VOLT PILLAGER

Hardware Based CPU Undervolting on The Cheap

Stealing Your Secrets for $30

Zitai Chen, Georgios Vasilakis, Kit Murdock,
David Oswald, Flavio D. Garcia

University of Birmingham, UK

TEEs

Trusted Execution Environments

arm
TRUSTZONE

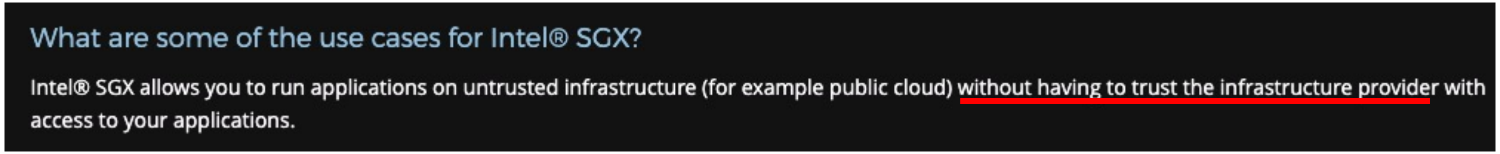
intel
SGX

AMD

Root of Trust
AMD Secure Processor

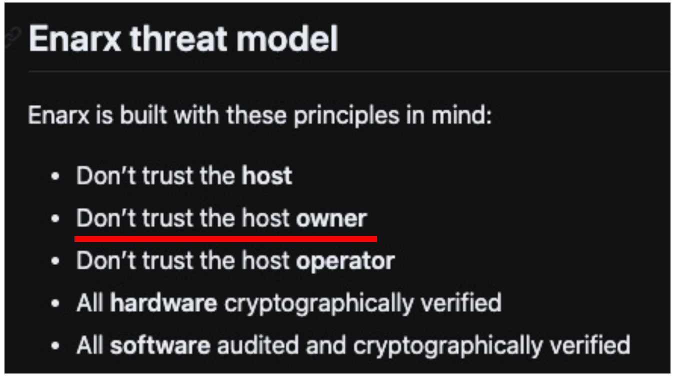
IBM Secure Service Container

Threat Model


What are some of the use cases for Intel® SGX?

Intel® SGX allows you to run applications on untrusted infrastructure (for example public cloud) without having to trust the infrastructure provider with access to your applications.

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

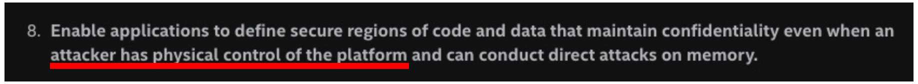

Enarx threat model

Enarx is built with these principles in mind:

- Don't trust the **host**
- Don't trust the host **owner**
- Don't trust the host **operator**
- All **hardware** cryptographically verified
- All **software** audited and cryptographically verified

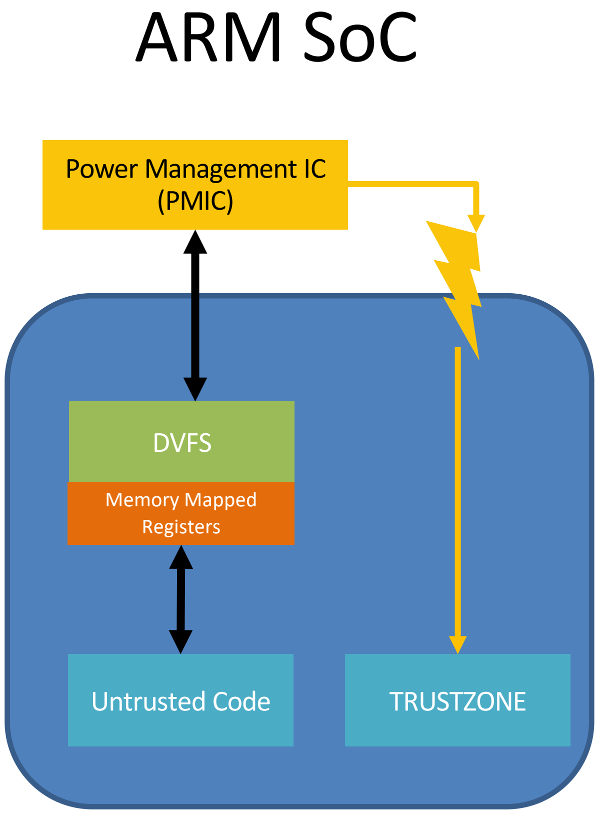
Source: Enarx Threat Model
https://github.com/enarx/enarx/wiki/Threat-Model

- Untrusted OS
- Untrusted owner
- Untrusted Infrastructure


8. Enable applications to define secure regions of code and data that maintain confidentiality even when an attacker has physical control of the platform and can conduct direct attacks on memory.

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

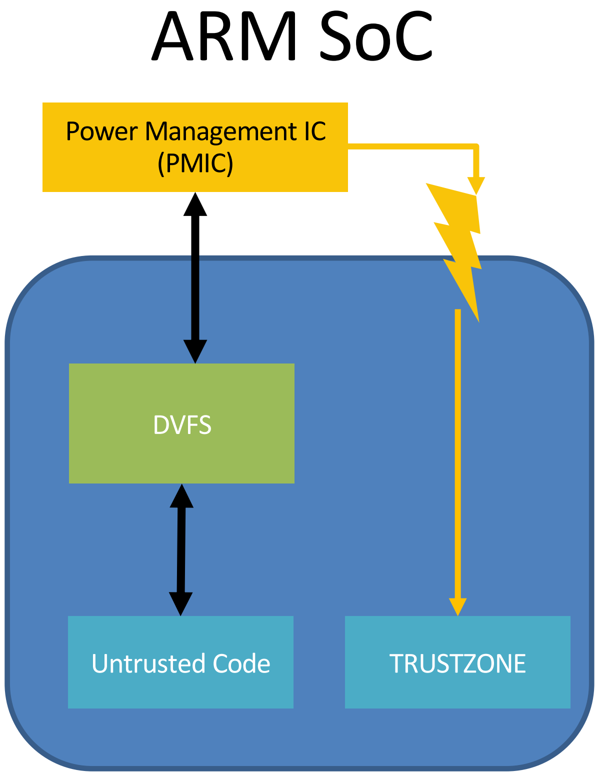
Fault injection - ARM

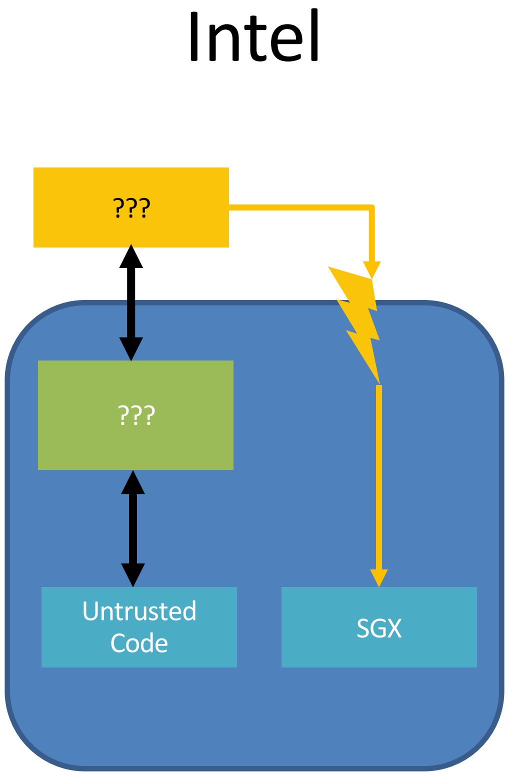


Adrian Tang et al. "CLKSCREW: exposing the perils of security-oblivious energy management"
In: USENIX Security Symposium. 2017

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

What about Intel?

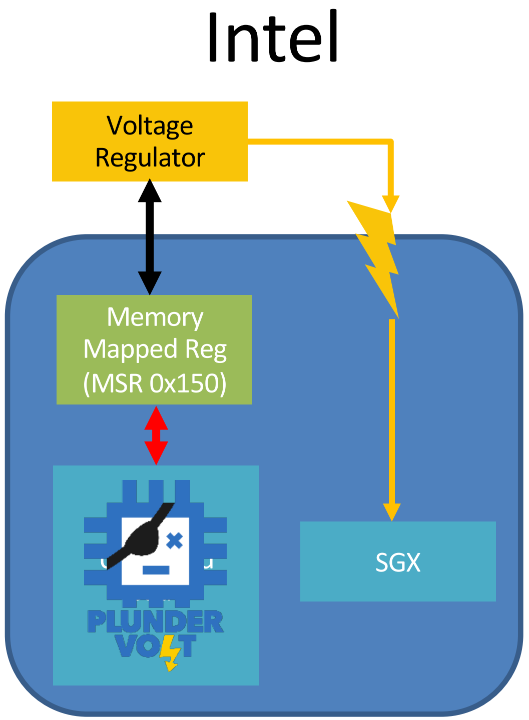




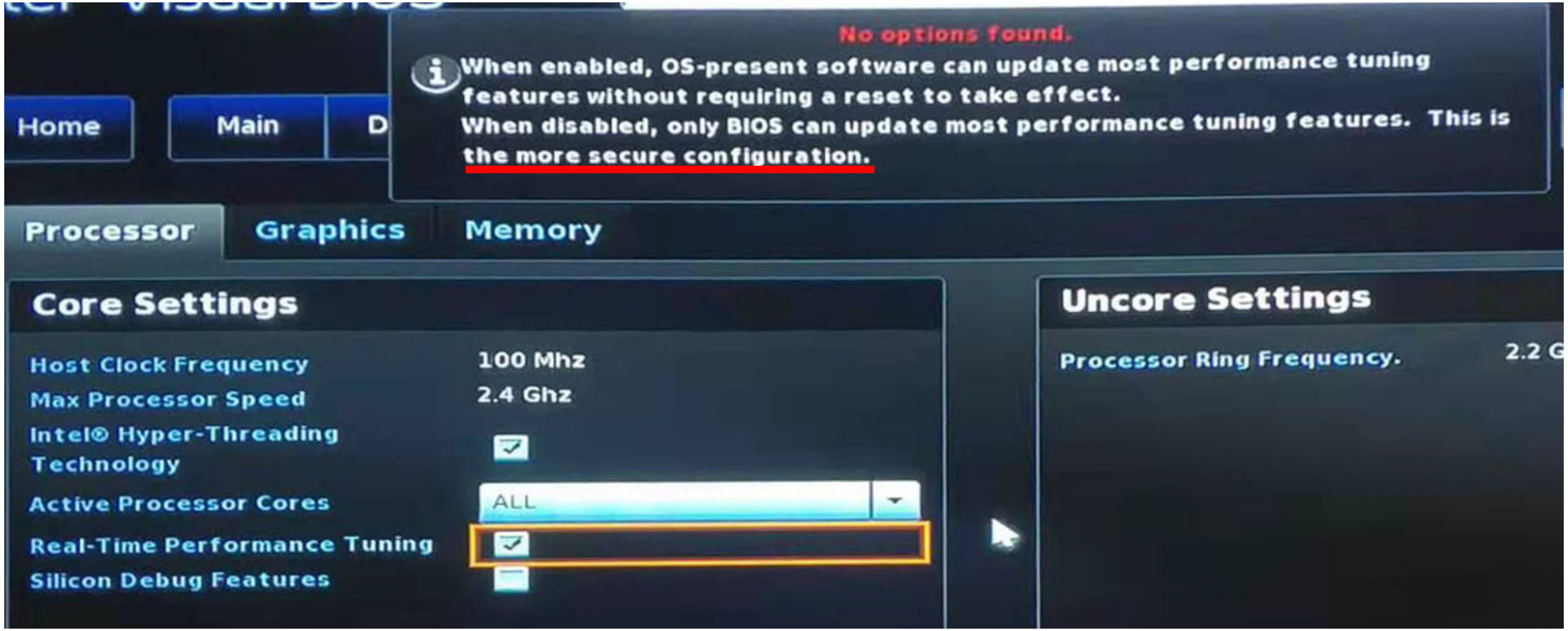
Plundervolt

- 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)



Undervolting via MSR 0x150 disabled



No options found.

When enabled, OS-present software can update most performance tuning features without requiring a reset to take effect.
When disabled, only BIOS can update most performance tuning features. This is the more secure configuration.

Core Settings

Host Clock Frequency	100 Mhz
Max Processor Speed	2.4 Ghz
Intel® Hyper-Threading Technology	✓
Active Processor Cores	ALL
Real-Time Performance Tuning	✓
Silicon Debug Features	

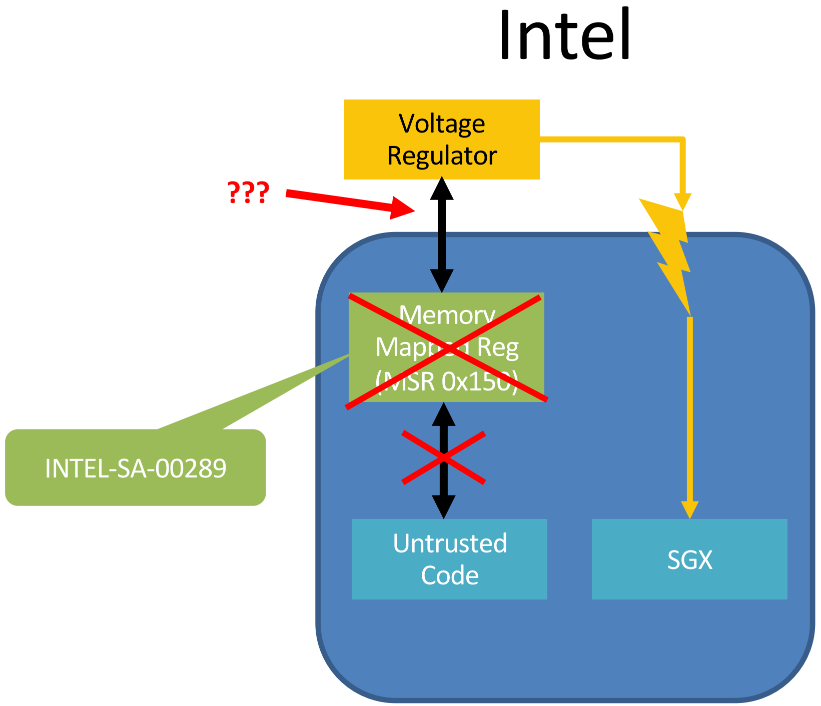
Uncore Settings

Processor Ring Frequency. 2.2 G

Undervolting via MSR 0x150 disabled

Recommendations:

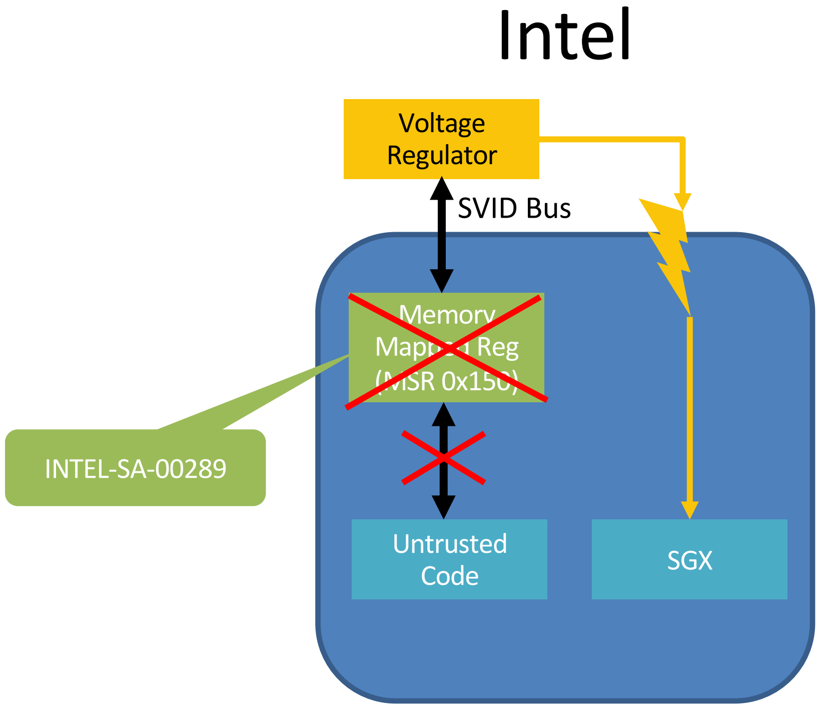
Intel recommends that users of the above Intel® Processors **update to the latest BIOS version** provided by the system manufacturer that addresses these issues.



SVID Bus

SVID Bus

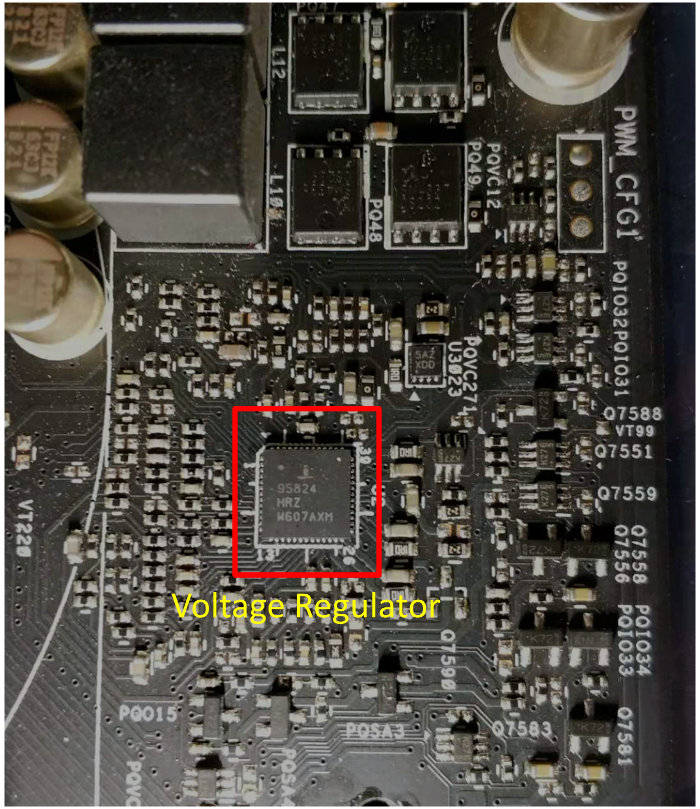
- 3 Wire interface
  - CLK, DATA and ALERT(Not required)
- Clock @ 25MHz
- Logical High >0.64V, Low <0.45V

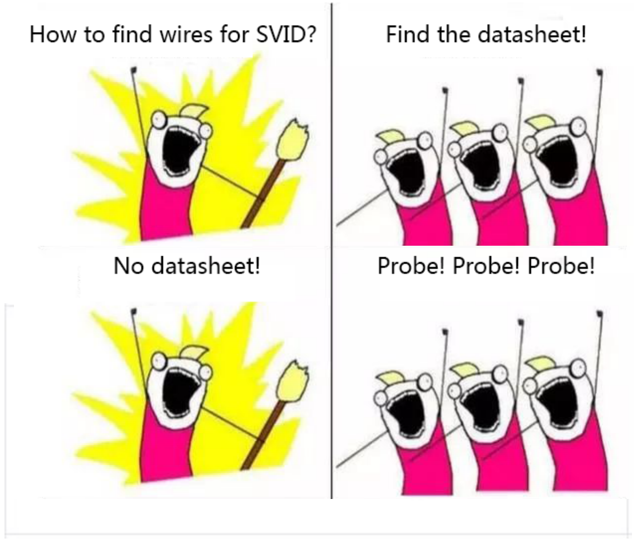


Ref:
1. [L6751C Digitally controlled dual PWM for Intel VR12 and AMD SVI]
2. 8th Generation Intel® CoreTM Processor Families Datasheet, Volume 1 of 2

VOLT PILLAGER

SVID Bus – Which wire?

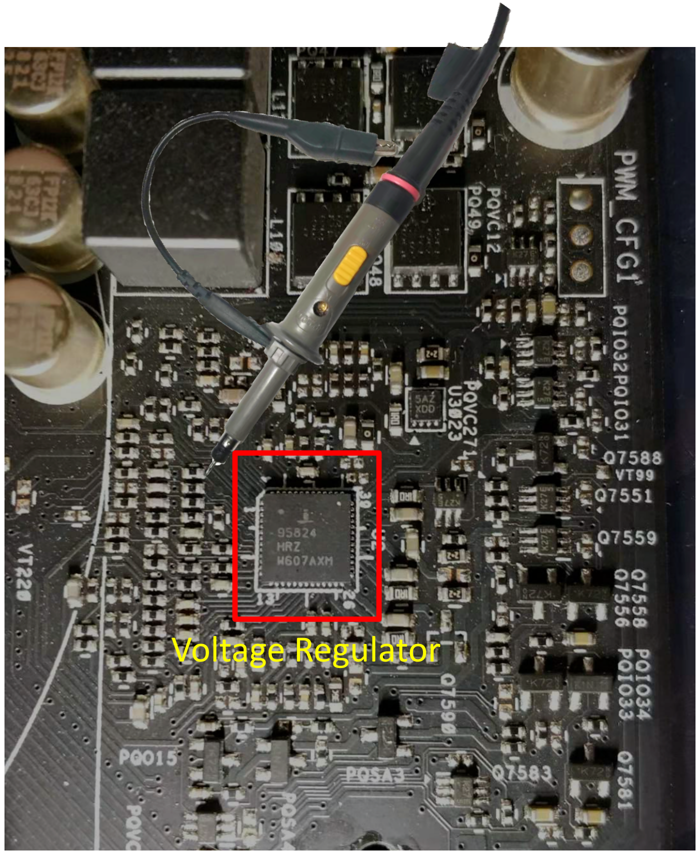




- 1*A4 page long
- Does not show pin definition
- No information about the signal

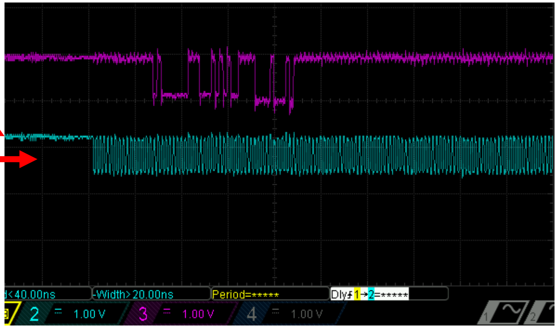
VOLT PILLAGER

SVID Bus – Which wire?

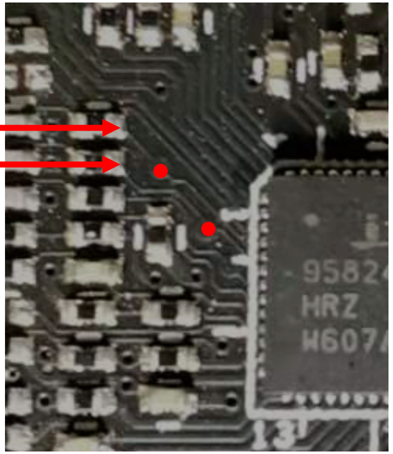


1V

25MHz



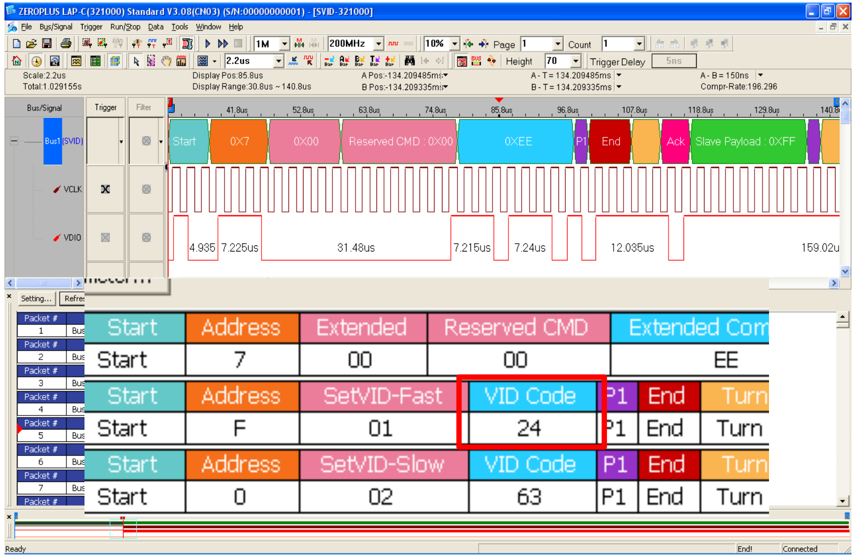
SVID Bus



VOLT PILLAGER

SVID Protocol

Commands & Packet Strucutres



ZEROPLUS LAP-C(321000) Standard V3.08(CN03) (S/N:00000000001) - [SVID-321000]

Scale:2.2us Total:1.029155s Display Pos:85.8us Display Range:30.8us ~ 140.8us A Pos:-134.209485ms B Pos:-134.209335ms A - T = 134.209485ms B - T = 134.209335ms A - B = 150ns Compr-Rate:196.296

Start	Address	Extended	Reserved CMD	Extended Com
Start	7	00	00	EE

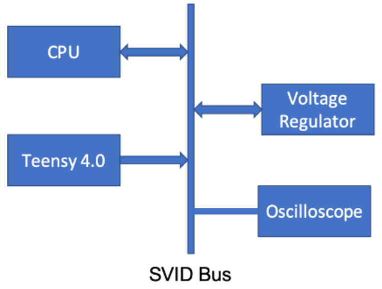
Start	Address	SetVID-Fast	VID Code	P1	End	Turn
Start	F	01	24	P1	End	Turn

Start	Address	SetVID-Slow	VID Code	P1	End	Turn
Start	0	02	63	P1	End	Turn

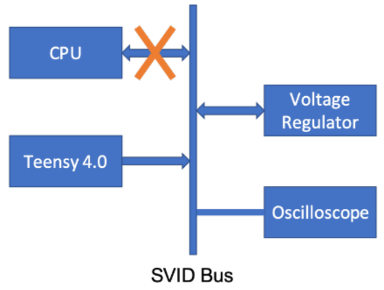
Src: ZEROPLUS Protocol Analyzer SVID_V1.04.0 [Link]

RE Voltage Identifiers

Observe

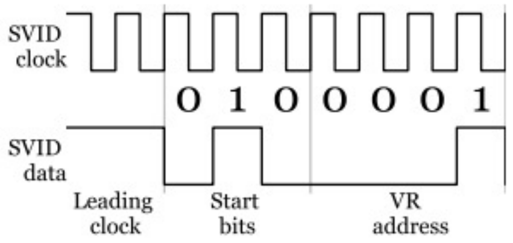


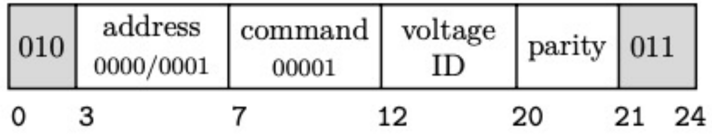
Verify

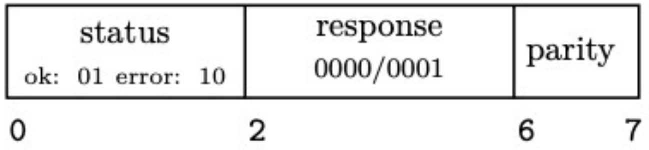


SVID Protocol

SVID signals and data frame







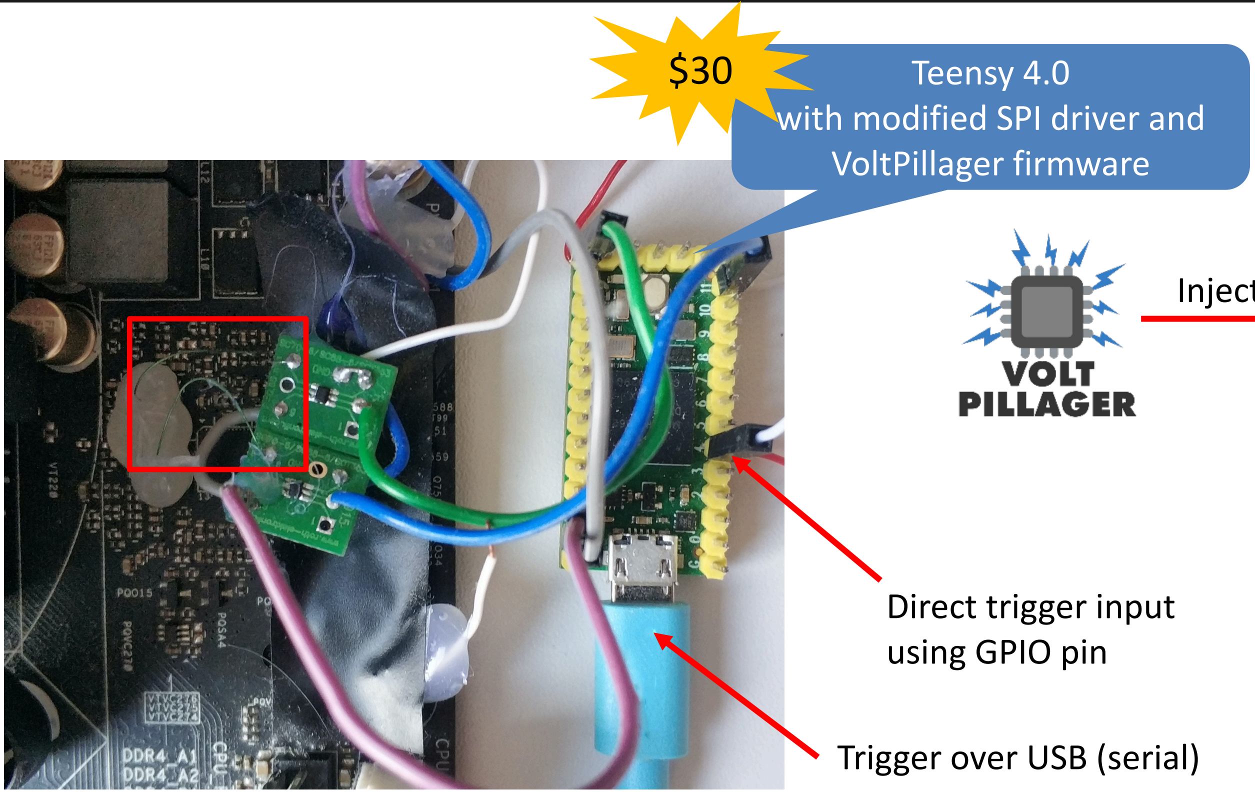
VID： 1byte, computed as (voltage U in volt):

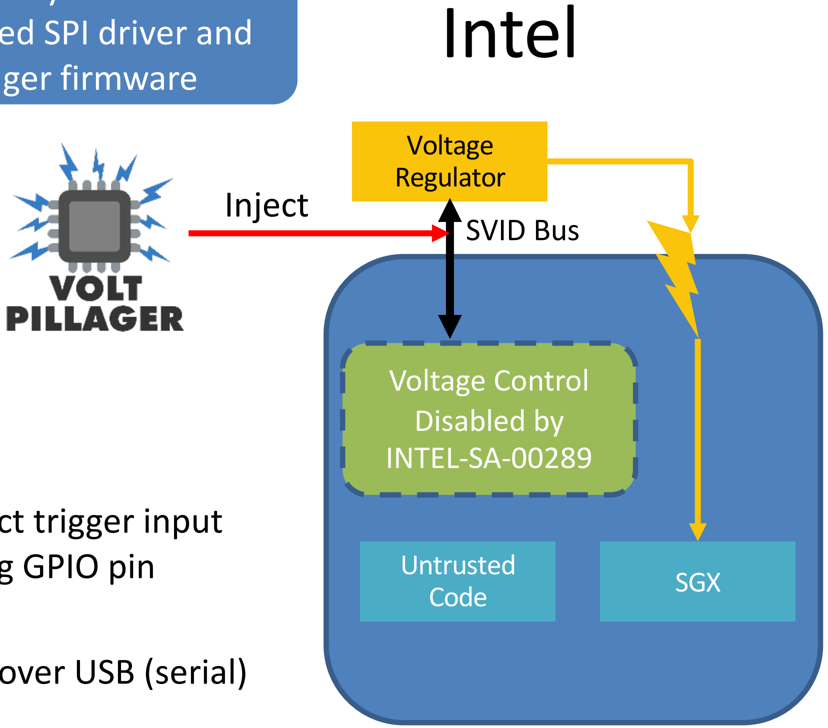
$$\mathrm{VID} = \left\lfloor \frac{U - 0.245}{0.005} \right\rfloor$$

VID Commands: 5bits

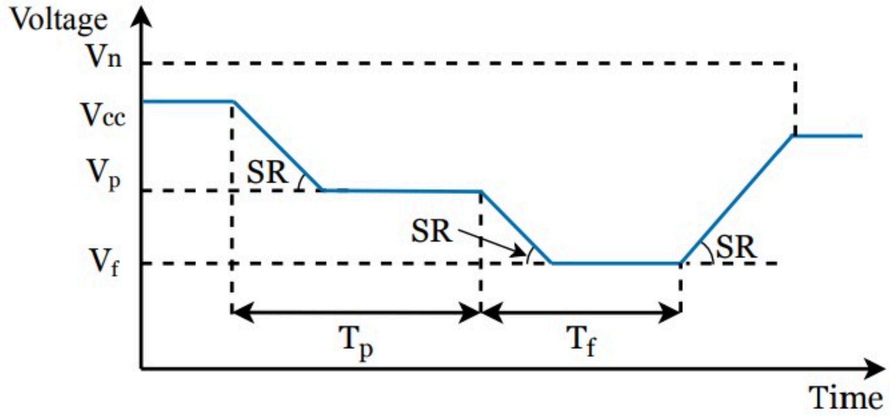
Command name	Value
Extended	0x00
SetVID-Fast	0x01
SetVID-Slow	0x02
SetVID-Decay	0x03
SetPS	0x04
SetRegADR	0x05
SetRegDAT	0x06

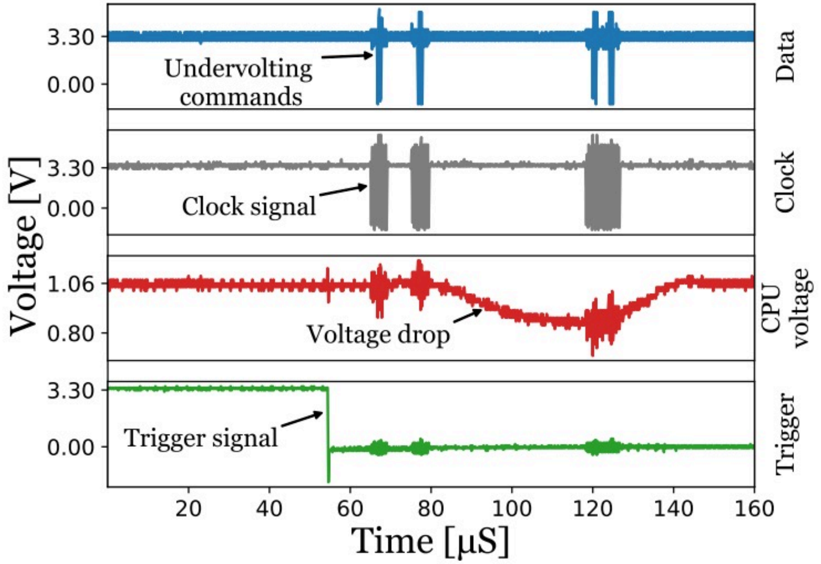
VoltPillager: Hardware undervolting


$30
Teensy 4.0 with modified SPI driver and VoltPillager firmware
Direct trigger input using GPIO pin
Trigger over USB (serial)



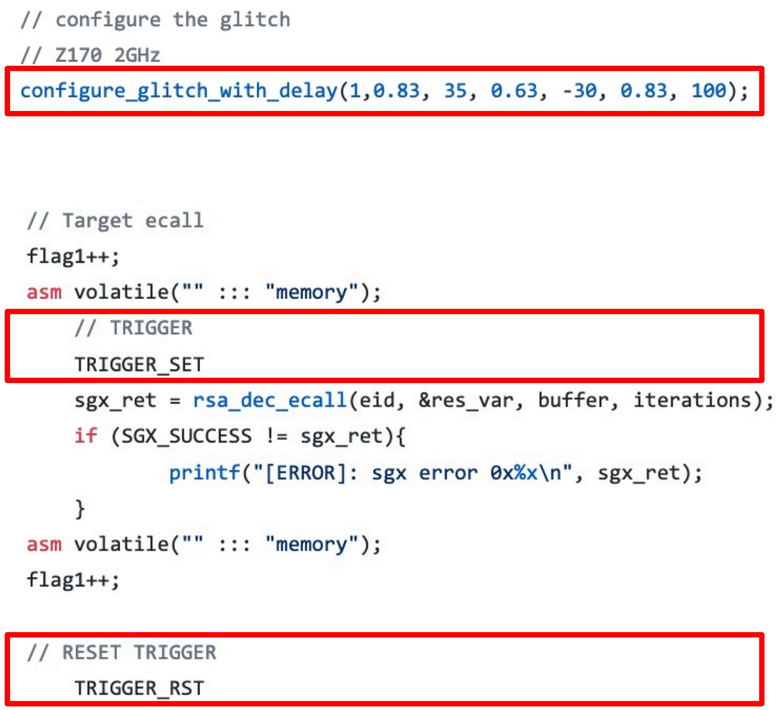
VoltPillager: Glitch Parameters





Let’s Inject Some Fault

Library for undervolting

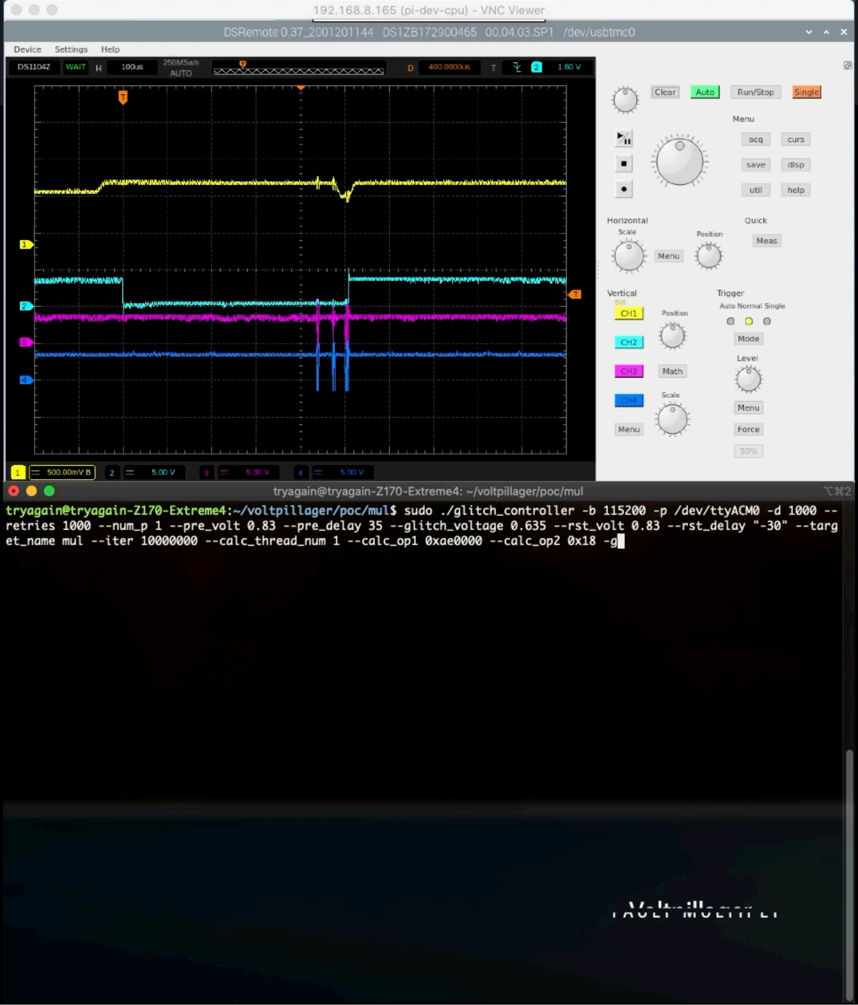


```
// configure the glitch
// Z170 2GHz
configure_glitch_with_delay(1,0.83, 35, 0.63, -30, 0.83, 100);


// Target ecall
flag1++;
asm volatile("" ::: "memory");
    // TRIGGER
    TRIGGER_SET
    sgx_ret = rsa_dec_ecall(eid, &res_var, buffer, iterations);
    if (SGX_SUCCESS != sgx_ret){
            printf("[ERROR]: sgx error 0x%x\n", sgx_ret);
    }
asm volatile("" ::: "memory");
flag1++;

// RESET TRIGGER
    TRIGGER_RST
```

Fault Injection with VoltPillager

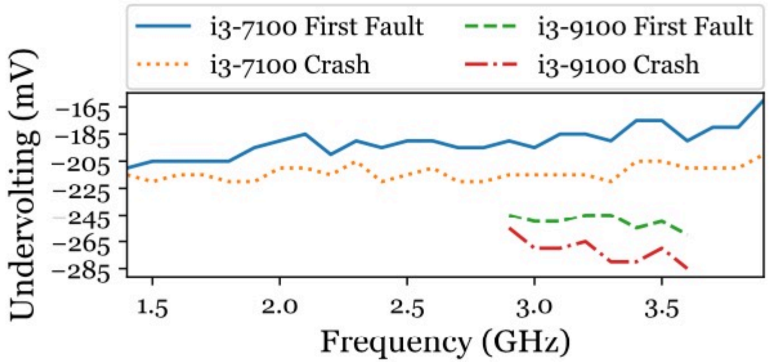
- Multiplication Fault
- RSA Fault (in SGX)
- AES-NI Fault (in SGX)
    - **mbedtls_aesni**
    - **Open Enclave file-encryptor**
- Delayed-Write Fault

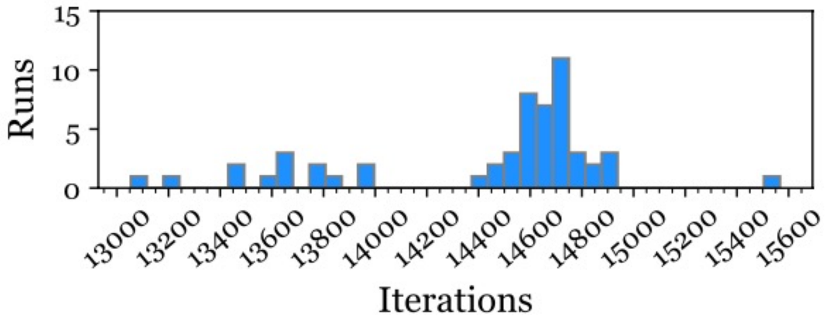
Multiplication



```
tryagain@tryagain-Z170-Extreme4:~/voltpillager/poc/mul$ sudo ./glitch_controller -b 115200 -p /dev/ttyACM0 -d 1000 --
retries 1000 --num_p 1 --pre_volt 0.83 --pre_delay 35 --glitch_voltage 0.635 --rst_volt 0.83 --rst_delay "-30" --targ
et_name mul --iter 10000000 --calc_thread_num 1 --calc_op1 0xae0000 --calc_op2 0x18 -g
```

VoltPillager V.S. Plundervolt





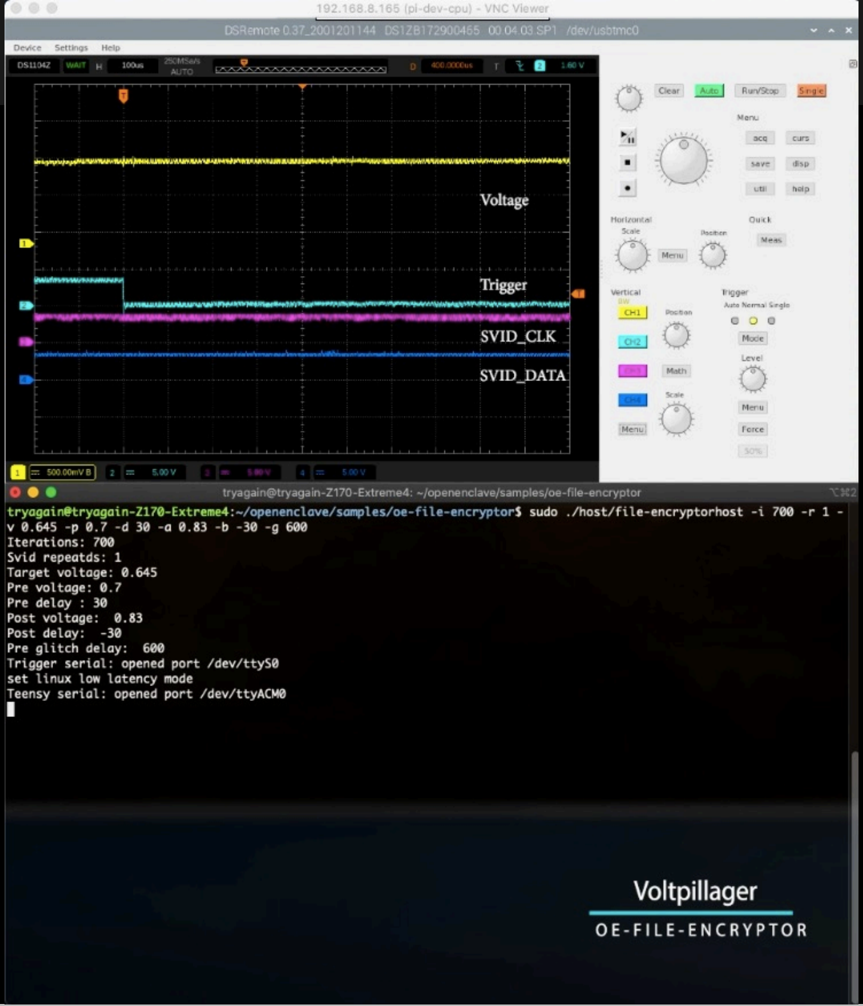
4VID Steps

14,634 ± 300 -> 75% of faults

Fault Encryptions

Fault Encryptions

- sgx_crt_rsa PoC of Plundervolt
    - Recover the private the key
- sgx_aes_ni
- Open Enclave file-encryptor sample (AES-CBC)

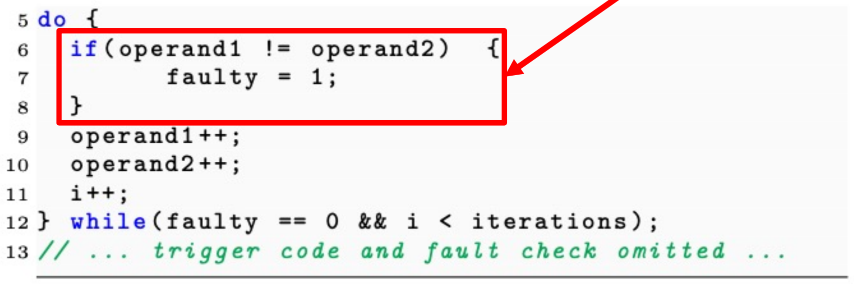

192.168.8.165 (pi-dev-cpu) - VNC Viewer
DSRemote 0.37_2001201144 DS1ZB172900465 00.04.03.SP1 /dev/usbtmc0
Voltage
Trigger
SVID_CLK
SVID_DATA
tryagain@tryagain-Z170-Extreme4: ~/openenclave/samples/oe-file-encryptor
Voltpillager
OE-FILE-ENCRYPTOR

```
tryagain@tryagain-Z170-Extreme4:~/openenclave/samples/oe-file-encryptor$ sudo ./host/file-encryptorhost -i 700 -r 1 -
v 0.645 -p 0.7 -d 30 -a 0.83 -b -30 -g 600
Iterations: 700
Svid repeatds: 1
Target voltage: 0.645
Pre voltage: 0.7
Pre delay : 30
Post voltage:  0.83
Post delay:  -30
Pre glitch delay:  600
Trigger serial: opened port /dev/ttyS0
set linux low latency mode
Teensy serial: opened port /dev/ttyACM0
```

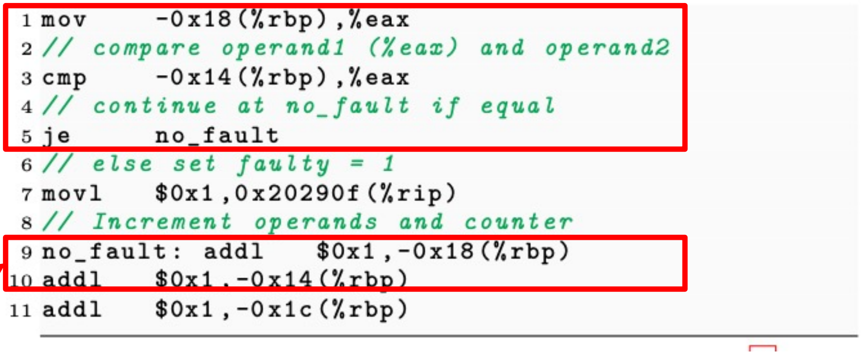
Delayed-write fault

Delayed-Write Fault – Initial PoC

Should never happen



```
do {
  if(operand1 != operand2)  {
        faulty = 1;
  }
  operand1++;
  operand2++;
  i++;
} while(faulty == 0 && i < iterations);
// ... trigger code and fault check omitted ...
```



```
mov      -0x18(%rbp),%eax
// compare operand1 (%eax) and operand2
cmp      -0x14(%rbp),%eax
// continue at no_fault if equal
je       no_fault
// else set faulty = 1
movl     $0x1,0x20290f(%rip)
// Increment operands and counter
no_fault: addl    $0x1,-0x18(%rbp)
addl     $0x1,-0x14(%rbp)
addl     $0x1,-0x1c(%rbp)
```

Not committed when CMP happen

Observed using VoltPillager

Delayed-Write Fault – Practical Exploitation

```c
uint32_t array[8] = { 0 };
// Attacker-supplied out-of-bounds size
int copy_size = 7;

// Ensure we stay within bounds
if(copy_size >= 5)
  copy_size = 4;

// overwrite elements 4, 3, 2, 1
while(copy_size >= 1)  {
  array[copy_size] = 0xabababab;
  copy_size--;
}
```

Normal execution:

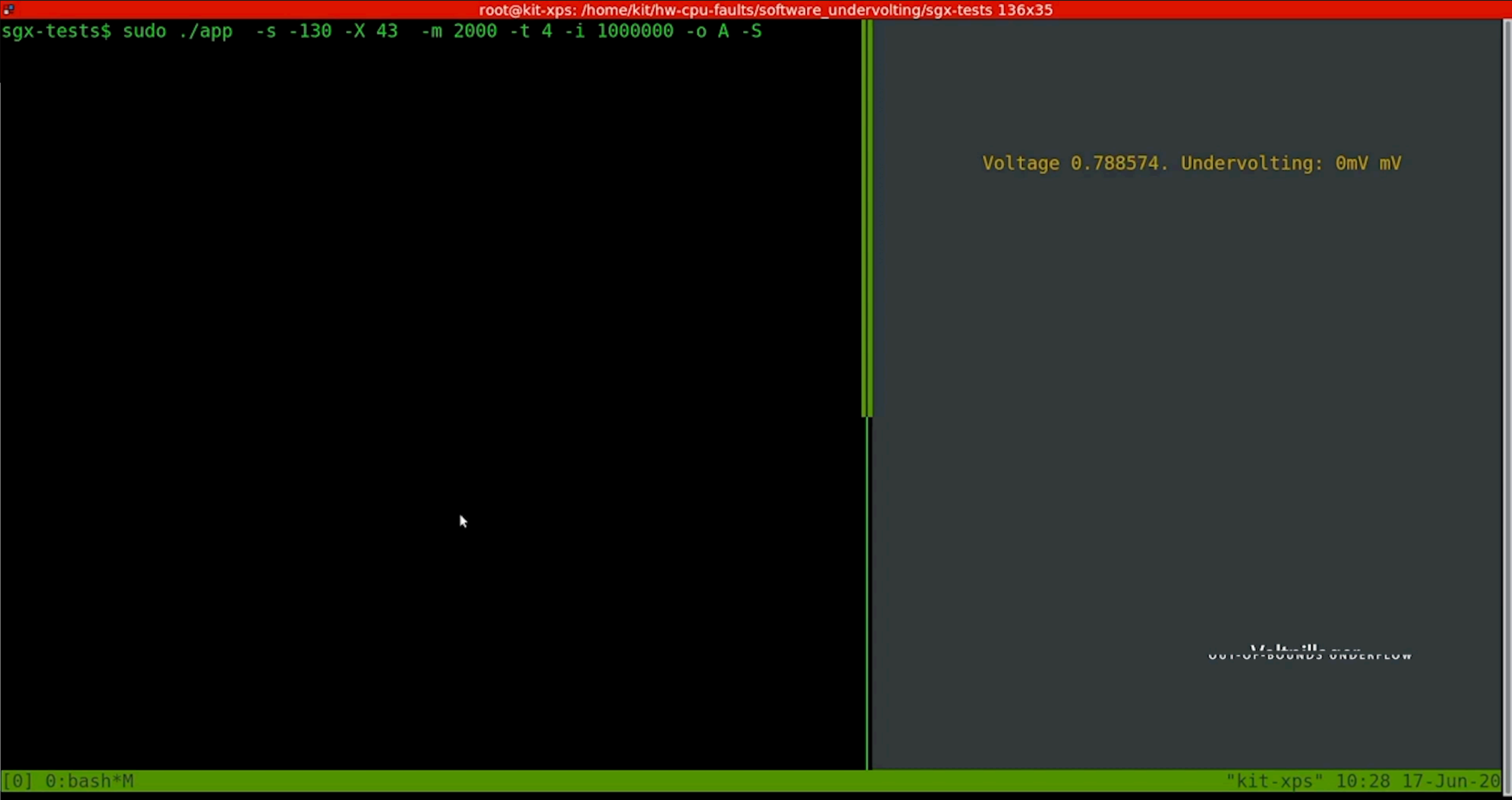
00...	AB...	AB...	AB...	AB...	00...	00...	00...

Fault 1 causing out-of-bounds underflow:

AB...	AB...	AB...	AB...	AB...	00...	00...	00...

Fault 2 causing out-of-bounds overflow:

00...	AB...	AB...	AB...	AB...	**AB...**	**AB...**	**AB...**



```
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 0.788574. Undervolting: 0mV mV
```

```
[illegible]
OUT-OF-BOUNDS UNDERFLOW
```

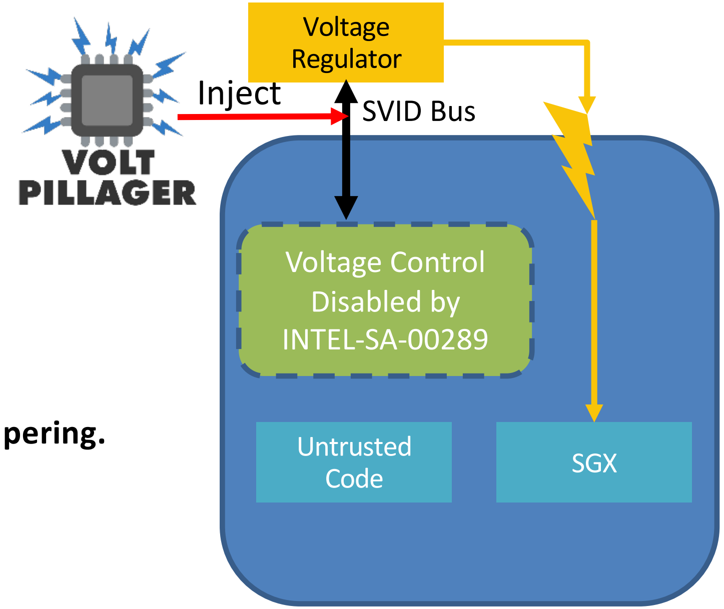
```
[0] 0:bash*M                                                    "kit-xps" 10:28 17-Jun-20
```

Intel’s response

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 still think SGX can protect against hardware tempering.



Summary

VOLT PILLAGER

- 1st hardware based undervolting against Intel CPUs
- Physical access -> CVE- 2019-11157(Plundervolt)
- Build for $30
- Rethink of Intel SGX Threat Model

Thank you.

VOLT
PILLAGER

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