Attestation Technical Details for more information.

you can not undervolt the new 10th gen CPUs because of the new plunderVolt vulnerability check it out here https://plundervolt.com/ the underVolt is disabled and you need to check if your laptop manufacturer has the intel xtu option in the advanced section of the bios this applies to 6th 7th 8th 9th 10th gen Intel CPUs but undervolting is only disabled on the 10th gen ⚠️ (Do this At your own Risk) ⚠️ (Warning) ⚠️

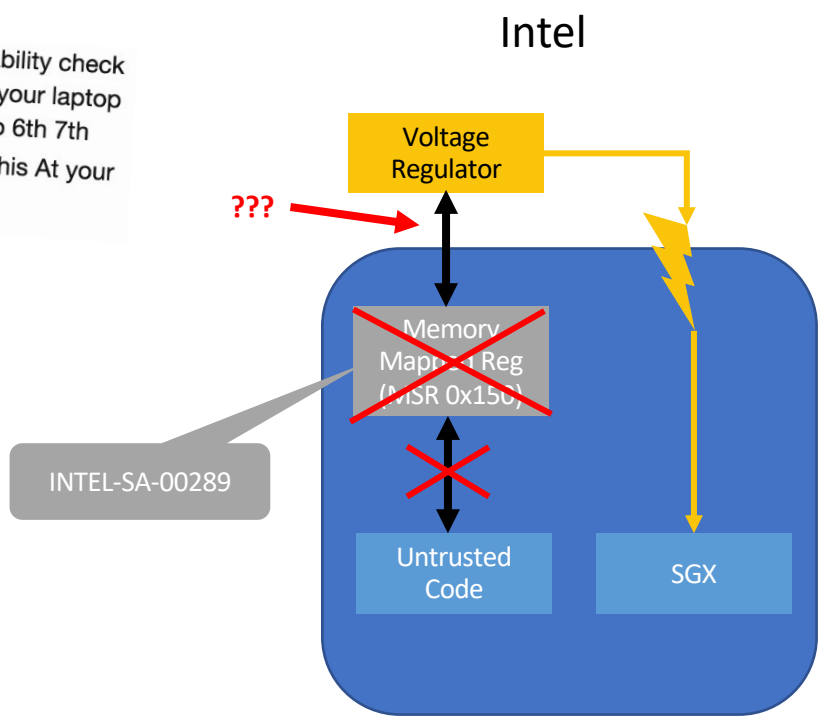
I can't find any way to Undervolt CPU i9 10980hk on my MSI GE75 Raider laptop.
I have tried Intel XTU but Voltage Offset is grey.



BRODIE 😎
@vaughnondrej16

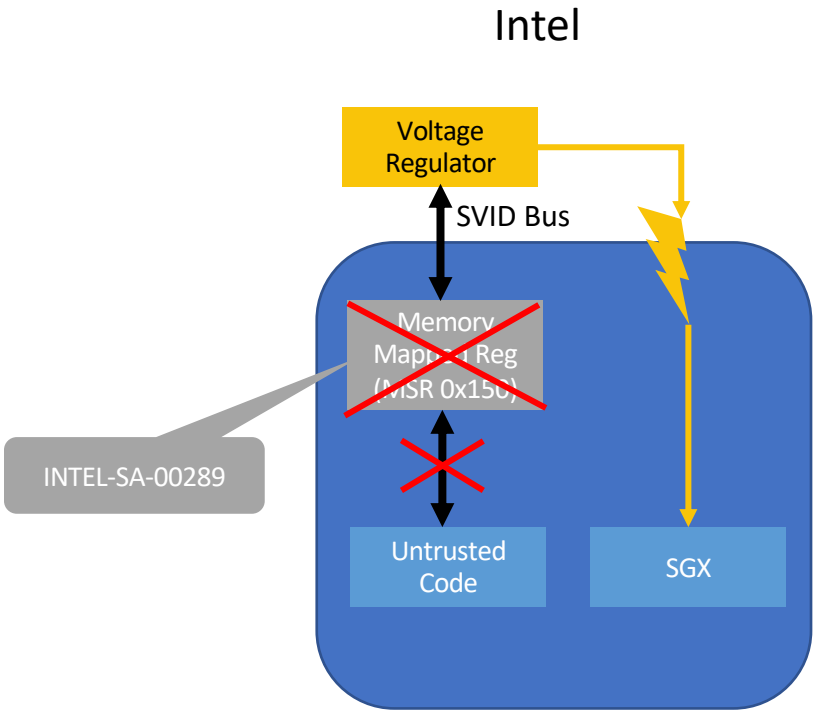
@intel, when will the Plundervolt issue be fixed? I don't like it when my CPU temps are going 85°C and up because I can't undervolt it.

2:21 PM · Sep 1, 2020 · Twitter for iPhone



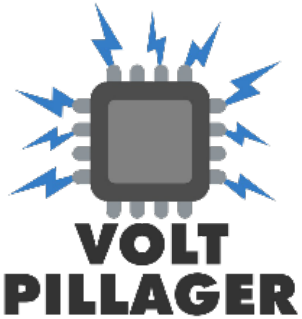
SVID Bus

- 3 Wire interface
  - CLK, DATA and ALERT(Not required)
- Clock @ 25MHz
- Logical High >0.64V, Low <0.45V

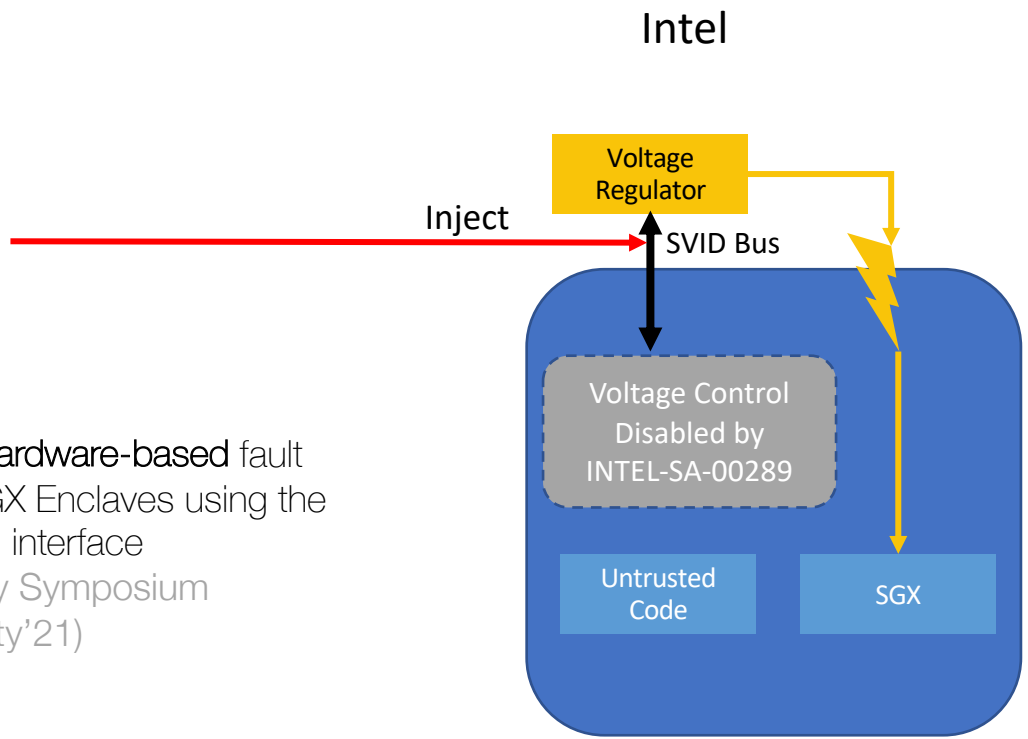


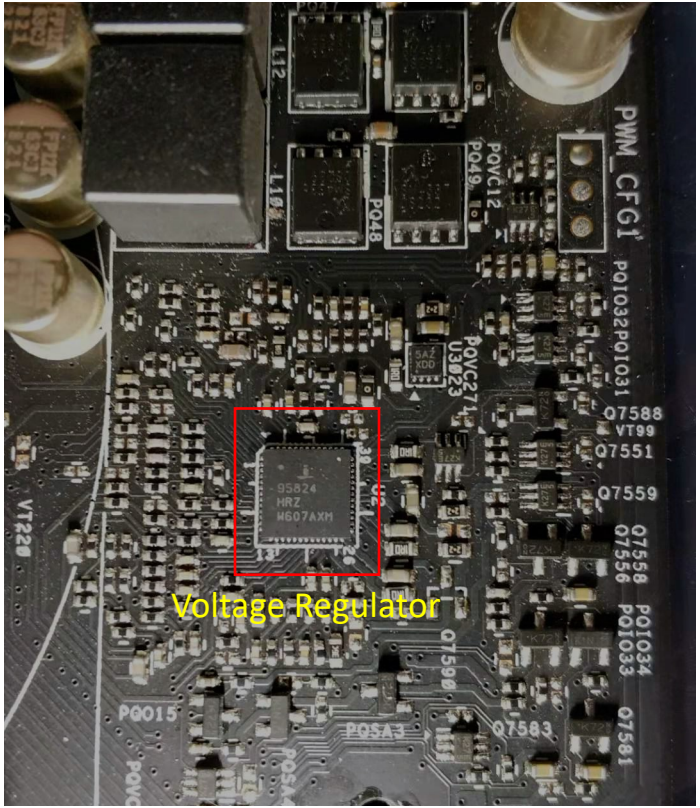
Ref:
1. L6751C Digitally controlled dual PWM for Intel VR12 and AMD SVI
2. 8th Generation Intel® CoreTM Processor Families Datasheet, Volume 1 of 2

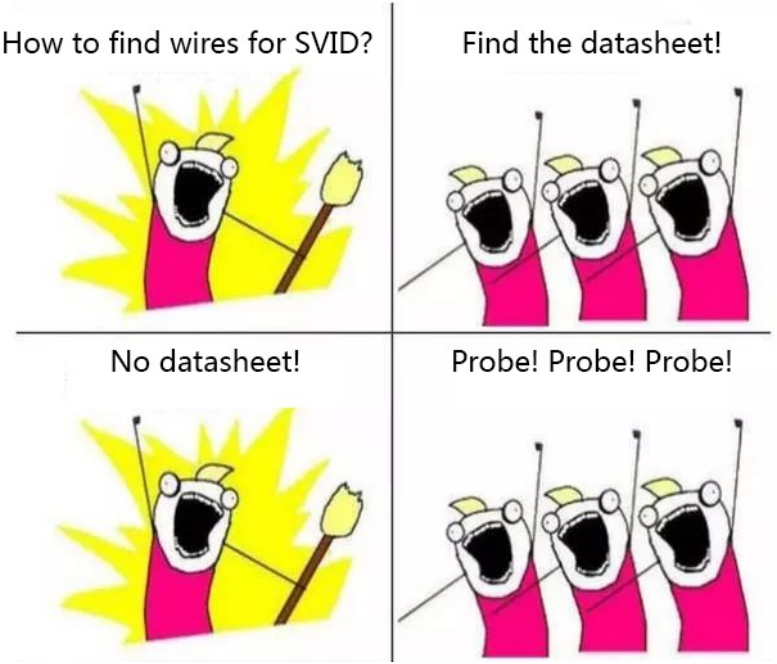
VOLT PILLAGER



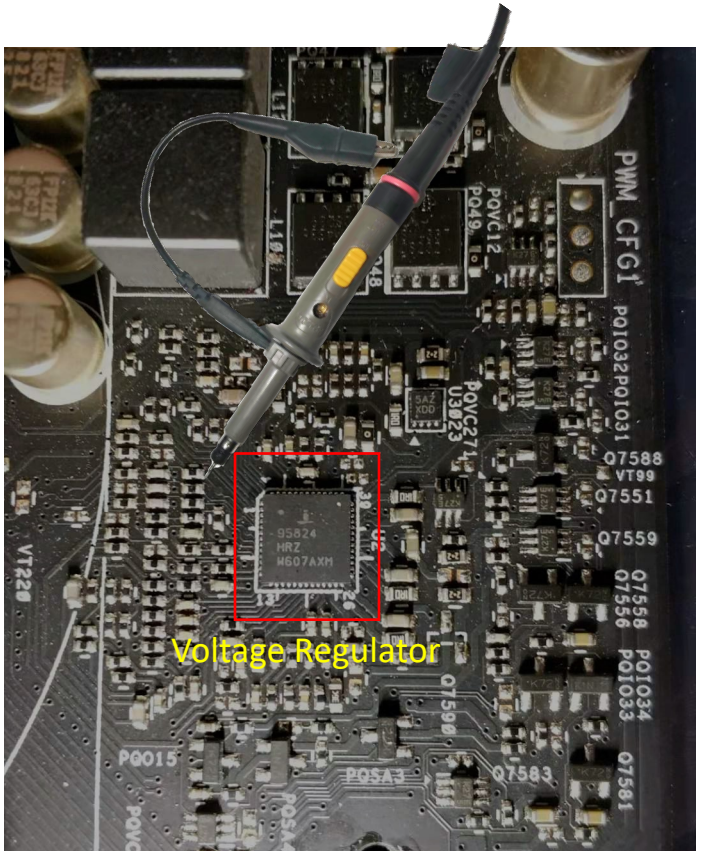
Zitai Chen et al. Voltpillager: Hardware-based fault injection attacks against Intel SGX Enclaves using the SVID voltage scaling interface
In: 30th USENIX Security Symposium (USENIX Security'21)





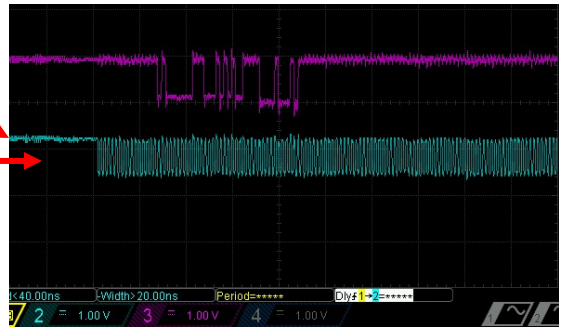


VOLT PILLAGER

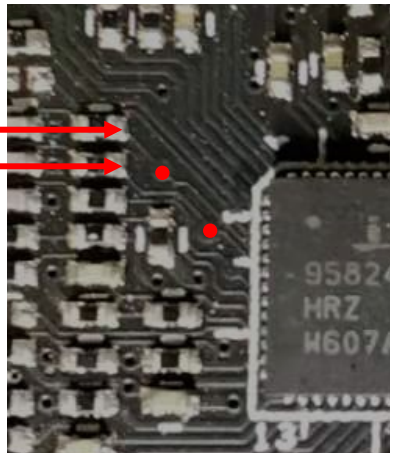


1V

25MHz

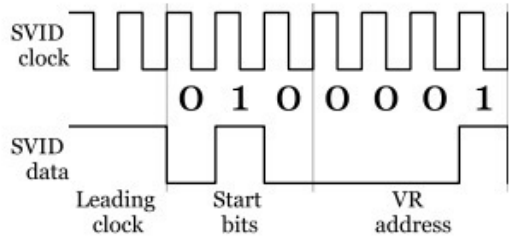


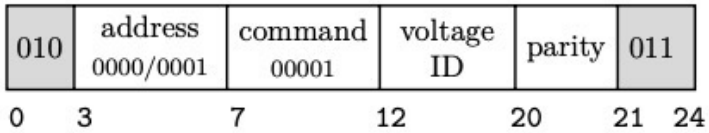
SVID Bus

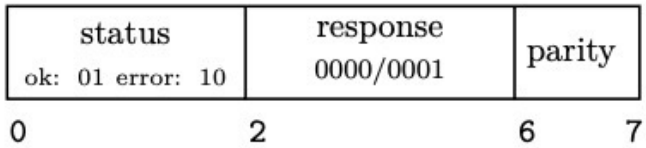


VOLT PILLAGER

SVID signals and data frame







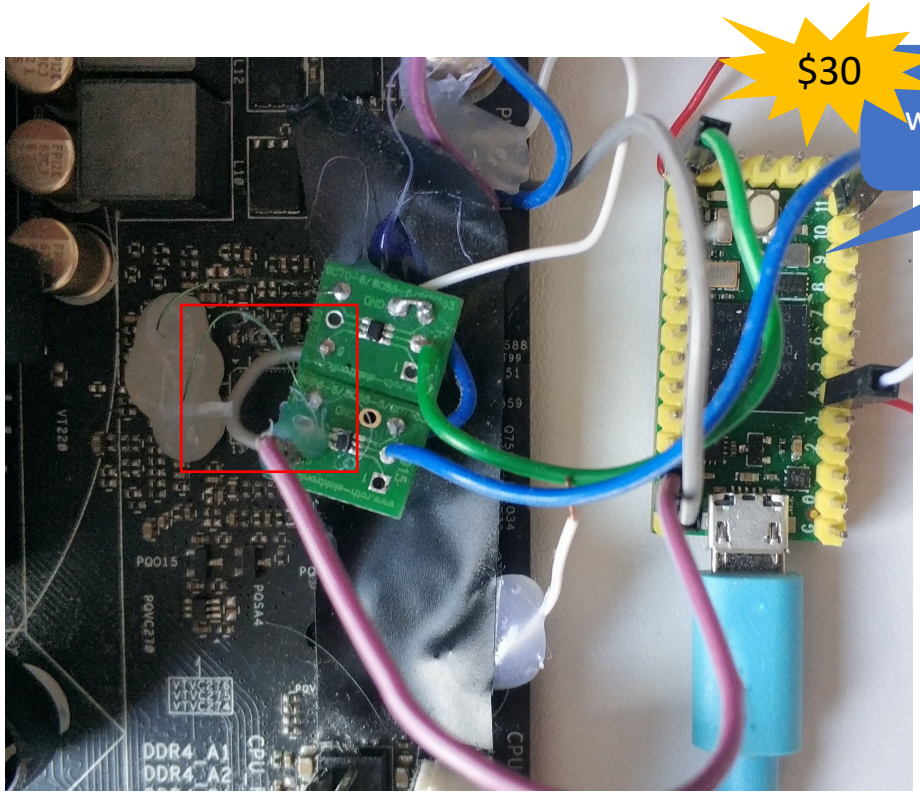
VID：1byte, computed as (voltage U in volt):

$$\mathrm{VID} = \left\lfloor \frac{U - 0.245}{0.005} \right\rfloor$$

VID Commands: 5bits

Command name	Value
Extended	0x00
SetVID-Fast	0x01
SetVID-Slow	0x02
SetVID-Decay	0x03
SetPS	0x04
SetRegADR	0x05
SetRegDAT	0x06

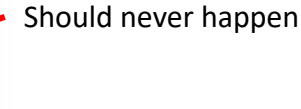
VOLT PILLAGER



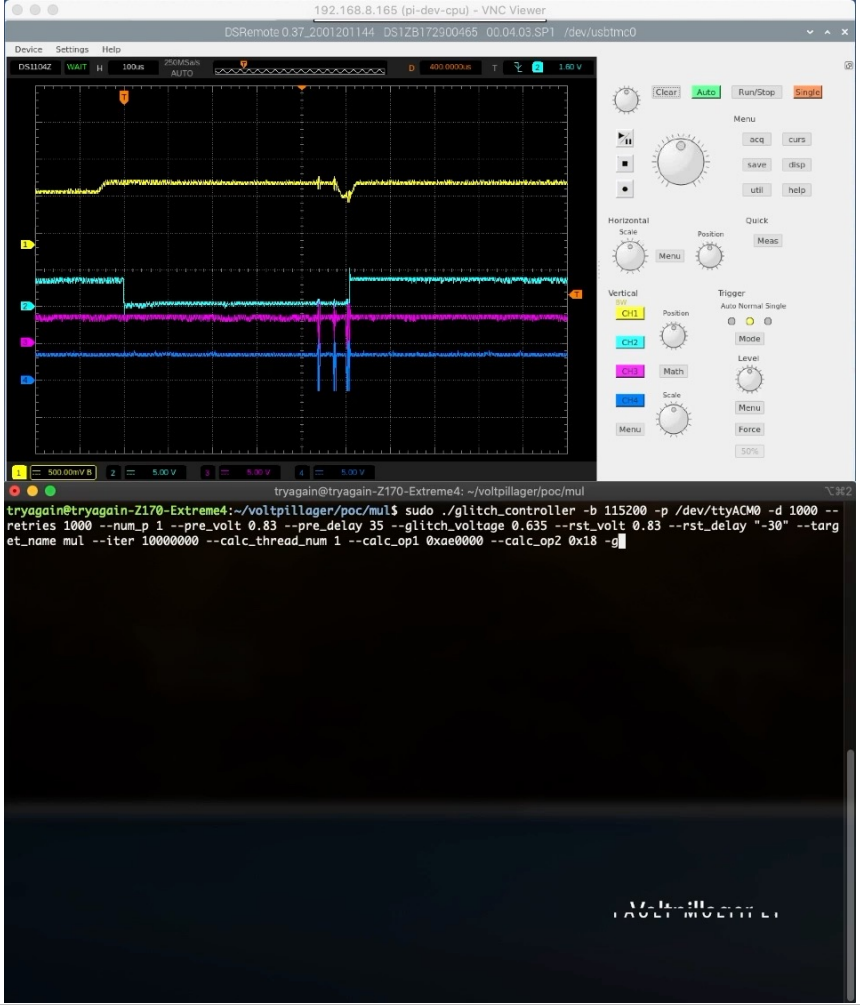
VOLT PILLAGER

Plundering and Pillaging (demos)

```
1  TRIGGER_SET // Set trigger
2  do {
3      i++;
4      correct_a = operand1 * operand2;
5      correct_b = operand1 * operand2;
6      if (correct_a != correct_b) {
7          faulty = 1;
8      }
9  } while (faulty == 0 && i < iterations);
10 TRIGGER_RST // Reset trigger
```



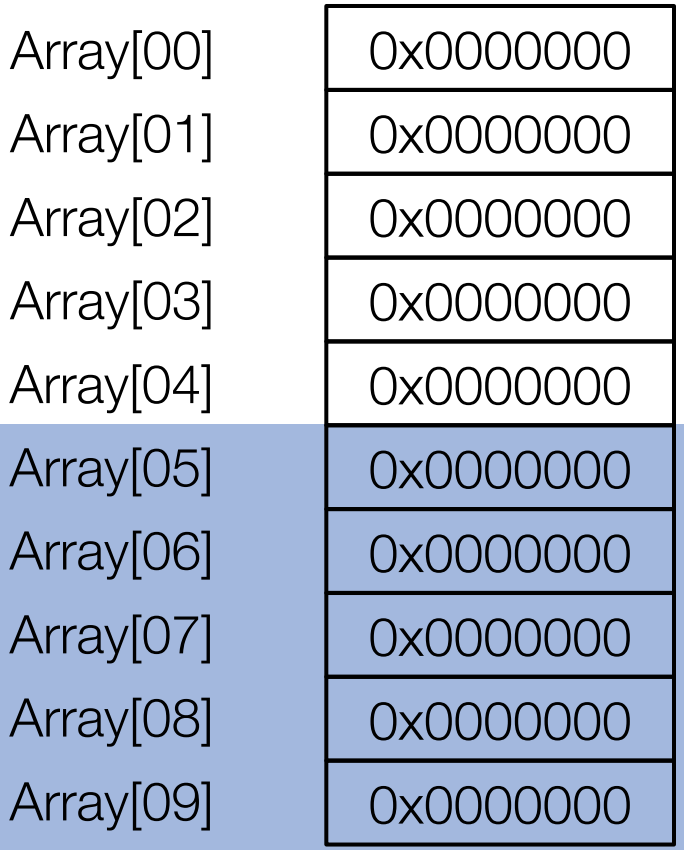
VOLT PILLAGER



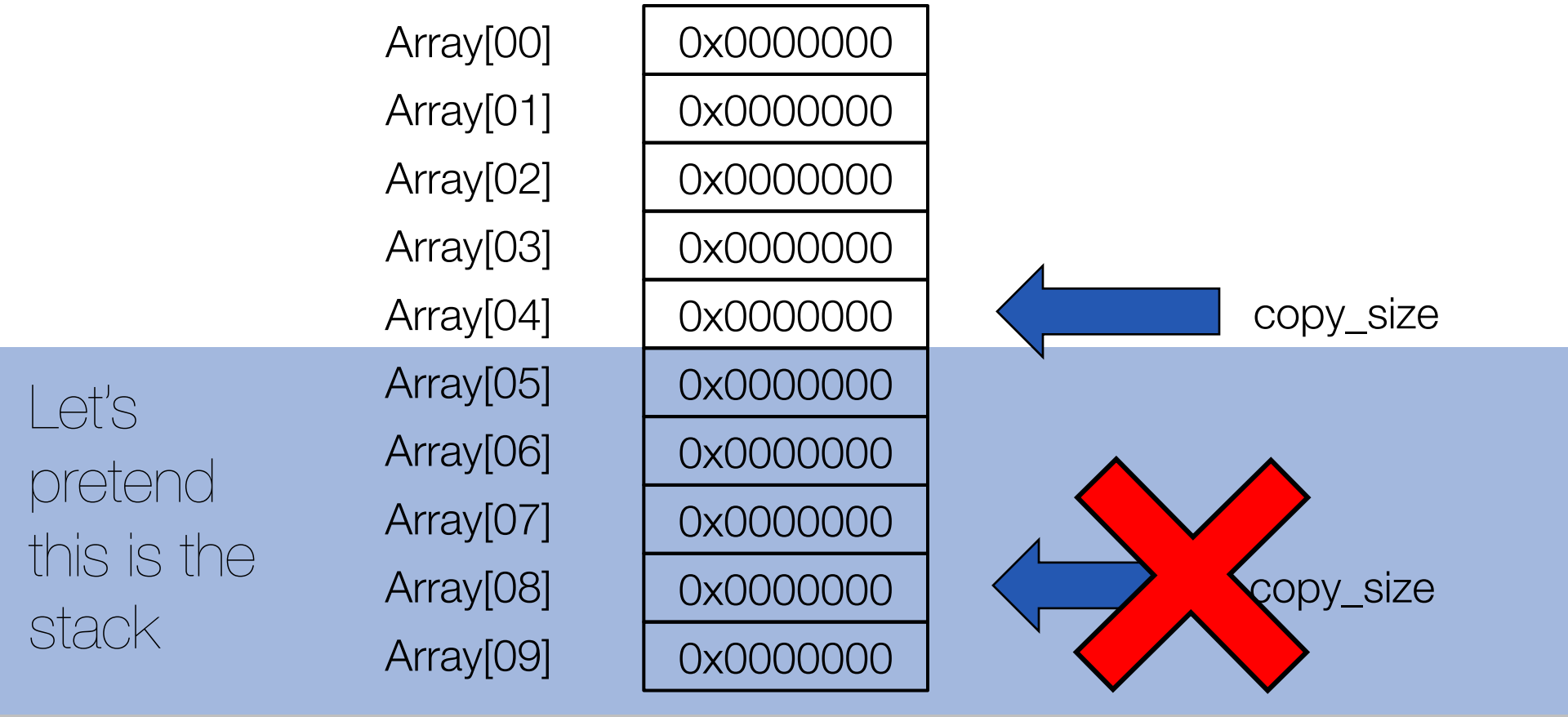
VOLT PILLAGER

Array bounds

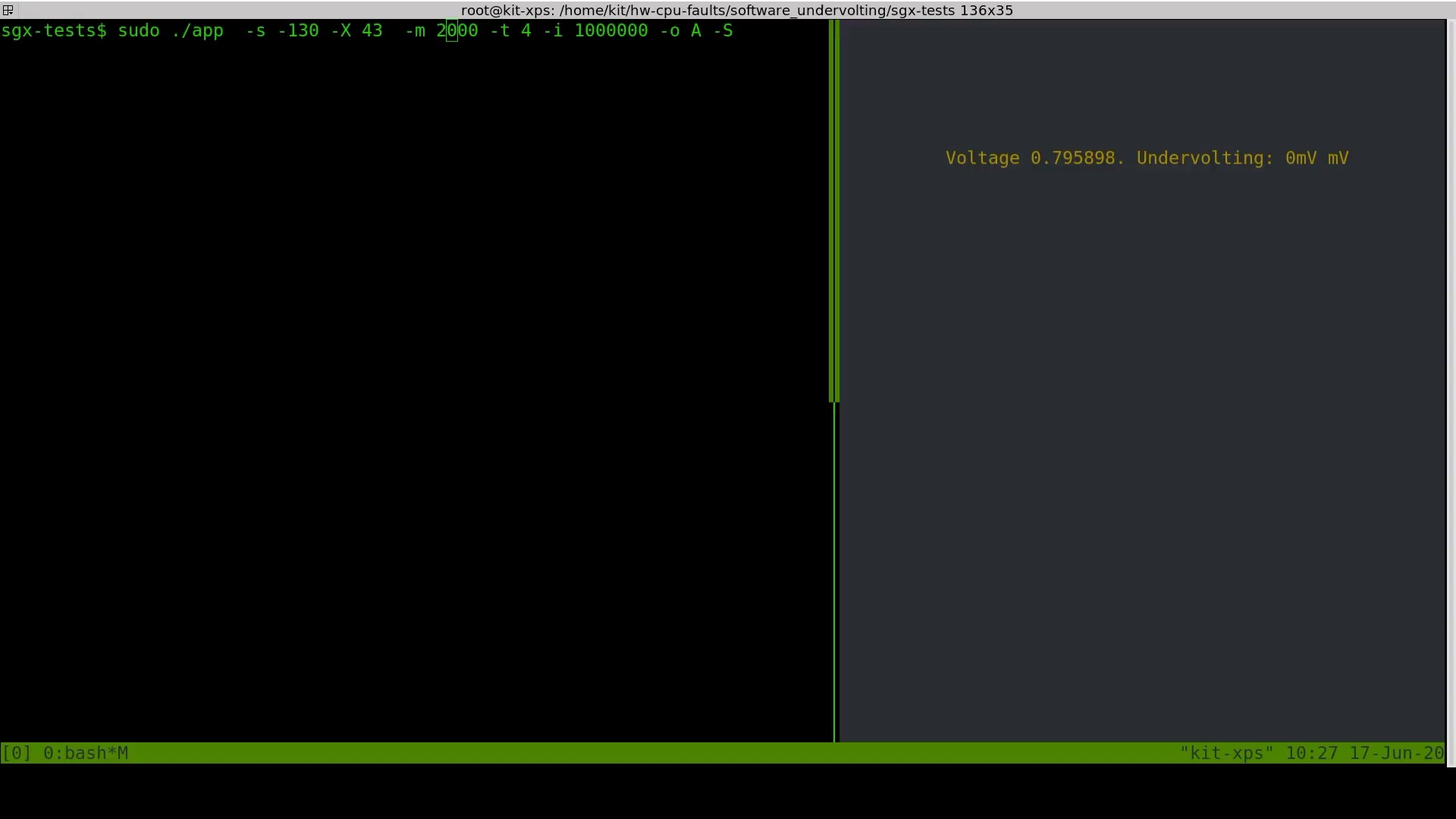
Let's pretend this is the stack



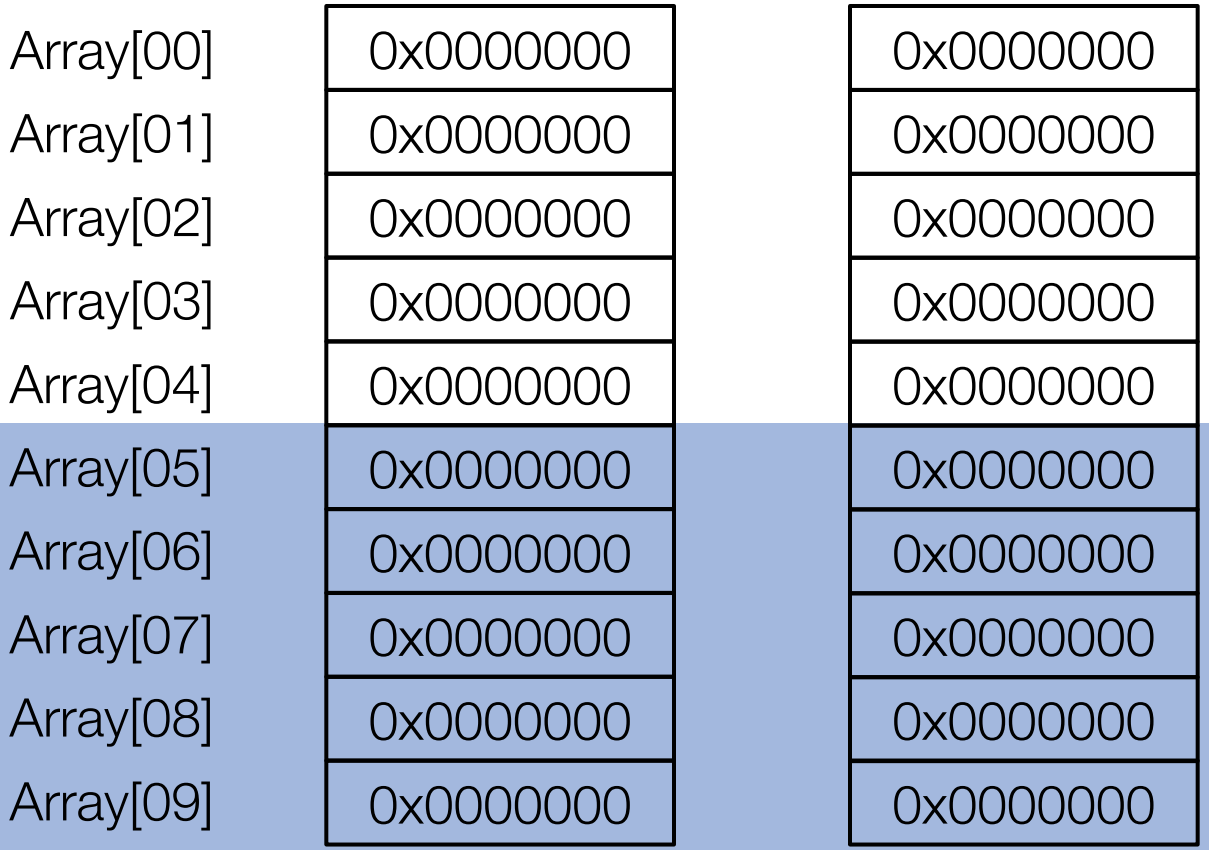
```
const int ARRAY_SIZE_MAX=10;
const int ARRAY_SIZE_TEST=5;
int copy_size=0;
uint32_t array[ARRAY_SIZE_MAX]={0};
do
{
        copy_size = ARRAY_SIZE_TEST + 3;

        // If an attacker tries to write beyond the array – prevent it
        if (copy_size >= ARRAY_SIZE_TEST)
           copy_size = ARRAY_SIZE_TEST - 1;
```


Array[00]
Array[01]
Array[02]
Array[03]
Array[04]
Array[05]
Array[06]
Array[07]
Array[08]
Array[09]
0x0000000
0x0000000
0x0000000
0x0000000
0x0000000
0x0000000
0x0000000
0x0000000
0x0000000
0x0000000
copy_size
Let's pretend this is the stack
copy_size

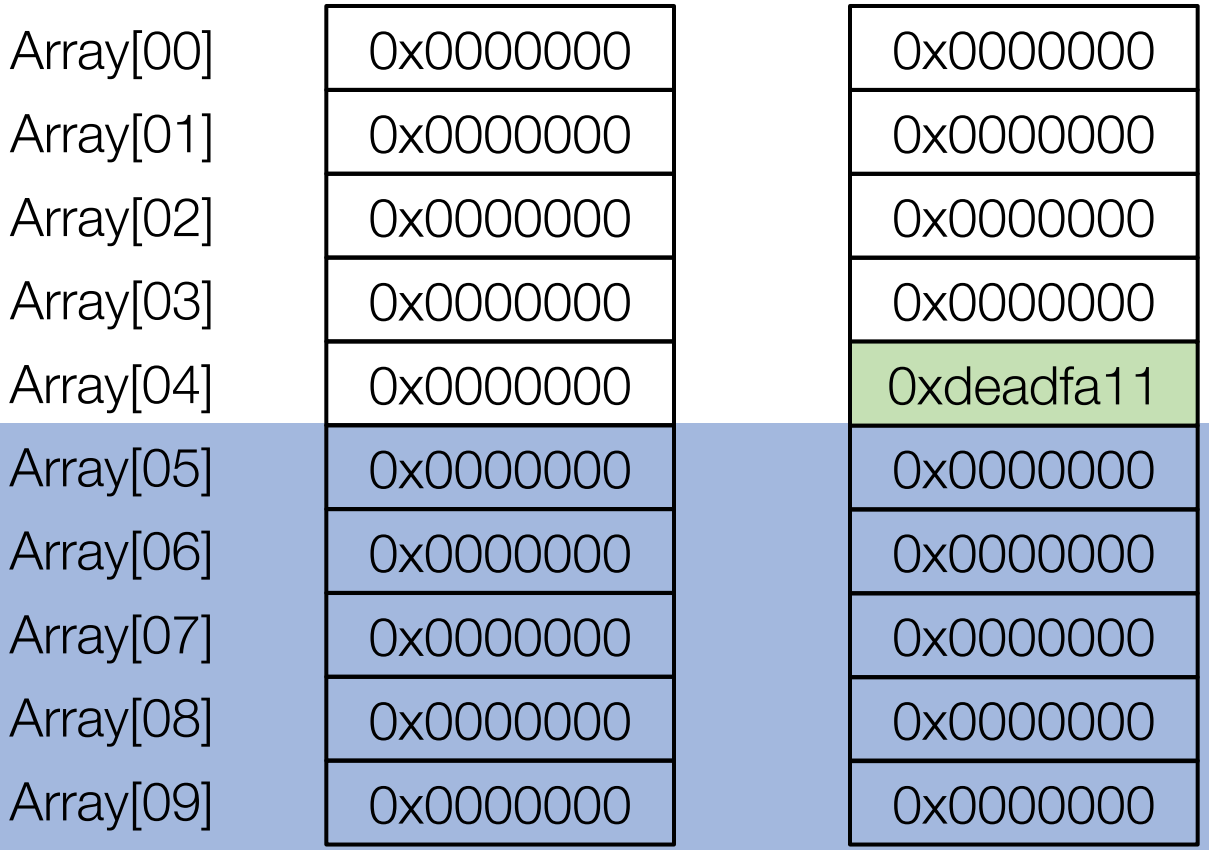
```
const int ARRAY_SIZE_MAX=10;
const int ARRAY_SIZE_TEST=5;
int copy_size=0;
uint32_t array[ARRAY_SIZE_MAX]={0};
do
{
        copy_size = ARRAY_SIZE_TEST + 3;

        // If an attacker tries to write beyond the array – prevent it
        if (copy_size >= ARRAY_SIZE_TEST)
           copy_size = ARRAY_SIZE_TEST - 1;

        // Overwrite elements  4...1
        while (copy_size >= 1)
        {
           array[copy_size] = 0xdeadfa11;
           copy_size--;
        }
} while the array is accurate
```


root@kit-xps: /home/kit/hw-cpu-faults/software_undervolting/sgx-tests 136x35
sgx-tests$ sudo ./app -s -130 -X 43 -m 2000 -t 4 -i 1000000 -o A -S
Voltage 0.795898. Undervolting: 0mV mV
[0] 0:bash*M
"kit-xps" 10:27 17-Jun-20

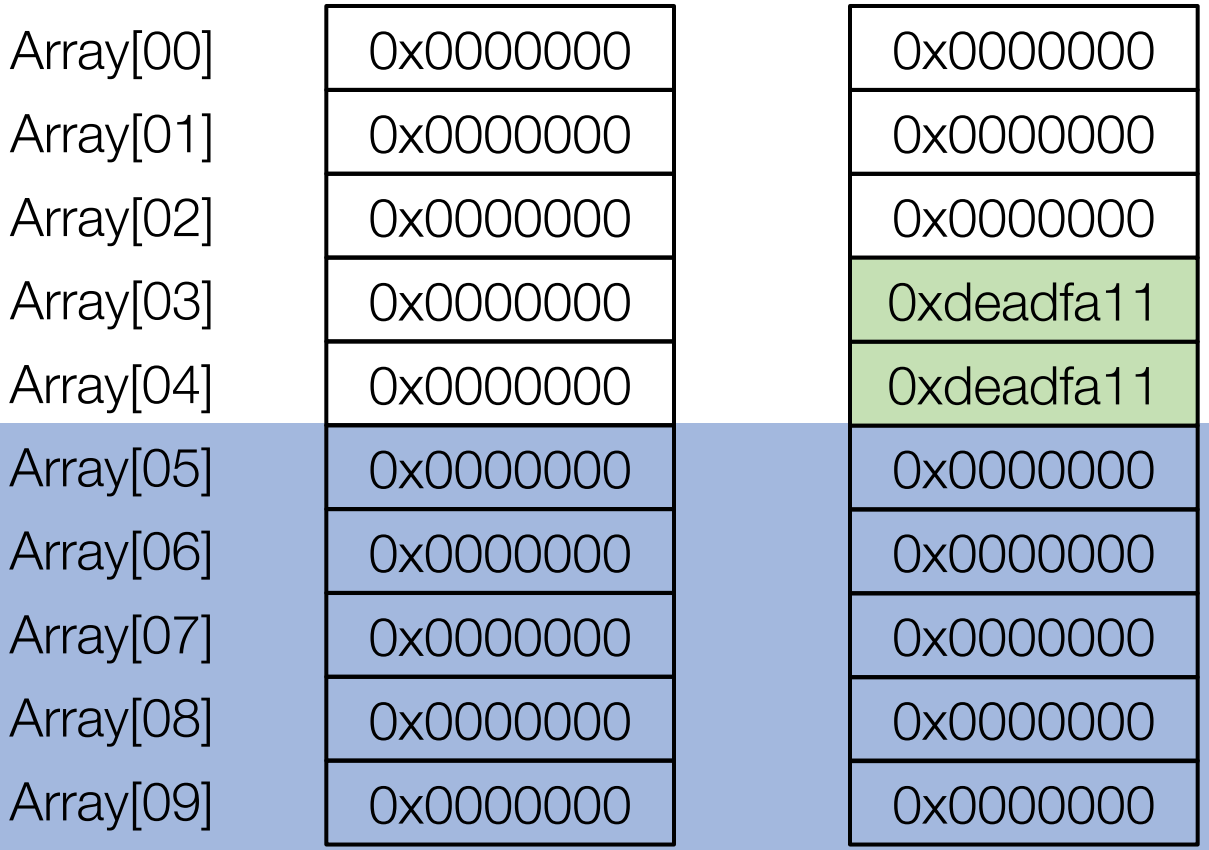
Let's pretend this is the stack



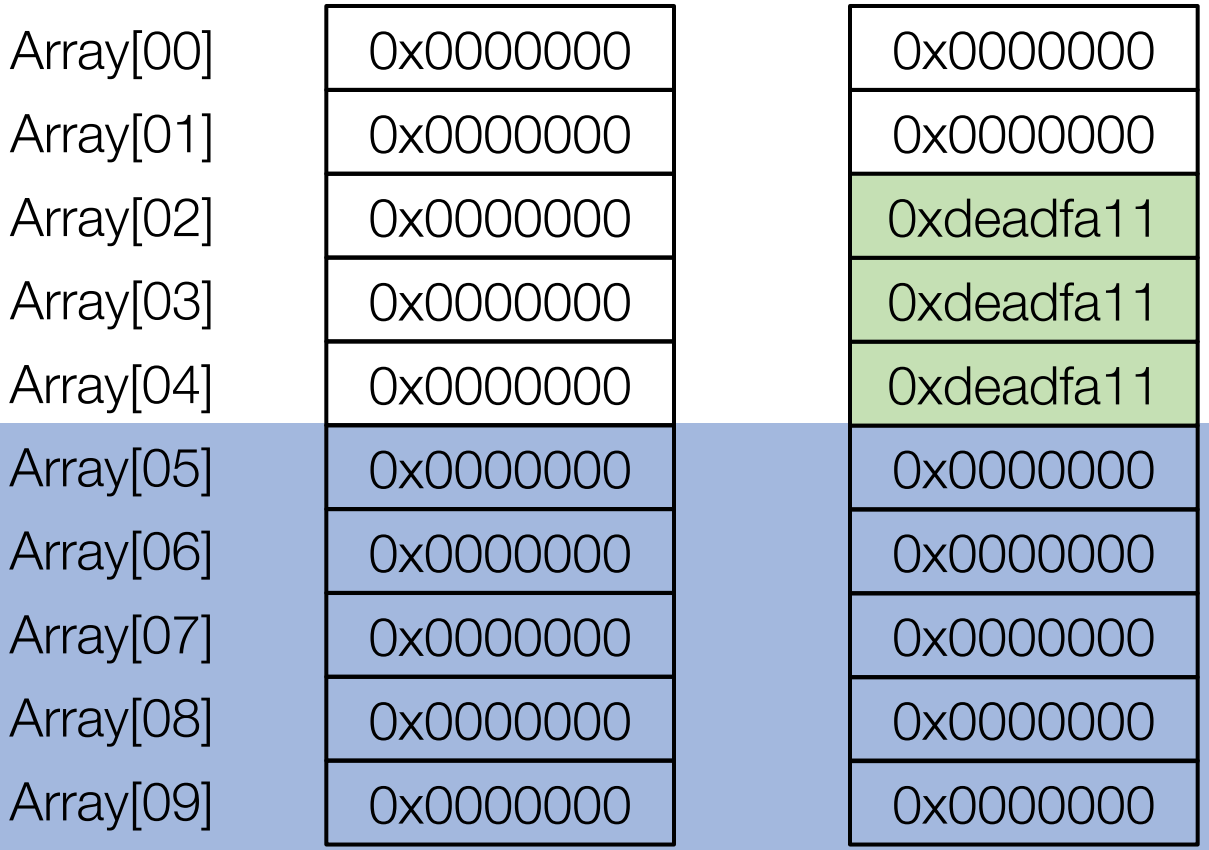
Let's pretend this is the stack



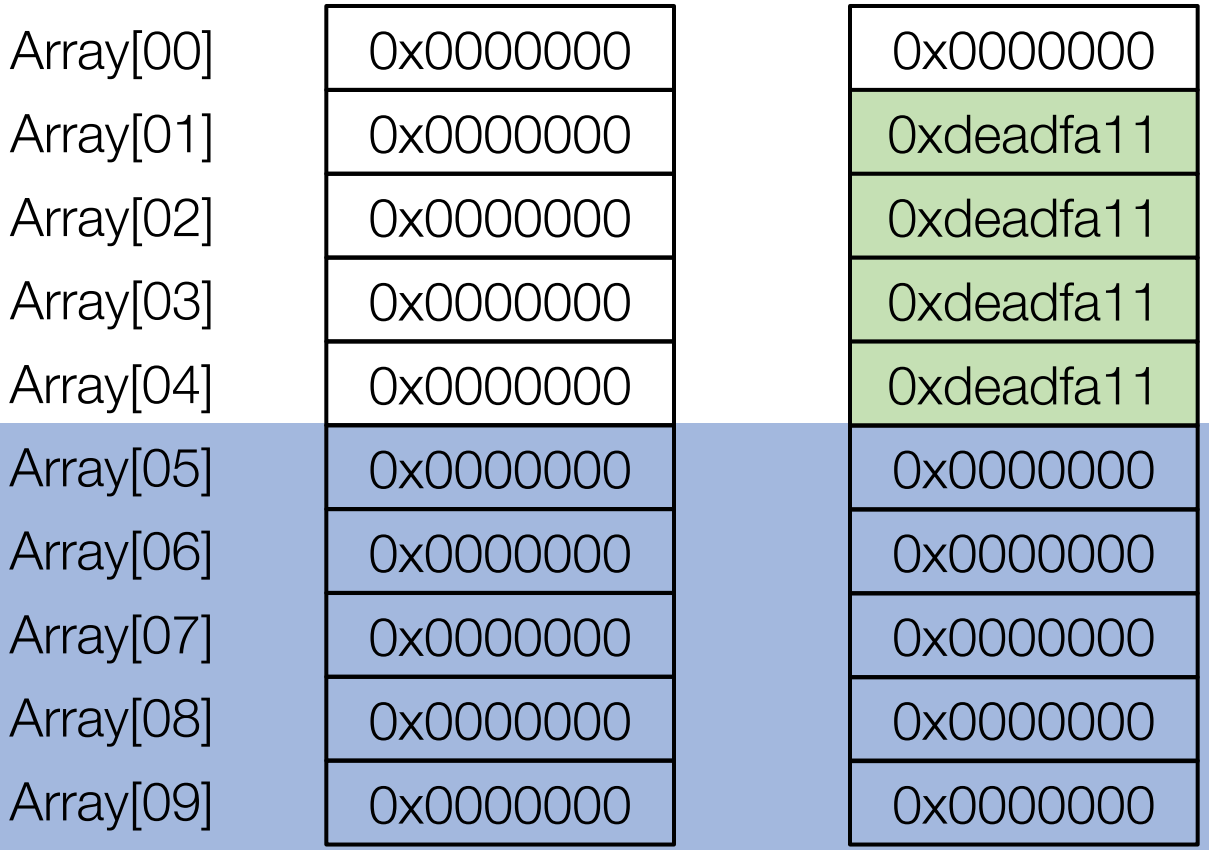
Let's pretend this is the stack



Let's pretend this is the stack

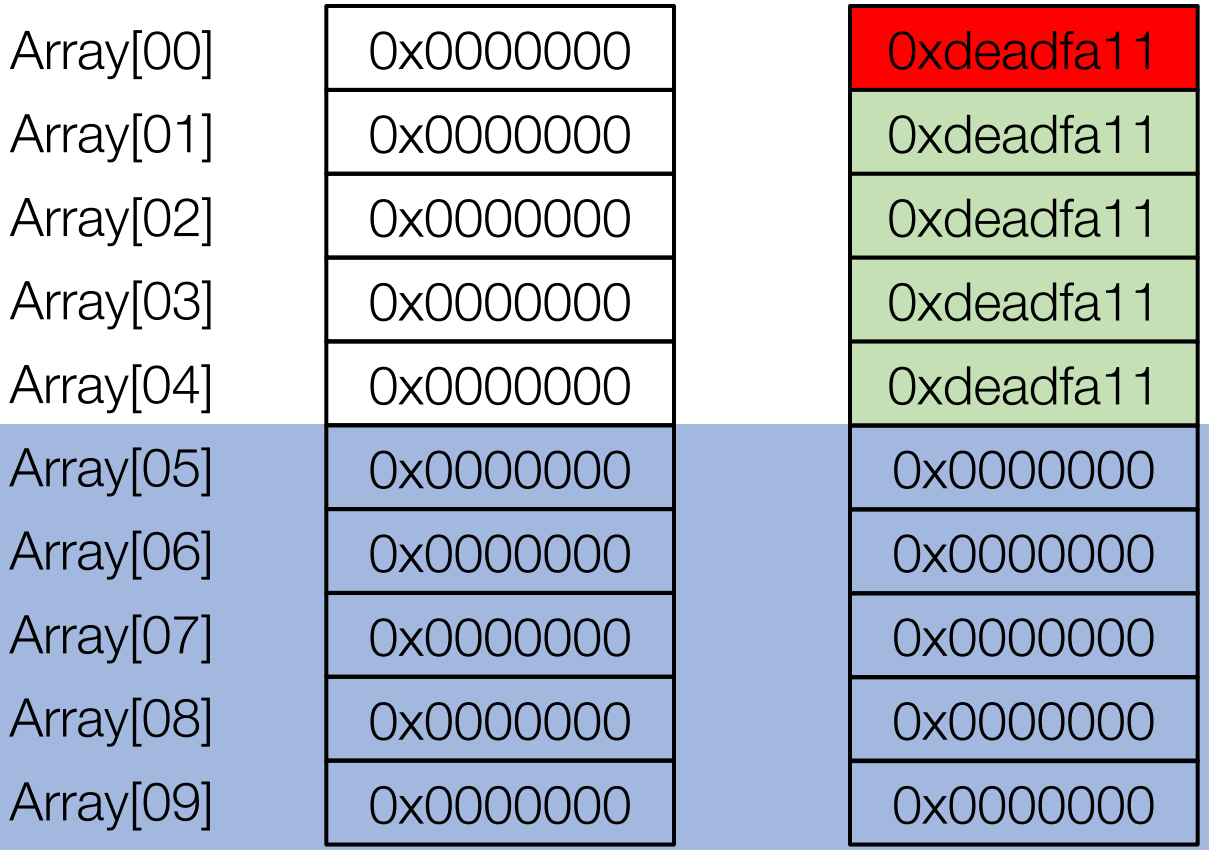


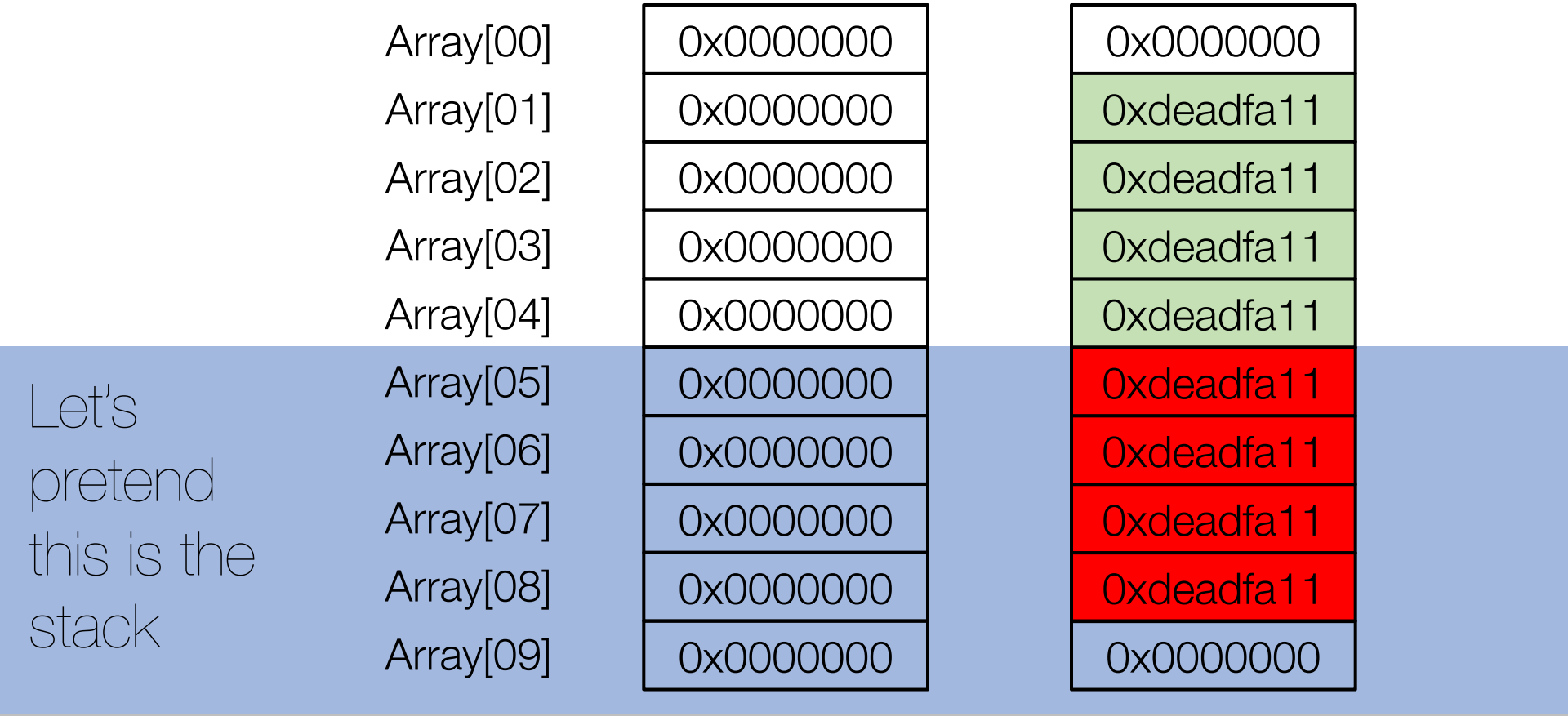
Let's pretend this is the stack



```
const int ARRAY_SIZE_MAX=10;
const int ARRAY_SIZE_TEST=5;
int copy_size=0;
uint32_t array[ARRAY_SIZE_MAX]={0};
do
{
        copy_size = ARRAY_SIZE_TEST + 3;

        // If an attacker tries to write beyond the array – prevent it
        if (copy_size >= ARRAY_SIZE_TEST)
           copy_size = ARRAY_SIZE_TEST - 1;

        // Overwrite elements  4...1
        while (copy_size >= 1)
        {
           array[copy_size] = 0xdeadfa11;
           copy_size--;
        }
} while the array is accurate
```

Let's pretend this is the stack




Array[00]
Array[01]
Array[02]
Array[03]
Array[04]
Array[05]
Array[06]
Array[07]
Array[08]
Array[09]
0x0000000
0x0000000
0x0000000
0x0000000
0x0000000
0x0000000
0x0000000
0x0000000
0x0000000
0x0000000
0x0000000
0xdeadfa11
0xdeadfa11
0xdeadfa11
0xdeadfa11
0xdeadfa11
0xdeadfa11
0xdeadfa11
0xdeadfa11
0x0000000
Let's pretend this is the stack

It's not just crypto!

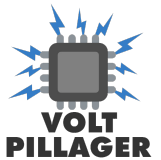
/bin/bash 158x41

```
versatile$ ./operation   -m 200 -s -177 -X 5  -i 200  -o P -c   "cat backup/text_file.txt" -r 0 -t 8

Summary
-------------------------------------------------
time (ms) interval:      200
Iterations:              200
Start Voltage:           -177
End Voltage:             0
Stop after x drops:      5
Voltage steps:           1
Threads:                 8
Operand1:                0x000000ffffffffff
Operand2:                0x000000ffffffffff
Operand1 is:             maximum
Operand2 is:             maximum
Operand1 min is:         0x0000000000000000
Operand2 min is:         0x0000000000000000
Calculation only:        No
Display calculation:     No
Verbose:                 Yes
Option:                  Command Line
Command Line options
 > Command line:         cat backup/text_file.txt
 > Result code:          0
```

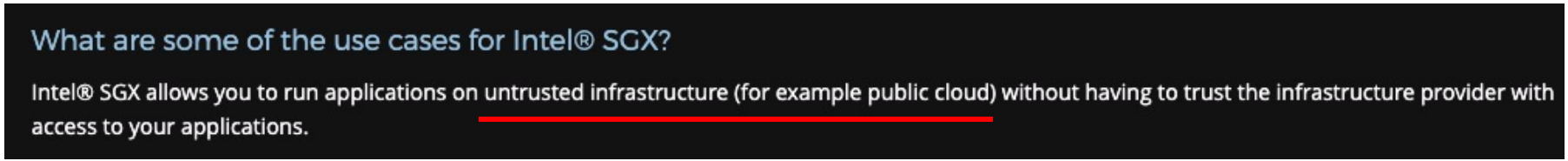
Intel’s response

"... opening the case and tampering of internal hardware to compromise SGX is out of scope for SGX threat model. Patches for CVE-2019-11157 (Plundervolt) were not designed to protect against hardware-based attacks as per the threat model" - Intel

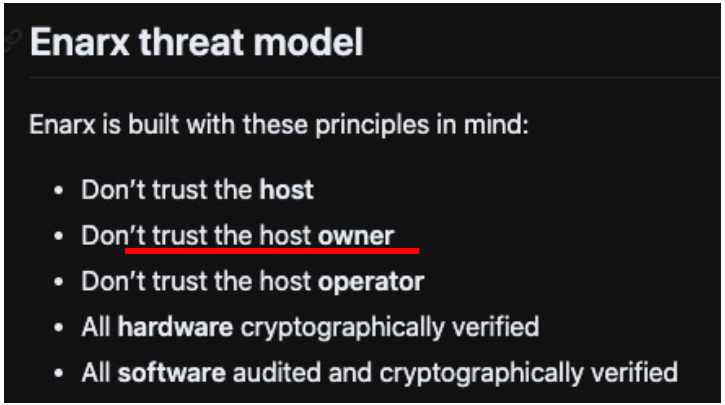
But… a lot of developers believe SGX can protect against hardware tempering.



a lot of developers believe SGX can protect against hardware tempering

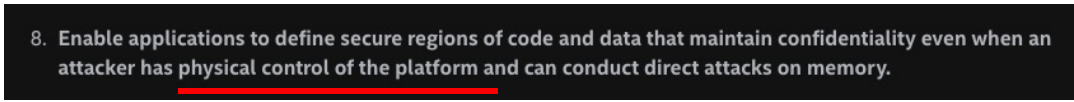


Source: Fortanix Intel SGX
https://web.archive.org/web/20201001235308/https://fortanix.com/intel-sgx/



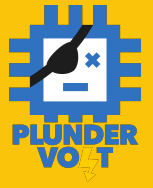
Source: Enarx Threat Model
https://github.com/enarx/enarx/wiki/Threat-Model

- Untrusted OS
- Untrusted owner
- Untrusted Infrastructure



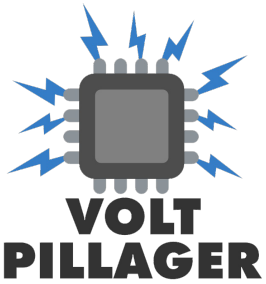
Source: Intel® SGX for Dummies (Intel® SGX Design Objectives)
https://software.intel.com/content/www/us/en/develop/blogs/protecting-application-secrets-with-intel-sgx.html

Concurrent work



Kenjar, Zijo et al "V0LTpwn: Attacking x86 Processor Integrity from Software"
In: USENIX Security Symposium 2020

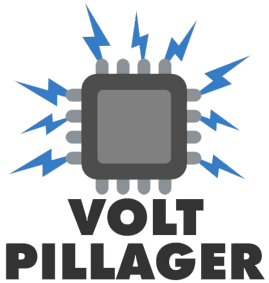
Qiu, P at al. "Breaking SGX by software-controlled voltage-induced hardware faults."
In AsianHOST 2019



- A new type of attack against Intel SGX
- Software based Attack -> Hardware based Attack
- Breaks the integrity of SGX
- Within SGX
    - Retrieve keys using AES-NI
    - Retrieve RSA key
    - Bypass comparison statements

Mitigation

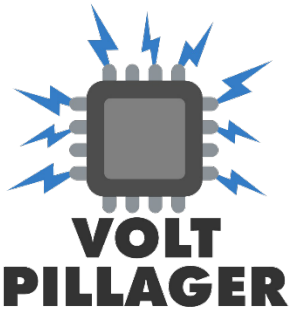
- I would say this is mitigated with the latest microcode/BIOS update



- Hard to mitigate
- May affect non-SGX programs
- Could affect other vendors
- Dedicated HW monitoring circuitry
- Redundant/fail to abort programming

Thank you.

https://plundervolt.com



http://hw.plundervolt.com

Or

https://zt-chen.github.io/voltpillager/