A Bad Dream: Subverting Trusted Platform Module While You Are Sleeping

SEUNGHUN HAN, WOOK SHIN, JUN-HYEOK PARK, HYOUNGCHUN KIM
NATIONAL SECURITY RESEARCH INSTITUTE

Presented by: Oskars Daukst s
November 6, 2018

Overview

Overview

- Background
- Assumptions and Threat Model
- Vulnerabilities and Exploits
- Evaluation
- Countermeasures
- Conclusion

Background

Background - Trusted Computing Group

- Specifies and standardizes the TPM technology
  - TPM (Core technology) – Trusted Platform Module
  - Security related functions
  - APIs
  - Protocols
- TCG technology has been applied to UEFI
  - UEFI – Unified Extensible Firmware Interface
    - Similar to BIOS
    - Low-level software that starts on PC boot

Background - TPM

- Designed for hardware security
- Tamper resistant
- Contains
  - Processor, RAM, ROM, non-volatile RAM
  - PCRs – platform configuration registers
- Encryption/Decryption
- TMP
  - 1.2 – 2003
  - 2.0 – 2013

Background – TMP cont.

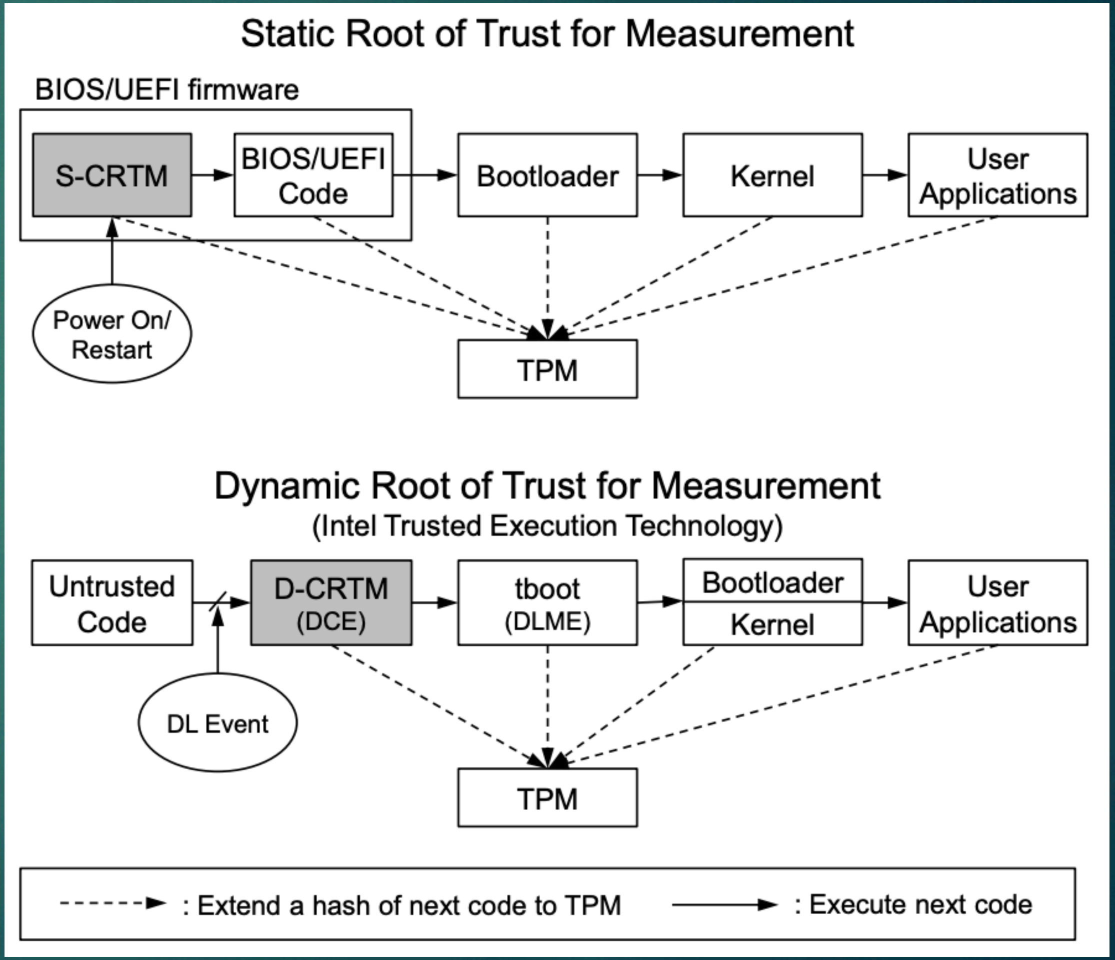
- Determines system trustworthiness
- Limits access to secret data
- Major component of integrity chain
- Operations
  - Seal – encrypts data to PCRs
  - Unseal – decrypts data based on current PCR values

Background – Integrity Chain

- Collection of components
  - Bootloader, kernel and other
- Starts statically or dynamically at RTM
  - RTM – Root of Trust
- Each components gets measured by TMP and hashed one at a time to a PCR
  - Each PCR’s hash gets updated with old + current measurement
  - *newPCR = HASH(oldPCR | | newMeasurement)*
  - *PCRs contain measurement results of a system*

Background – SRTM & DRTM

- SRTM
  - Initialized by S-CRTM
  - POWER-ON, RESTART
- DRTM
  - Started by D-CRTM
  - Runtime (no reset required)
  - Utilizes tboot (trusted boot)



Static Root of Trust for Measurement

Dynamic Root of Trust for Measurement (Intel Trusted Execution Technology)

------▶ : Extend a hash of next code to TPM ——▶ : Execute next code

Background – ACPI

- Advanced Configuration and Power Interface
  - Manages and coordinates power management between devices such as CPUs, networks, storage, graphics, etc.
- Sleeping States
  - S1 – Power on suspend
    - CPU stops executing instructions, but devices remain on
  - S2 – S1, but CPU powered off
  - S3 – Sleep, all devices are off, except RAM
  - S4 – Hibernation (suspend to disk), platform context in main memory saved to disk. All devices are powered off.

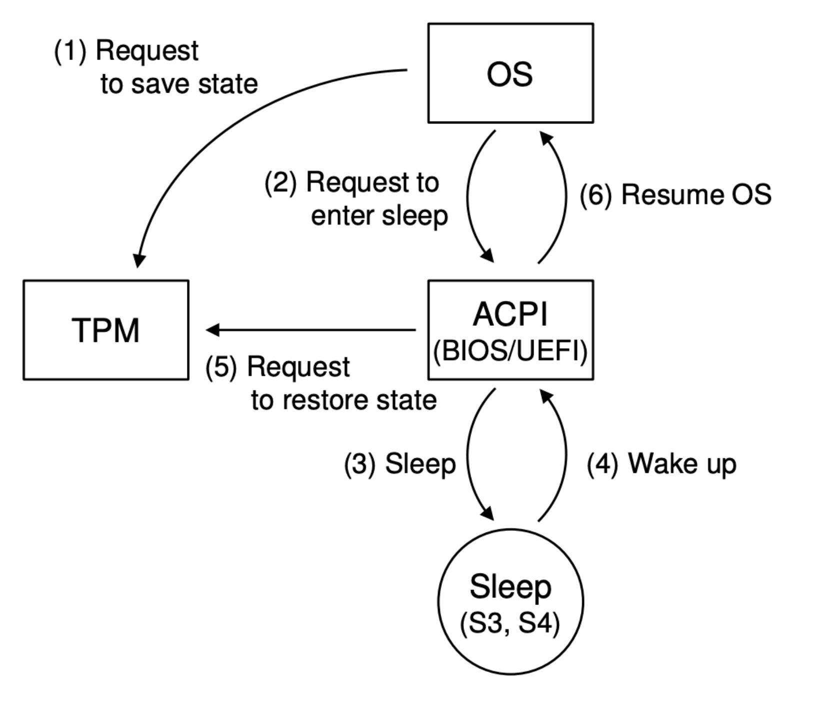

(1) Request to save state
OS
(2) Request to enter sleep
(6) Resume OS
TPM
ACPI (BIOS/UEFI)
(5) Request to restore state
(3) Sleep
(4) Wake up
Sleep (S3, S4)

Figure 2: ACPI sleep process with TPM

Assumptions and Threat Model

Assumptions and Threat Model

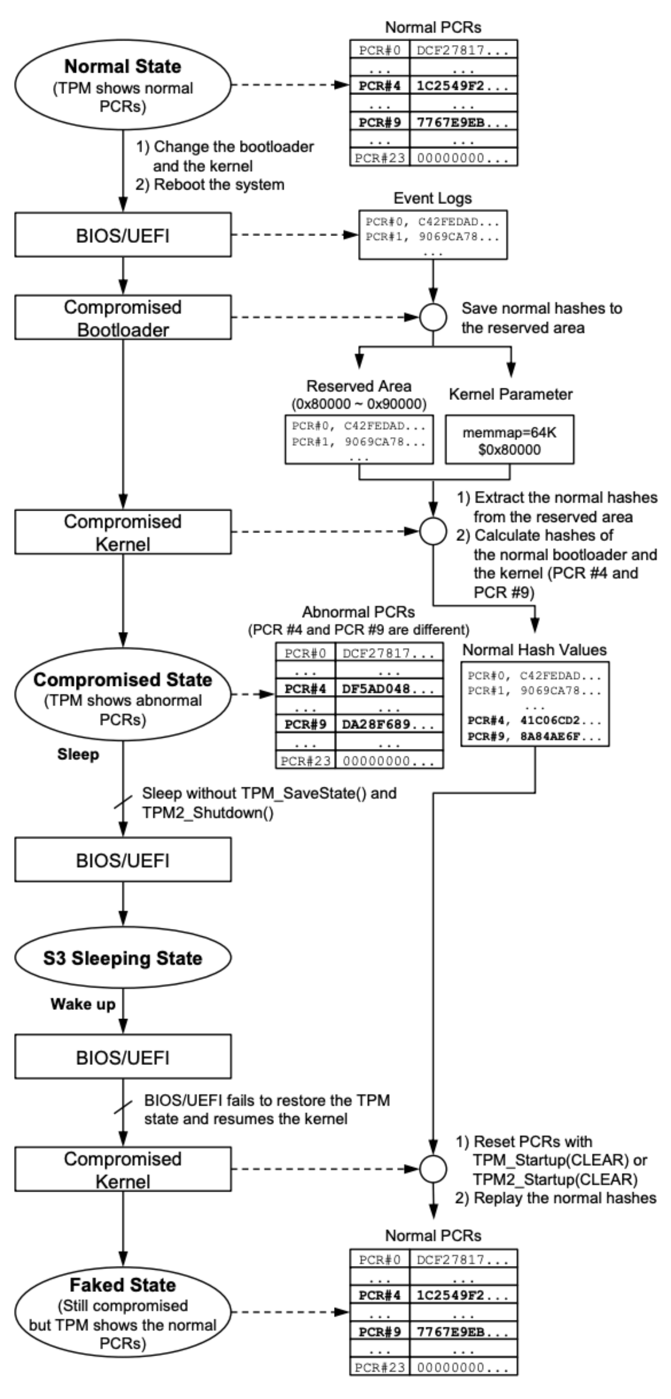
- System utilizes SRTM and DRTM to measure components
  - Measurements stored in PCRs are verified by remote verifier
  - Any bootloader and kernel modification are detected.
- Root privilege
  - Hide the breach and retain root privilege
  - No access to system circuitry
  - Cannot flash the firmware with arbitrary code
- Attacks ignored
  - DOS, Hardware Attack and Vulnerabilities of SMM

Vulnerabilities and Exploits

Vulnerabilities and Exploits - Overview

- SRTM (CVE-2018-6622) – grey area vulnerability
  - 1) Change bootloader and Kernel and reboot the system
  - 2) Bootloader - Save normal hashes to reserved area
  - 3) Kernel Extract normal hashes and calculate hashes from normal bootloader and the kernel
  - 4) Sleep by skipping TPM_SaveState(), S1 and S2
  - 5) On restore replay reset TPM and replay normal hashes

Vulnerabilities and Exploits - Overview SRTM



Normal State (TPM shows normal PCRs)

Normal PCRs

PCR#0	DCF27817...
...	...
PCR#4	**1C2549F2...**
...	...
PCR#9	**7767E9EB...**
...	...
PCR#23	00000000...

1) Change the bootloader and the kernel
2) Reboot the system

BIOS/UEFI

Event Logs

```
PCR#0, C42FEDAD...
PCR#1, 9069CA78...
...
```

Compromised Bootloader

Save normal hashes to the reserved area

Reserved Area (0x80000 ~ 0x90000)

```
PCR#0, C42FEDAD...
PCR#1, 9069CA78...
...
```

Kernel Parameter

memmap=64K $0x80000

Compromised Kernel

1) Extract the normal hashes from the reserved area
2) Calculate hashes of the normal bootloader and the kernel (PCR #4 and PCR #9)

Compromised State (TPM shows abnormal PCRs)

Abnormal PCRs (PCR #4 and PCR #9 are different)

PCR#0	DCF27817...
...	...
PCR#4	**DF5AD048...**
...	...
PCR#9	**DA28F689...**
...	...
PCR#23	00000000...

Normal Hash Values

```
PCR#0, C42FEDAD...
PCR#1, 9069CA78...
...
PCR#4, 41C06CD2...
PCR#9, 8A84AE6F...
```

Sleep

Sleep without TPM_SaveState() and TPM2_Shutdown()

BIOS/UEFI

S3 Sleeping State

Wake up

BIOS/UEFI

BIOS/UEFI fails to restore the TPM state and resumes the kernel

Compromised Kernel

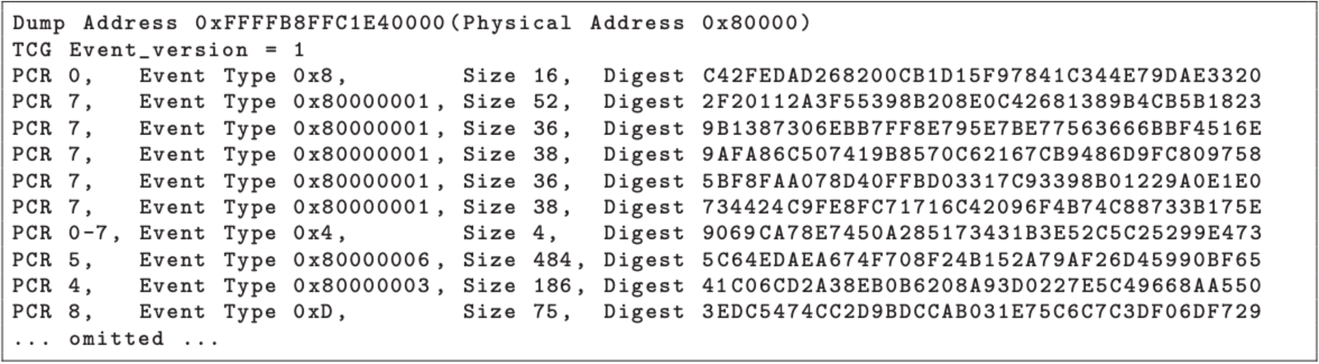
1) Reset PCRs with TPM_Startup(CLEAR) or TPM2_Startup(CLEAR)
2) Replay the normal hashes

Faked State (Still compromised but TPM shows the normal PCRs)

Normal PCRs

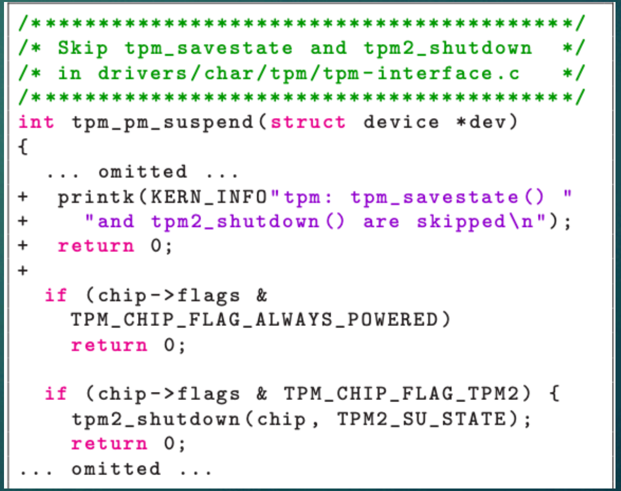
PCR#0	DCF27817...
...	...
PCR#4	**1C2549F2...**
...	...
PCR#9	**7767E9EB...**
...	...
PCR#23	00000000...

1. Process



```
Dump Address 0xFFFFB8FFC1E40000(Physical Address 0x80000)
TCG Event_version = 1
PCR 0,   Event Type 0x8,          Size 16,  Digest C42FEDAD268200CB1D15F97841C344E79DAE3320
PCR 7,   Event Type 0x80000001,   Size 52,  Digest 2F20112A3F55398B208E0C42681389B4CB5B1823
PCR 7,   Event Type 0x80000001,   Size 36,  Digest 9B1387306EBB7FF8E795E7BE77563666BBF4516E
PCR 7,   Event Type 0x80000001,   Size 38,  Digest 9AFA86C507419B8570C62167CB9486D9FC809758
PCR 7,   Event Type 0x80000001,   Size 36,  Digest 5BF8FAA078D40FFBD03317C93398B01229A0E1E0
PCR 7,   Event Type 0x80000001,   Size 38,  Digest 734424C9FE8FC71716C42096F4B74C88733B175E
PCR 0-7, Event Type 0x4,          Size 4,   Digest 9069CA78E7450A285173431B3E52C5C25299E473
PCR 5,   Event Type 0x80000006,   Size 484, Digest 5C64EDAEA674F708F24B152A79AF26D45990BF65
PCR 4,   Event Type 0x80000003,   Size 186, Digest 41C06CD2A38EB0B6208A93D0227E5C49668AA550
PCR 8,   Event Type 0xD,          Size 75,  Digest 3EDC5474CC2D9BDCCAB031E75C6C7C3DF06DF729
... omitted ...
```

2. Event Logs



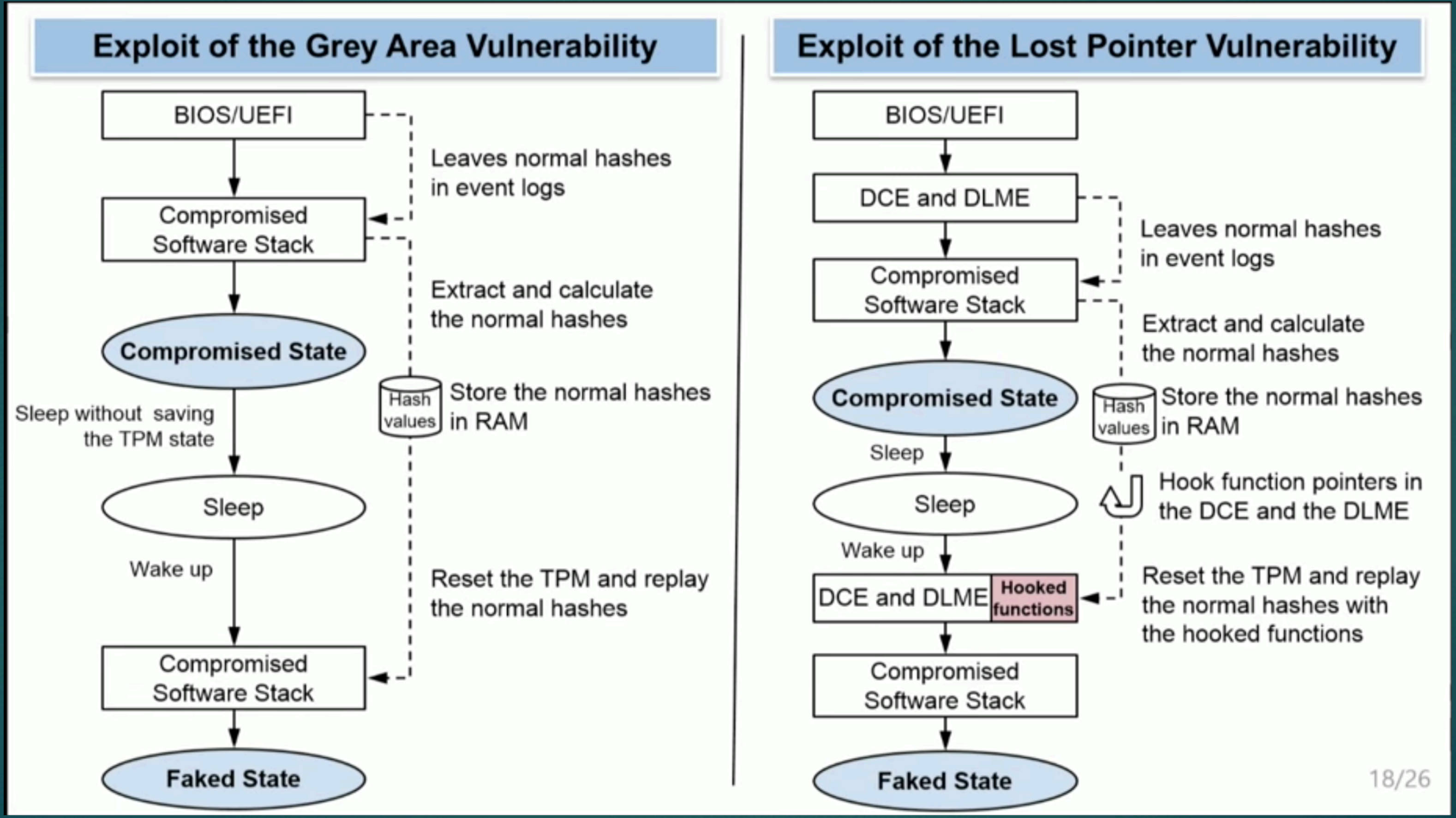
```
/*****************************************/
/* Skip tpm_savestate and tpm2_shutdown  */
/* in drivers/char/tpm/tpm-interface.c   */
/*****************************************/
int tpm_pm_suspend(struct device *dev)
{
  ... omitted ...
+  printk(KERN_INFO"tpm: tpm_savestate() "
+    "and tpm2_shutdown() are skipped\n");
+  return 0;
+
  if (chip->flags &
    TPM_CHIP_FLAG_ALWAYS_POWERED)
    return 0;

  if (chip->flags & TPM_CHIP_FLAG_TPM2) {
    tpm2_shutdown(chip, TPM2_SU_STATE);
    return 0;
... omitted ...
```

3. Custom Kernel patch for TPM reset

Vulnerabilities and Exploits - Overview

- DRTM (CVE-2017-16837)
  - 1) Change the kernel and reboot the system
  - 2) Extract normal hashes from txt logs at tboot (post launch) and function pointers
  - 3) Calculate hashes of the normal kernel
  - 4) Change function pointers to extend normal hashes
  - 5) During sleep hook into tboot function pointers that are not being measured
  - 7) On wake reset all states and replay normal hashes

Vulnerabilities and Exploits - Compare



Exploit of the Grey Area Vulnerability

BIOS/UEFI

Leaves normal hashes in event logs

Compromised Software Stack

Extract and calculate the normal hashes

Compromised State

Hash values

Store the normal hashes in RAM

Sleep without saving the TPM state

Sleep

Wake up

Reset the TPM and replay the normal hashes

Compromised Software Stack

Faked State

Exploit of the Lost Pointer Vulnerability

BIOS/UEFI

DCE and DLME

Leaves normal hashes in event logs

Compromised Software Stack

Extract and calculate the normal hashes

Compromised State

Hash values

Store the normal hashes in RAM

Sleep

Sleep

Hook function pointers in the DCE and the DLME

Wake up

DCE and DLME | Hooked functions

Reset the TPM and replay the normal hashes with the hooked functions

Compromised Software Stack

Faked State

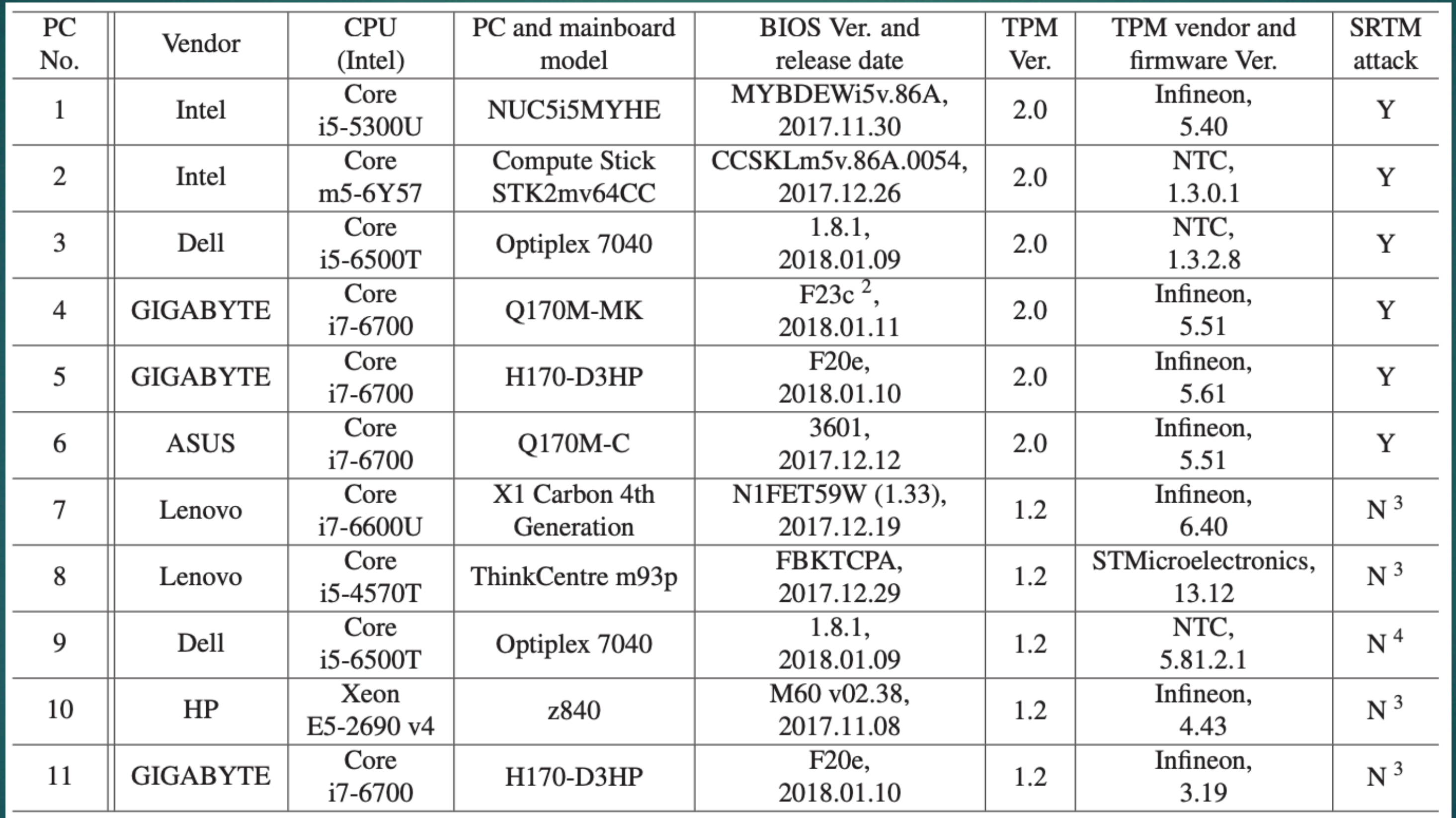
18/26

Evaluation

Evaluation

- Host: Ubuntu 16.04.03
- Kernel: 4.13.0-21-generic
  - Used for removal of tpm_saveState() and tpm_shutdown()
- SRTM attack
  - Source code of CoreOS GRUB 2.0
- DRTM attack
  - Source code from tboot project
  - Requires support of Intel and tboot

Evaluation – cont.



PC No.	Vendor	CPU (Intel)	PC and mainboard model	BIOS Ver. and release date	TPM Ver.	TPM vendor and firmware Ver.	SRTM attack
1	Intel	Core i5-5300U	NUC5i5MYHE	MYBDEWi5v.86A, 2017.11.30	2.0	Infineon, 5.40	Y
2	Intel	Core m5-6Y57	Compute Stick STK2mv64CC	CCSKLm5v.86A.0054, 2017.12.26	2.0	NTC, 1.3.0.1	Y
3	Dell	Core i5-6500T	Optiplex 7040	1.8.1, 2018.01.09	2.0	NTC, 1.3.2.8	Y
4	GIGABYTE	Core i7-6700	Q170M-MK	F23c [2], 2018.01.11	2.0	Infineon, 5.51	Y
5	GIGABYTE	Core i7-6700	H170-D3HP	F20e, 2018.01.10	2.0	Infineon, 5.61	Y
6	ASUS	Core i7-6700	Q170M-C	3601, 2017.12.12	2.0	Infineon, 5.51	Y
7	Lenovo	Core i7-6600U	X1 Carbon 4th Generation	N1FET59W (1.33), 2017.12.19	1.2	Infineon, 6.40	N [3]
8	Lenovo	Core i5-4570T	ThinkCentre m93p	FBKTCPA, 2017.12.29	1.2	STMicroelectronics, 13.12	N [3]
9	Dell	Core i5-6500T	Optiplex 7040	1.8.1, 2018.01.09	1.2	NTC, 5.81.2.1	N [4]
10	HP	Xeon E5-2690 v4	z840	M60 v02.38, 2017.11.08	1.2	Infineon, 4.43	N [3]
11	GIGABYTE	Core i7-6700	H170-D3HP	F20e, 2018.01.10	1.2	Infineon, 3.19	N [3]

Table 4: List of PC and mainboard models and results of the SRTM attack

Countermeasures

Countermeasures

- SRTM
    - Disable S3 sleeping state
        - Some BIOS/UEFI have the option to disable S3 state
    - Enter failure mode if no state to restore
- DRTM
    - Apply patch to the bug
        - Hide the virtual function tables or make them read only
    - Update tboot to latest version

Conclusion

Conclusion

- TPM technology is used across many machines
  - Intended to provide root of trust

- Two vulnerabilities presented
  - Flawed specification in SRTM TPM 2.0
  - Implementation defect DRTM TPM 1.2
    - Flaw in implementation of tboot