Lecture Embedded System Security

Introduction to Trusted Computing

Prof. Dr.-Ing. Ahmad-Reza Sadeghi
System Security Lab
Technische Universität Darmstadt
Summer Term 2017
Roadmap: Trusted Computing

- Motivation
- Notion of trust
- Trusted Computing
  - Chain of trust
  - Integrity measurements
Trustworthiness in distributed IT systems
- Different parties with potentially conflicting requirements involved
- Cryptographic methods are of limited help

The challenges
- How to define “trustworthiness”?  
- How to determine/verify trustworthiness?  
- How could common computing platforms support such functionality?  
  - Even a secure OS cannot verify own integrity

The role of Trusted Computing
- Enable the reasoning about “trustworthiness” of own and other IT systems
Demand for Trusted Computing

- Increasing threats for IT systems
  - Malware, phishing, targeted attacks, etc.
- Inflexibility of traditional secure systems
  - Example: reference monitors
- Improve security of existing IT systems and infrastructures
  - Servers, PCs, mobile phones, embedded systems, VPN, etc.
- Enable new applications with sophisticated (security) requirements
Roadmap: Trusted Computing

- Motivation
- Notion of trust
- Trusted Computing
  - Chain of trust
  - Integrity measurements
Notions of Trust
Trust

- Complicated notion
  - Studied and debated in different areas (social sciences, philosophy, psychology, computer science, ...)

- Notion relating to belief in honesty, truthfulness, competence, reliability, etc. of the trusted entity

- Social trust
  - Belief in the safety or goodness of something because of reputation, association, recommendation, perceived benefit, etc.
Secure, Trusted, Trustworthy

- **Secure**: System or component will not fail with respect to security goals

- **Trusted**: System or component whose failure can break the (security) policy (Trusted Computing Base, TCB)

- **Trustworthy**: Degree to which behavior of a component or system is demonstrably compliant with its stated functionality
Roadmap: Trusted Computing

- Motivation
- Notion of trust
- Trusted Computing
  - Chain of trust
  - Integrity measurements
Trusted Platform (Basic Idea)

- Has trusted components in hardware and software
- Provides a variety of trusted functions
  - In particular a set of cryptographic and security functions
- Ideally creates a foundation of trust for software
- Provides hardware protection for sensitive data
  - Examples: Keys, counters, etc.
- Desired goals in practice
  - Trusted Computing Base (TCB) should be minimized
  - Compatibility to commodity systems
Problems of Existing IT Systems

- Insufficient protection in software and hardware of existing computing platforms
  - Malicious code (e.g., viruses, Trojan horses)
  - Runtime attacks (e.g., return-oriented programming)
  - DMA (Direct Memory Access)
  - No secure storage

- Main reasons
  - High complexity and poor fault isolation of existing operating systems
  - Lack of protection mechanisms in hardware
  - User unawareness and poor usable security
Primary Goals of TC in Practice

- Improve security of (existing) computing platforms
- Reuse existing modules
  - GUI, common OS, etc.
- Applicable to different operating systems
  - No monopoly, room for innovation
- Open architecture
  - Use open standards and open source components
  - Trustworthiness, costs, reliability, compatibility
- Efficient portability
- New applications/business models
  - Providing security needed for underlying applications
    (based on various sets of assumptions and trust relations)
- Avoid potential misuse of trusted computing
1. Metric for code configuration/identity
   - I/O behavior of machine determined by its code
   - Example of a simple metric: Hash value of binary code
     - Problematic when code functionality depends on other code not included in hash digest (e.g., shared or dynamically linked libraries)

2. Integrity verification (Attestation)
   - Allows computing platform to export verifiable information about its properties (e.g., identity and initial state)
   - Comes from the requirement of assuring the code configuration and execution environment of an application located on a remote computing platform

**Desired Primitives and Tools**
Desired Primitives and Tools (ctd.)

3. Secure storage
   - Securely stores data on untrusted storage (e.g. hard disks)
   - Encrypts data and assures that no other entity can decrypt it

4. Strong process isolation
   - Assures process (memory space) separation
   - Prevents a process from reading or modifying the memory used by another process

5. Secure I/O
   - Allows applications to assure the endpoints of input and output operations
   - Assures to the user that he securely interacts with the intended application
Need for Secure Hardware and Software

- **Need for secure hardware**
  - Even a secure operating system cannot verify its own integrity
    - Another party is needed
  - Secure storage (e.g., for cryptographic keys)
  - Isolation of security-critical programs (e.g., by DMA control)
  - Hardware-based random numbers
    - Fundamental to cryptography

- **Need for secure software (operating systems)**
  - Hardening
    - Still too complex and too large TCB (Trusted Computing Base)
  - Complete new design
    - Compatibility problem, low market acceptance
  - Secure virtual machine monitors
    - Allow reuse of legacy software (operating system and applications)
Trusted Computing Group (TCG)

- Consortium of many IT-Enterprises
  - Founded in April 2003
  - www.trustedcomputinggroup.org

- Focus
  - development of hardware-enabled trusted computing and security technologies across multiple platforms/devices

- Publications
  - Various specifications on Trusted Platforms and Infrastructures
TCG Main Specifications

- **Trusted Platform Module (TPM)**
  - Provides a set of immutable cryptographic and security functions

- **Trusted Software Stack (TSS)**
  - Issues low-level TPM requests and receives low-level TPM responses on behalf of higher-level applications
Roadmap: Trusted Computing

- Motivation
- Notion of trust
- Trusted Computing
  - Chain of trust
  - Integrity measurements
Main TCG Concept: Chain of Trust

- Consider Entities $E_0, \ldots, E_n$
- Goal is to gain trust in entity $E_n$
  - Operational standpoint: $E_0$ launches $E_1$, $E_1$ launches $E_2$, etc.
  - To trust $E_n$ one must trust $E_{n-1}$
  - The sequence $E_0, E_1$ to $E_n$ creates a “chain of trust”
- Transitive trust from $E_0$ to $E_1$ to $E_2$, etc.
  - Trusting $E_2$ requires one to trust $E_0$ and $E_1$
  - However: Trusting $E_0$ does not imply that one trusts $E_2$
What is needed to “trust” the chain?
- The identity of each entity $E_i$ in the chain
  - Identity = measurement (according to TCG definition)
  - Example: Hash digest of the binary code of $E_i$
- Generic flow: Each $E_i$ measures $E_{i+1}$ before passing control to it

Who measures $E_0$?
- $E_0$ must be trusted, no mechanism to measure $E_0$
- $E_0$ is called Root of Trust for Measurement (RTM)
Root of Trust for Measurement (RTM)

- Immutable portion of trusted platform’s initialization code executed on every platform boot
- Trust in all integrity measurements based on integrity of RTM
Performing Integrity Measurements

1. RTM measures entity $E$
Performing Integrity Measurements

1. RTM measures entity $E$

2. RTM creates Event Structure in Event Log
   - Event Log contains Event Structures for all measurements extended to the SM
   - Event Log can be stored on (untrusted) storage device (e.g., hard disk)

**Event Log**
- One Event Structure per measurement

**Event Structure**
- Extend Value (hash digest of $E$)
- Extend Data (information about $E$)
Performing Integrity Measurements

1. RTM measures entity $E$
2. RTM creates Event Structure in Event Log
   - Event Log contains Event Structures for all measurements extended to the SM
   - Event Log can be stored on (untrusted) storage device (e.g., hard disk)
3. RTM extends measurement value into SM registers
Performing Integrity Measurements

1. RTM measures entity $E$
2. RTM creates Event Structure in Event Log
   - Event Log contains Event Structures for all measurements extended to the SM
   - Event Log can be stored on (untrusted) storage device (e.g., hard disk)
3. RTM extends measurement value into SM registers
4. RTM executes/passes control to entity $E$
Abstract Model of TCG Concept

**Trusted Platform** $P$
*Provides integrity of host* $H$

**Host** $H$ *(untrusted)*  
- Firmware  
- Operating system  
- Applications

**Attestor** $A$
- Trusted component *(hard- and software)*  
- Securely stores $C_H$

$C_H$ - initial configuration/state of host $H$ when platform $P$ has been booted

User / Adversary

---

**execute**($C_H$) → $H$

$init(C_H)$ ← **RTM**

---

insecure channel  
secure channel
Abstract Model of TCG Concept

**Trusted Platform $P$**
Provides integrity of host $H$

**Host $H$ (untrusted)**
- Firmware
- Operating system
- Applications

**Attestor $A$**
- Trusted component (hard- and software)
- Securely stores $C_H$

**Verifier $V$**

- **Challenge**
- **Response**

**User / Adversary**

- **$C_H$ - initial configuration/state of host $H$ when platform $P$ has been booted**

---

$C_H$ - initial configuration/state of host $H$ when platform $P$ has been booted
Abstract Model of TCG Concept

**Attest:** Verify system integrity

\( C_H \) - initial configuration/state of host \( H \) when platform \( P \) has been booted
Abstract Model of TCG Concept

**Trusted Platform** $P$
- Provides integrity of host $H$

**Host** $H$ (untrusted)
- Firmware
- Operating system
- Applications

**Attestor** $A$
- Trusted component (hard- and software)
- Securely stores $C_H$

**Verifier** $V$
- $H$
  - Can decide whether $C_H$ violates its security requirements
  - Can “bind/seal” data $D$ to a specific (probably secure) configuration/state of $H$

**Challenger / Verifier**
- $H$

**User / Adversary**

**Attest**: Verify system integrity
**Bind/Seal**: Access control depending on system configuration

**$C_H$**: initial configuration/state of host $H$ when platform $P$ has been booted
**$D$**: data to be revealed only if host $H$ is in the (secure) configuration $C_H$

- **insecure channel**
- **secure channel**