Today: Trusted Computing Technology

- Lecture discusses basics in context of TPMs
- 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
ANONYMITY

ISP

Proxy

Server

TU Dresden

Security Architectures
ANONYMITY

ISP

MIX

Server

Server
ANONYMITY

ISP

MIX

MIX

Server

Server
ANONYMITY

AN.ON

[Diagram with four green figures and question marks]
Do you spy?

No
SYSTEM LAYERS

AN.ON MIX
OS
Boot Loader
BIOS
Hardware
Platform Configuration Register

\[
\text{PCR} := \text{SHA-1}(\, \text{PCR} \mid X \,)
\]
BOOTING + TPM

AN.ON MIX

OS

Boot Loader

BIOS

PCR  4490EF83
Remote Attestation
AN.ON

Linux

Windows

TPM
Monolithic
OS
L4/AN.ON

MIX

TPM Driver

Memory

L4.Fiasco

Network Stack

L4Linux

Network Driver

GUI

USB Driver
Klicken Sie auf die Mix-Icons, um Informationen über die einzelnen Betreiber dieses Dienstes zu erhalten.

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.
Zertifikat: verifiziert, gültig (Was bedeutet das?)
<table>
<thead>
<tr>
<th>PCR</th>
<th>00</th>
<th>0b 35 2b e2 2b 1b a1 46 bf 3b b5 4a 29 58 80 13</th>
</tr>
</thead>
<tbody>
<tr>
<td>PCR</td>
<td>01</td>
<td>3a 3f 78 0f 11 a4 b4 99 69 fca 80 cd 6e 3b 22 75</td>
</tr>
<tr>
<td>PCR</td>
<td>02</td>
<td>3a 3f 78 0f 11 a4 b4 99 69 fca 80 cd 6e 3b 22 75</td>
</tr>
<tr>
<td>PCR</td>
<td>03</td>
<td>3a 3f 78 0f 11 a4 b4 99 69 fca 80 cd 6e 3b 22 75</td>
</tr>
<tr>
<td>PCR</td>
<td>04</td>
<td>fa 68 bf fd c1 33 3f ad 5d 7e ff 67 36 7f 3f bd c2 05 51 67</td>
</tr>
<tr>
<td>PCR</td>
<td>05</td>
<td>3a 3f 78 0f 11 a4 b4 99 69 fca 80 cd 6e 3b 22 75</td>
</tr>
<tr>
<td>PCR</td>
<td>06</td>
<td>3a 3f 78 0f 11 a4 b4 99 69 fca 80 cd 6e 3b 22 75</td>
</tr>
<tr>
<td>PCR</td>
<td>07</td>
<td>3a 3f 78 0f 11 a4 b4 99 69 fca 80 cd 6e 3b 22 75</td>
</tr>
<tr>
<td>PCR</td>
<td>08</td>
<td>00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00</td>
</tr>
<tr>
<td>PCR</td>
<td>09</td>
<td>00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00</td>
</tr>
<tr>
<td>PCR</td>
<td>10</td>
<td>00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00</td>
</tr>
<tr>
<td>PCR</td>
<td>11</td>
<td>00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00</td>
</tr>
<tr>
<td>PCR</td>
<td>12</td>
<td>00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00</td>
</tr>
<tr>
<td>PCR</td>
<td>13</td>
<td>00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00</td>
</tr>
<tr>
<td>PCR</td>
<td>14</td>
<td>00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00</td>
</tr>
<tr>
<td>PCR</td>
<td>15</td>
<td>00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00</td>
</tr>
<tr>
<td>PCR</td>
<td>16</td>
<td>00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00</td>
</tr>
<tr>
<td>PCR</td>
<td>17</td>
<td>79 3e 9f a7 5c 23 24 bb ac c0 48 ab f8 cd fd 36 2d 82 dd ae</td>
</tr>
<tr>
<td>PCR</td>
<td>18</td>
<td>00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00</td>
</tr>
<tr>
<td>PCR</td>
<td>19</td>
<td>15 6b f3 58 45 c9 1d 2a de ab cd d6 7b 9b d7 42 dc 21 56 ed</td>
</tr>
<tr>
<td>PCR</td>
<td>20</td>
<td>00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00</td>
</tr>
<tr>
<td>PCR</td>
<td>21</td>
<td>00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00</td>
</tr>
<tr>
<td>PCR</td>
<td>22</td>
<td>00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00</td>
</tr>
<tr>
<td>PCR</td>
<td>23</td>
<td>00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00</td>
</tr>
</tbody>
</table>
Klicken Sie auf die Mix-Icons, um Informationen über die einzelnen Betreiber dieses Dienstes zu erhalten.

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: no support, Unknown state of software stack.

Zertifikat: verifiziert, gültig (Was bedeutet das?)
THE TRUSTED PLATFORM MODULE
TPM HARDWARE

- 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 8/10/RT tablets
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 & PLATFORM

Platform

RAM

CPU

Chipset

BIOS

CRTM

TPM

Reset
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
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
KEY HIERARCHY

EK

SK

AIK

SRK

SK

SK

AIK

AIK

SigK

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

Authenticated Booting

PCR  4490EF83
Remote Attestation with Challenge/Response
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
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

OS

Boot Loader

BIOS

PCR

83E2FF9A
Multiple Apps

- Use one PCR per application:
  - Application measurements independent
  - Number of PCRs is limited (TPM 1.2: max 24)

- 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
EXTENDING TPMS

- 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
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

Implemented for L4: TPM multiplexer Lyon
A SECOND LOOK AT VPFS
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 reliably?“
A CLOSER LOOK AT THE WHOLE PICTURE
- *User cannot just trust what he / she sees on the screen!*

- **Solution:**
  - Remote attestation
  - For example with trusted device:
    - User’s smartphone sends **nonce** to PC
    - PC replies with quote of **nonce** + **PCR** values
    - User can decide whether to trust or not
A SECOND LOOK AT THE CHAIN OF TRUST
When you press the power button ...

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

BIOS boot block is **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?
- **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
First idea: 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 CHALLENGE

- 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
- New **DRTM** can be established anytime
- Flicker [7] approach:
  - Pause legacy OS
  - Execute critical code as **DRTM** using skinit
  - Restore CPU state
  - Resume legacy OS
Pause untrusted legacy OS, stop all CPUs

Execute skinit:
- Start Flicker code as „secure loader“
- Unseal input / sign data / seal output

Restore state on all CPUs
Resume untrusted legacy OS
If needed: create quote with new PCRs

*TCB in order of only few thousand SLOC!*
FLICKER LIMITS

- Problems with Flicker approach:
  - Untrusted OS must cooperate
  - Only 1 CPU active, all other CPUs stopped
  - Secure input and output only via slow TPM functionality (seal, unseal, sign)
  - Works for some server scenarios (e.g., handling credentials)
  - Client scenarios require more functionality (e.g., trusted GUI for using applications)
ISA EXTENSIONS

- **Intel Software Guard Extensions (SGX):** \[9\]
  - Secure enclaves: protected regions of address space for code, stack, heap
  - Sealed memory and remote attestation

- **ARM TrustZone:** \[8\]
  - New processor mode for critical software
  - Private memory partition (accessible only in secure processor mode)
  - Can be used to implement software TPM
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
January 21, 2020:
- Lecture „Resilience“
- Paper Reading Exercise


