TRUSTED COMPUTING

CARSTEN WEINHOLD
Today: Trusted Computing Technology

- Lecture discusses basics in context of TPMs + outlook
- More theoretical concepts also covered in lecture „Distributed Operating Systems“

Things you should have heard about:

- How to use asymmetric encryption
- Concept of digital signatures
- Collision-resistant hash functions
AN EXAMPLE USE CASE

- Anonymity Service
- Microkernel
- TPM
ANONYMITY

ISP

Proxy

Server
ANONYMITY

ISP

MIX

Server

Server
ANONYMITY
Do you spy?
No
SYSTEM LAYERS

AN.ON MIX

OS

Boot Loader

BIOS

Hardware
TPM

Platform Configuration Register

PCR := SHA256( PCR | X )

Picture for illustration purposes only. SHA256 requires TPM 2.0.
BOOTING + TPM

AN.ON MIX

OS

Boot Loader

BIOS

PCR 4490EF83
Remote Attestation
2. test mix
Position: 2 von 3 (Mittlerer Mix)
Betreiber: TU-Dresden, TUDOS/L4
E-Mail: boettcher@os.inf.tu-dresden.de
Standort: Dresden, Saxony, Deutschland

TPM support: detected. Software stack is in expected state.
Ein Einstellungen-Fenster für ein Tor-System ist geöffnet. Der Bildschirm zeigt eine Liste möglicher Mischkaskaden, darunter eine Auswahl an Mischern und ein Feld für den Anonymitätsgrad. 

Unter dem Punkt "Kostenpflichtige Mischkaskaden" sind folgende Optionen aufgelistet:
- Dresden-Dresden
- Forsen-JAP (Test)
- JonDoe-Test Mix
- CookieCacker das cascade
- 127.0.0.1:86544
- Pythagoras-Tulpe-Saturn
- Apfel-Rousseau-Augenklappe
- Ramses-Lite-Jupiter

Einige der Optionen sind aktiviert, und die Anonymitätsgraden sind nicht erreichbar. Die Ports sind ebenfalls nicht erreichbar.

Unter "2. test mix" steht:
- Position: 2 von 3 "Mittlerer Mix"
- Betreiber: TU-Dresden, TUDOS/L4
- E-Mail: boettcher@os.informatik.tu-dresden.de
- Standort: Dresden, Sachsen, Deutschland
- TPM support: keine Unterstützung, Unklarer Zustand des Softwarestacks.
- Zertifikat: verifiziert, gültig (Was bedeutet das?)

Zusätzlich dazu ist ein Button "OK" und ein Hinweis auf "Klicken Sie auf die Misch-Icons, um Informationen über die einzelnen Betreiber dieses Dienstes zu erhalten."
AN.ON

L4

TPM
THE TRUSTED PLATFORM MODULE
TPMs are tightly integrated into platform:
- Soldered on motherboard
- Insecure / for experimentation only: Pluggable modules (PC, Raspberry Pi, ...)
- Built into chipset / SoC
- Implemented in Firmware
- Tamper resistant casing

Widely deployed:
- Business notebooks + desktops
- Windows RT/8/10 tablets + all Windows 11 PCs
TPM OVERVIEW

- TPM is cryptographic coprocessor:
  - **RSA** (encryption, signatures), **AES** (encryption), **SHA-1** (cryptographic hashes)
  - Other crypto schemes (e.g., **DAA**)
  - Random number generator
  - Platform Configuration Registers (**PCRs**)
  - Non-volatile memory
- TPMs are passive devices!
TPM SPECS

- TPMs specified by Trusted Computing Group [2]
- Multiple implementations
- TPM specifications [3,4] cover:
  - Architecture, interfaces, security properties
  - Data formats of input / output
  - Schemes for signatures, encryption, ...
  - TPM life cycle, platform requirements
TPM IDENTITY

- TPM identified by Endorsement Key **EK**:  
  - Generated in manufacturing process  
  - Certified by manufacturer  
  - Unique among all TPMs  
  - Can only decrypt, serves as root of trust  
- Creating entirely new **EK** possible (e.g., for use in corporate environments)  
- Private part of **EK** never leaves TPM
KEY HIERARCHY

- All keys except for **EK** are part of key hierarchy below Storage Root Key **SRK**:
  - **SRK** created when user „takes ownership“
  - Key types: **storage**, **signature**, **identity**, ...
  - Storage keys are parent keys at lower levels of hierarchy (like **SRK** does at root level)
  - Keys other than **EK** / **SRK** can leave TPM:
    - Encrypted under parent key before exporting
    - Parent key required for loading and decrypting
AIKs required for Remote Attestation
- Special key type for remote attestation: Attestation Identity Key (AIKs)
  - TPM creates AIK + certificate request
  - **Privacy CA** checks certificate request + EK, issues certificate and encrypts under EK
  - TPM can decrypt certificate using EK
- **AIK** certificate:
  - „This **AIK** has been created by a valid TPM“
  - TPM identity (EK) cannot be derived from it
BOOTING + TPM

Application

OS

Boot Loader

BIOS

PCR  4490EF83

Authenticated Booting
Remote Attestation with Challenge/Response

TPM_Quote(AIK, Nonce, PCR)

Challenger

4490EF83
AE58B991

System

4490EF83
Applications require secure storage

TPMs can lock data to **PCR values**:

- **TPM_Seal():**
  - Encrypt user data under specified storage key
  - Encrypted blob contains *expected* PCR values

- **TPM_Unseal():**
  - Decrypt encrypted blob using storage key
  - Compare **current** and *expected* PCR values
  - Release user data *only if* PCR values *match*
Only the TPM_SEALED_DATA structure is encrypted
- Sealed data is stored outside the TPM
- Vulnerable to replay attacks:
  - Multiple versions of sealed blob may exist
  - Any version can be passed to TPM
  - TPM happily decrypts, if crypto checks out
- Problem:
  - What if sealed data must be current?
  - How to prevent use of older versions?
TPMs provide **monotonic counters**

- Only two operations: **increment, read**
- Password protected
- Prevent replay attacks:
  - Seal expected value of counter with data
  - After unseal, compare unsealed value with current counter
  - Increment counter to invalidate old versions
TPM SUMMARY

- Key functionality of TPMs:
  - Authenticated booting
  - Remote attestation
  - Sealed memory

- Problems with current TPMs:
  - No (sensible) support for virtualization
  - Can be slow (hundreds of ms / operation)
  - Linear chain of trust
TPMS IN NIZZA ARCHITECTURE
BOOTING + TPM

App A

App B

OS

Boot Loader

BIOS

PCR

83E2FF9A
Use one PCR per application:
- Application measurements independent
- Number of PCRs is limited (usually 24 PCRs)

Use one PCR for all applications:
- Chain of trust / application log grows
- All applications reported in remote attestation (raises privacy concerns)
- All applications checked when unsealing
Idea: per-application PCRs in software:

- Measure only base system into TPM PCRs (microkernel, basic services, TPM driver, ...)
- „Software TPM“ provides „software PCRs“ for each application
- More flexibility with „software PCRs“:
  - Chain of trust common up to base system
  - Extension of chains of trust for applications fork above base system
  - Branches in **Tree of Trust** are independent
SOFTWARE PCRS

App B

App A

App C

TPM Multiplexer

TPM Driver

Loader

Memory

Network

GUI

Secure Storage

I/O Support

Microkernel

PCR: 4490EF83

vPCR(A): 6B17FC28

vPCR(B): 153B9D14

PCR: 4490EF83

vPCR(A): 6B17FC28

vPCR(B): 153B9D14
Operations on software PCRs:
- Seal, Unseal, Quote, Extend
- Add_child, Remove_child

Performed using software keys (AES, RSA)
- Software keys protected with real TPM
- Link between software PCRs and real PCRs: certificate for RSA signature key
A SECOND LOOK AT VPFS
VPFS SECURITY

Inode File

Sealed Memory
83E2FF9A
VPFS can access secrets only, if its own vPCR and the vPCR for the app match the respective expected values.
VPFS SECURITY

- VPFS uses **sealed memory:**
  - Secret encryption key
  - Root hash of Merkle hash tree
- Second use case is **remote attestation:**
  - Trusted backup storage required, because data in untrusted storage can be lost
  - Secure access to backup server needed
  - VPFS challenges backup server: „Will you store my backups..."
A SECOND LOOK AT THE CHAIN OF TRUST
When you press the power button ...

- First code to be run: BIOS boot block (stored in ROM)
- Starts chain of trust:
  - Initialize TPM
  - Hash BIOS into TPM
  - Pass control to BIOS

**Core Root of Trust for Measurement (CRTM)**
- Discussed so far:
  - CRTM & chain of trust
  - How to make components in chain of trust smaller
- **Observation**: BIOS and boot loader only needed for booting
- **Question**: can chain of trust be shorter?
**DRTM**

- **CRTM** starts chain of trust early
- **Dynamic Root of Trust for Measurement (DRTM)** starts it late:
  - Special CPU instructions (AMD: skinit, Intel: senter)
  - Put CPU in known state
  - Measure small „secure loader“ into TPM
  - Start „secure loader“
- **DRTM**: Chain of trust can start anywhere
- Simple: **DRTM** put right below OS
- Smaller TCB:
  - Large and complex BIOS / boot loader removed
  - Small and simple **DRTM** bootstrapper added
- Open Secure Loader **OSLO**: 1,000 SLOC, 4KB binary size [6]
DRTM CHALLENGES

- DRTM remove boot software from TCB
- Key challenges:
  - „Secure loader“ must not be compromised
  - Requires careful checking of platform state
  - Secure loader must actually run in locked RAM, not in insecure device memory
- DRTM can also run after booting OS
BEYOND THE TRUSTED PLATFORM MODULE
Simple implementations in smartphones, etc.

- Non-modifiable boot ROM loads OS
- OS is signed with manufacturer key, checked by ROM-based boot loader
- Small amount of flash integrated into SoC
- Cryptographic co-processor: software can use (but not obtain) encryption key

Not open: closed or secure boot instead of authenticated booting
Intel TDX: 4th Gen Xeon Scalable Processors

Arm Confidential Compute Architecture (CCA) (introduced with Armv9)

TPM support in VMs
- Software TPM: libtpms + SWTPM
- SWTPM runs as process outside VM
- SWTPM identity linked to hardware TPM
TRUSTED EXECUTION ENVIRONMENTS
WHAT IS A TEE?

What is your state? + NONCE

\[
\text{Sig}_K\{ \text{NONCE, Hash(App), "App by AppSoft", "Version1.1", ... } \}
\]
- Computation
- Measurement
- Root of Trust
- Isolation
- Management
- Environment

What is your state? + NONCE

\[ \text{Sig}(\text{NONCE, Hash(App), "App by AppSoft", "Version1.1", ...}) \]
SPLIT TEE?

- Environment
- Compute
- Isolation
- Management
- Measurement
- Root-of-Trust
SPLIT TEE?

Environment    Management

Compute        Measurement

Isolation      Root-of-Trust
MODULAR TEE?

- Environment
- Management
- Compute
- Measurement
- Isolation
- Root-of-Trust
REFERENCES


