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
Trusted Execution Environments

arm TRUSTZONE

AMD

Root of Trust
AMD Secure Processor

Intel SGX

IBM Secure Service Container

Hardware Based CPU Undervolting on The Cheap --Stealing Your Secrets for $30
**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

---

**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

Source: Intel® SGX for Dummies (Intel® SGX Design Objectives)
ARM SoC

Power Management IC (PMIC)

DVFS

Memory Mapped Registers

Untrusted Code

TRUSTZONE

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

Power Management IC (PMIC)

DVFS

Untrusted Code

TRUSTZONE

Intel

???

Untrusted Code

SGX
Plundervolt

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

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

Intel

Voltage Regulator

Memory Mapped Reg (MSR 0x150)

SGX
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.
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

- 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® Core™ Processor Families Datasheet, Volume 1 of 2
SVID Bus – Which wire?

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

How to find wires for SVID? Find the datasheet!

No datasheet! Probe! Probe! Probe!
SVID Bus – Which wire?

SVID Bus

1V
25MHz

Voltage Regulator

Hardware Based CPU Undervolting on The Cheap --Stealing Your Secrets for $30
SVID Protocol

Commands & Packet Structures

Observe

RE Voltage Identifiers

Verify

Src: ZEROPLUS Protocol Analyzer SVID_V1.04.0 [Link]
SVID Protocol

SVID signals and data frame

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

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

VID Commands: 5 bits

<table>
<thead>
<tr>
<th>Command name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Extended</td>
<td>0x00</td>
</tr>
<tr>
<td>SetVID-Fast</td>
<td>0x01</td>
</tr>
<tr>
<td>SetVID-Slow</td>
<td>0x02</td>
</tr>
<tr>
<td>SetVID-Decay</td>
<td>0x03</td>
</tr>
<tr>
<td>SetPS</td>
<td>0x04</td>
</tr>
<tr>
<td>SetRegADR</td>
<td>0x05</td>
</tr>
<tr>
<td>SetRegDAT</td>
<td>0x06</td>
</tr>
</tbody>
</table>
VoltPillager: Hardware undervolting

$30

Teensy 4.0
with modified SPI driver and
VoltPillager firmware

Intel

Voltage Regulator

Inject

SVID Bus

Voltage Control Disabled by INTEL-SA-00289

Untrusted Code

SGX

Direct trigger input using GPIO pin

Trigger over USB (serial)
VoltPillager: Glitch Parameters

![Diagram of voltage and time parameters with labeled variables: V_n, V_cc, V_p, V_f, T_p, T_f.]

- Undervolting commands
- Clock signal
- Voltage drop
- Trigger signal

Hardware Based CPU Undervolting on The Cheap -- Stealing Your Secrets for $30
Let’s Inject Some Fault
```c
// 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```

Library for undervolting
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
Hardware Based CPU Undervolting on The Cheap -- Stealing Your Secrets for $30
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 key
- `sgx_aes_ni`
- Open Enclave file-encryptor sample (AES-CBC)
Hardware Based CPU Undervolting on The Cheap --Stealing Your Secrets for $30
Delayed-write fault
Delayed-Write Fault – Initial PoC

```
5  do {
6     if (operand1 != operand2) {
7         faulty = 1;
8     }
9     operand1++;
10    operand2++;
11    i++;
12  } while (faulty == 0 && i < iterations);
13  // ... trigger code and fault check omitted ...
```

Should never happen

```
1 mov -0x18(%rbp),%eax
2 // compare operand1 (%eax) and operand2
3 cmp -0x14(%rbp),%eax
4 // continue at no_fault if equal
5 je no_fault
6 // else set faulty = 1
7 movl $0x1,0x20290f(%rip)
8 // Increment operands and counter
9 no_fault: addl $0x1,-0x18(%rbp)
10 addl $0x1,-0x14(%rbp)
11 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...
```
Hardware Based CPU Undervolting on The Cheap -- Stealing Your Secrets for $30

gsx-tests$ sudo ./app -s -130 -X 43 -m 2000 -t 4 -i 100000 -o A -S

Voltage 0.788574. Undervolting: 0mV mV

[0] 0:bash*M
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

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

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