CS 6324: Information Security
Trusted Computing

Junia Valente

Department of Computer Science
The University of Texas at Dallas

March 14, 2019
Note:

- Based on a guest lecture I gave for CS 6324: Information Security course on February 26, 2014.
Trusted Computing: Overview

Challenge: software-based security mechanisms are often not sufficient to protect sensitive information and detect the presence of malicious code in a system

Promising solution: Trusted Computing

- Trusted Computing leverages hardware-based security mechanisms to increase the trustworthiness of computer systems
Lecture Objective

Overview of existing techniques for bootstrapping trust in computer systems, i.e., conveying information about a platform’s state to an interested party.

1. Securely collect information of the platform state
2. Securely store sensitive information
3. Securely report on stored information
Lecture Overview: Trusted Computing in a Nutshell

Integrity Measurement

Secure Storage

Measurement Reporting

Integrity Measurement

Secure Storage

Reporting

Root of Trust for Measurements

Root of Trust for Storage

Root of Trust for Reporting

Trusted Platform Module

SIGNED

Integrity

Authenticity
Trusted Platform Module - TPM 1.2

- A **hardware cryptographic chip** attached to modern motherboards. Most PC manufactures use a TPM.
- Core component in Trusted Computing

### Trusted Platform Module (TPM)

- **Cryptographic Co-Processor**
  - Asymmetric en-/decryption (RSA)
  - Digital signature (RSA)
- SHA-1
- HMAC
- Random Number Generator
- Platform Configuration Registers (PCR)
  - Storage for integrity measurements
  - 24 PCRs

### Input / Output
- Protocol en-/decoding
- Enforces access policies

### Opt-In
- Stores TPM state information (e.g., if TPM is disabled)

### Non-Volatile Memory
- Endorsement Key (2048-bit)
- Storage Root Key (2048-bit)
- Owner Auth Secret
A **hardware cryptographic chip** attached to modern motherboards. Most PC manufactures use a TPM.

- Core component in Trusted Computing

**Based on our previous lectures, what is an obvious limitation with TPM 1.2?**
TCG Software Stack: Architecture

TCG Software Stack (TSS)
Architectural Overview

Application

TSP Interface (Tspi)

TSS Service Provider (TSP)

TCS Interface (Tcspi)

TSS Core Services (TCS)

TDDL Interface (Tddli)

TSS Device Driver Library (TDDL)

TDD Interface (Tddi)

TPM Device Driver (TDD)

IBM Software TPM

TPM Emulator

Trusted Platform Module (TPM)

TCG Software Stack (TSS)

TrouSerS

User Mode

Kernel Mode

Hardware
1. Integrity Measurement

2. Secure Storage

3. Measurement Reporting
Measuring Platform Integrity

Compute a hash value (measurement) over a software’s binary

Chain of Trust

Let $S_n$ be the software currently in control of the platform. $S_n$ is responsible for measuring software $S_{n+1}$ before giving it control of the platform. Thus, before executing a next piece of software $S_{n+1}$, $S_n$ measures and records a measurement of $S_{n+1}$. 
Who Measures the First Software?

- Why don’t we let the first software measure itself?
  - Malicious software may lie about its measurements

- Requires a hardware-based root of trust
  - Secure hardware, e.g., secure coprocessors
  - Recent research proposes using general-purpose CPUs

Definition

A root of trust is a component that must always behave in the expected manner.
A hardware-based root of trust initiates the chain of trust by measuring the initial BIOS code. The BIOS then measures and executes the bootloader, and the bootloader measures and executes the operating system.

Note: This process can be extended to allow the OS to record measurements of each application it executes.
Outline

1. Integrity Measurement
2. Secure Storage
3. Measurement Reporting
Securing Measurements

Measurements must be secured otherwise malicious code might erase the record of its presence.

- e.g., if an attacker replaces the OS with a malicious OS, then the bootloader may blindly hand off control to the malicious OS during the boot process. The malicious OS may then attempt to erase any previously recorded measurements.
A hash chain is a technique for recording software measurements.

- Only requires a small amount of secure memory to record an append-only list of code measurements
- Only the current value of the hash chain is necessary to be stored in secure memory
Hardware-based Hash Chain

- Hardware sets aside protected memory registers (e.g., platform configuration registers) that are initialized to known value (e.g., 0) when the computer boots.

- Computes a hash and updates a register with the output of the hash:

\[ V \leftarrow Hash(V || M), \]

where \( V \) is the current value of the register and \( M \) is the measurement for the new code. \( M \) is kept in a log file in an untrusted storage.
TPM-based Trusted Boot Example

1. Measure
   - CRTM
   - BIOS
   - OS Loader
   - OS

2. Record: $\text{PCRi} \leftarrow \text{SHA1} (\text{PCRi} \parallel \text{value})$
   - $\text{PCR-0}$
   - $\text{PCR-1}$
   - $\text{PCR-2}$
   - ...

3. Handoff

Stored Measurement Log (SML)
BIOS Measurements - Integrity Log

```
> sudo cat /sys/kernel/security/tpm0/ascii_bios_measurements
0 29df12526e86201bf08150c03873e95d0e 08 [S-CRTM Version]
0 2aa8d5856c2779c1c141638e0947a49c76385ba 01 [POST CODE]
0 5a07933df15f4d307b41001d319cd43f44ed9dd 80000008
0 762a9922a53f6c6f6f6683008a2a9a23f9e4f9 80000003
0 0f6f411be0c1a8f14c509b9dld76af70e06 [Option ROM]
0 0a8d7e72790d64a8755d1295e7f291ac95aa 07 [S-CRTM Contents]
0 0d9a9611cd47f89e902ed211ca948e8c49f 07 [S-CRTM Contents]
0 b422130486abc5e507b07f41ec13aa4cf5b2c 07 [S-CRTM Contents]
0 7d8a1467b1e1faa63146d1b3e3557c58f547f4d4c 07 [S-CRTM Contents]
0 0a83ed51dd0b400f1e8223978b1a1e6493e93 07 [S-CRTM Contents]
0 3e637ae3fe2f5be52227875db3fl242193a252 07 [S-CRTM Contents]
0 02880007db8802a9a8f5a586f34119de4f771 07 [S-CRTM Contents]
0 fe2ae30d8f17a27f436ed48b3edf0a5c74667 07 [S-CRTM Contents]
0 42ed2e8691877083553f4989d6b07ac53e765d 80000009
0 c807f0d056b6911e0282f0a27174ae888249f 07 [S-CRTM Contents]
0 0906ca978e7450a285173431b3e52c5c252994e473 04 []
0 1906ca978e7450a285173431b3e52c5c252994e473 04 []
0 2906ca978e7450a285173431b3e52c5c252994e473 04 []
0 3906ca978e7450a285173431b3e52c5c252994e473 04 []
0 4906ca978e7450a285173431b3e52c5c252994e473 04 []
0 5906ca978e7450a285173431b3e52c5c252994e473 04 []
0 6906ca978e7450a285173431b3e52c5c252994e473 04 []
0 7906ca978e7450a285173431b3e52c5c252994e473 04 []
0 4f87548ca1cd19af774c3c4d804f4e41dad708e 80000003
0 455a6b782b74564f2e393cd05a588b7fa49a8 80000007
0 475545ddc978d7bf0363a7ef7289f48198f0d 80000007
```
Integrity Log: need root to read on csgrads1

```bash
{csgrads1:/sys/kernel/security/ima} sudo cat ascii_runtime_measurements

^___^
\ (oo)\______
  (\__)\______\/
   \----w
   ||
   {csgrads1:/sys/kernel/security/ima}
```
Integrity Measurement Architecture (IMA) - nowadays

```
root # head /sys/kernel/security/ima/ascii_runtime_measurements
10 ddee6004dc3b4ee30014cd93181c5a2187b59b ima-ng shal:9797edf8d0eed36b1cf92547816051c8af4e5ee boot_aggregate
10 180eafba6fadbcece09b057bc0d55d39f1a8a52 ima-ng shal:db8291bf7d1849a5ee9b0ae01b2e8ebe0051c8af4e5ee /init
10 ac792e08a7cf8de7656003125c7276968d84ea65 ima-ng shal:f778e2082b08d21babc9898f47757a7e8f2a4df /bin/bash
10 0a0d9258c151356204aae2498bbaa4be34d6bb05 ima-ng shal:3ba2e7e0db22c4d17d975def0db881f52dc14859a7 /lib64/ld-2.27.so
10 0d6b1d90350778355f1302d00e59493d0b0011 ima-ng shal:ce8204c948bbf3eae6b94625ad620420c1dc880 /etc/ld.so.cache
10 d69ac2c1d60d28b2da07c7f0cbdbd9e33e99ca277 ima-ng shal:8526466068709356630490ff5156c95a18092b8 /lib64/libc.so.6.0
10 ef3212c12dfb994de9534b0bb9df0c8ea50a77b ima-ng shal:ff08ba928a6e390a80a7a3deef8ae921f0c8ca4e /lib64/libc-2.27.so
10 f805861177a99c61eabebe21003b3c831ccf288b ima-ng shal:261a3cd586363f2f2421662b5b4455df09d941168 /lib64/libncurses.so.6.1
10 52f680881893b28e0f0ce2b132d723a885333500 ima-ng shal:bb53a3fa38e64dfde927de94c333183b85626b0 /lib64/libncurses.so.6.1
10 4da8ce3c51a7814d4e38be355a2a990a5ce88b27 ima-ng shal:99a9c095c7928ecca8c3a4b4b06246fc5f49de /etc/passwd
```
Platform Configuration Registers (PCR)
Outline

1. Integrity Measurement
2. Secure Storage
3. Measurement Reporting
Goal: Use collected measurements (1) to convince the user that the platform has booted into a secure state, and (2) to provide access control to protected storage.
Secure Boot

The platform is in an approved state only if it boots successfully

- Compare the measurement of the code to be loaded to a list of approved measurements
- Halt the boot process if there is an attempt to load unauthorized code
**Fig. 1 Trusted Boot vs. Secure Boot.** The state of a computer system changes as programs run with particular configurations. Trusted boot accumulates a list (L) of measurements for each program executed, but it does not perform any enforcement. Secure boot (§3.1) will halt the system if any attempt is made to execute a program that is not on an approved list (L*). Note that both systems must always measure programs before executing them.
Access Control Mechanism

- Access control mechanism for cryptographic keys based on allowed platform configurations (i.e., platform measurements)

- TPM-based solution for restricting access based on the platform and the platform state:
  - **Bind** and **seal** operations
TPM Bind and Seal Operations

Bind
- Bind (encrypt) secret to the TPM and the platform
- Only the same TPM can unbind (decrypt) the secret

Seal
- Seal (encrypt) secret to the TPM and the platform state
- Specify policy to control the unseal (decrypt) operation, i.e., specify a set of PCR indices and values they must hold
- TPM unseals secret only if the current PCR values (and an optional passcode) match expected values
Bind and Seal Operation: Details

- Bind and seal operations use an asymmetric RSA key generated by the TPM.
- Private part of the keys never leave the TPM in the clear.
- TPM ensures that these keys are used only for encryption and never for signing.
- Note: TPM has limited memory. In reality, only two keys are stored inside the chip: storage root key (SRK) and endorsement key (EK).
- Additional keys are encrypted (using the SRK key) creating a key hierarchy stored in disk.
TPM Keys and Key Hierarchy

Key Hierarchy on Persistent Storage

- **TSS_KEY_TSP_SRK**
  \{0,0,0,0,0,0,0,0,0,0,0,1,0}\}

- **TSS_KEY_TYPE_STORAGE**
  \{0,0,0,0,0,0,0,0,0,0,0,1,0}\}

- **TSS_KEY_TYPE_SIGNING**
  \{0,0,0,0,0,0,0,0,0,0,0,1,1}\}

- **TSS_KEY_TYPE_BIND**
  \{0,0,0,0,0,0,0,0,0,0,0,1,2}\}

  Symmetric Key
Using Platform Measurements Remotely: Attestation

Goal: Convey platform measurements (hash chain) to an external entity in an authentic manner.

Pre-requisite: The trusted boot process securely stores the platform configuration in measurement chains.
An Attestation Protocol

TPM-based Attestation

1. Ask for attestation (nonce)

2. Generate Attestation Keys (sign PCR values and nonce)

3. Respond (signed PCR values and nonce, SML)

4. Verify (signature)

5. Validate (SML)

Remote party can compare attested configuration to a set of reference configurations known to be trustworthy
Reboot Attacks

- A naive implementation of the attestation protocol is subject to a reboot attack:
  - Time-of-check to time-of-use (TOCTOU) vulnerability can be exploited when attesting platform is subject to physical tampering
  - e.g., once verifier has received the attestation, the attacker reset the attestor and boot a malicious software.
  - A solution: Establish a secure channel between attestor and verifier. Then the connection channel breaks if the attestor is reboot.
Conclusion

- Trusted Computing leverages specialized hardware to act as roots of trust to improve the security posture of computer systems

- The TPM is a core component in Trusted Computing
  - TPM has been used in modern computers to securely store the platform integrity measurements and cryptographic keys
  - TPM can act as a root of trust for storage (RTS) and as a root of trust for reporting (RTR)

- TPM is just a beginning into the Trusted Computing world
Summary: Trusted Computing in a Nutshell

Integrity Measurement

Secure Storage

Reporting

Root of Trust for Measurements

Root of Trust for Storage

Root of Trust for Reporting

Trusted Platform Module

*Bootstrapping Trust in Modern Computers*, vol. 10. Springer, 2011.

Extra Reference:

- Trusted Computing Group. URL: www.trustedcomputinggroup.org