Trust Ain’t Easy: Challenges of TEE Security

Niek Timmers
niek@raelize.com
@tieknimmers

Cristofaro Mune
cristofaro@raelize.com
@pulsoid
Overview

• Introduction

• Challenges for TEEs

• Conclusion

• Q&A
We’ve been analyzing and testing TEEs for ~10 years
Incorrect perspective.
Definition?

• A TEE is often believed to be a ‘processor feature’

This is mostly incorrect.
Raelize TEE Reference Model

REE

Hardware Platform

Execution

Memory

I/O

IP

Hardware separation primitives

Root of Trust

Driver

Drivers (Kernel)

Bootloaders

Library

Trusted Application

Trusted Application

System TA

System TA (Kernel)

TEE OS

Trusted Application Driver

Trusted Application Library

REE

TEE
Actually…

- **Separations are fundamental for a TEE**
  - Memory
  - Hardware modules (i.e. IP)
- **Separations are enforced** by hardware controllers
  - Memory Protection Unit (MPU)
  - TrustZone Address Space Controller (TZASC)
  - TrustZone Protection Controller (TZPC)
  - …
Pointers are historically causing headaches...
(e.g. memory addresses)
<table>
<thead>
<tr>
<th>CVE ID</th>
<th>CVE-2020-11256</th>
</tr>
</thead>
<tbody>
<tr>
<td>Title</td>
<td>Use of Out-of-Range Pointer Offset in TrustZone</td>
</tr>
<tr>
<td>Description</td>
<td>Memory corruption due to lack of check of validation of pointer to buffer passed to trustzone</td>
</tr>
</tbody>
</table>

Source: Qualcomm Security Bulletin (January 2021)

Unchecked pointers leading to TEE code execution
Consistency is challenging.
Secure Memory: MMU and Controllers views

- Baseband Modem: AxProt[1] = 1
- Wi-Fi SoC: AxProt[1] = 1
- DMA engine: AxProt[1] = 0

Independently and unrelated.

**MMU Configuration**

**TZASC Configuration**

**TZPC Configuration**

AMBA AXI3 bus

- TZASC
- DDR
- GPU

- ARM TZ core TEE SW
- MMU

- DMA engine
- Touch
- Fingerprint
Fragmented view of secure memory

• No **system-level** view of (secure) memory

• Information spread across many configurations
  - TrustZone controllers, MMU, MPU, ...
  - Secure range configuration in software (i.e. tables)

• No dedicated **functionality** to determine what’s REE or TEE memory
Threat modeling is hard.
Using hardware to cross boundaries

- Design may let hardware IPs unrestricted access to memory
- Use DMA-capable engines to access across boundaries
HW IP separation for TAs is often overlooked
Technology ain’t easy.
Availability is not enough.

- ARMv8.3 pointer authentication
  - Great, but slow adoption...

- Software exploitation mitigations (i.e. ASLR, W^X, canaries, etc.)
  - Common in REEs; but less for TEEs...

- Also... are security features (e.g. Secure Boot) really secure?
Technology has limitations

• Not all **platforms** support advanced security features
  • E.g. No pointer authentication on ARMv7, ARMv8-M, etc.

• Some security features are not **effective** in restricted environments
  • E.g. ASLR implementations in a TEE may enjoy little entropy
Complexity is significant.
Configuration can be challenging

• Securely configuring a TEE is not trivial
  • Controllers, HW modules, registers, memory layout,…

• Dynamic configuration by multiple components
  • Personalization, bootloaders, operating system, etc.

• Maintenance required across product releases
Diverse ecosystem

• Devices are not made by a single entity (e.g. company)
  • E.g. SoC manufacturer is not the developer of the TEE OS

• Multiple entities with different responsibilities
  • E.g. SoC manufacturer is not responsible for configuring the TEE securely

• Inconsistencies at boundaries yield opportunities for attacks
  • E.g. boundary between components
Product certification is sub-optimal.
Certification

- Works well for hardware (immutable)
  - Once evaluated, it will not change anymore

- Works less for software (mutable)
  - Software is dynamic in nature (i.e. updates, etc.)
  - Code base size of a TEE is often large

Do you prefer a certified product or a secure product?
All products are vulnerable... security reduces risks.
In other industries...

Keeping products secure is key...
Provoking thoughts

• Handling security incidents should be the new “NORMAL”
  • This needs a well-defined process

• Why don’t we evaluate and certify THAT process?

• Certifying companies vs certifying (only) their products
Does your organization have a security contact?
The bright side...
Positive developments #1

• New technology is available
  • Actively developed operating systems (i.e. OP-TEE, Trustonic, etc.)
  • Hardware partitioning (i.e. ARM v8.4+)
  • Security hardening features (i.e. ARM v8.3+)

• Check the presentations at LVC2021 on these topics!
Positive developments #2

• Interfaces are (being) standardized
  • ARM Trusted Firmware (i.e. TF-A, TF-M)
  • ARM Platform Security Architecture (PSA) Firmware Framework
  • GlobalPlatform API specification

• Having a proper security posture is becoming more widespread
  • Security contact
  • Collaboration with researchers
  • “Vulnerability reward programs” (aka “Bug bounties”)

30
Let’s wrap up.
Conclusions

• Thorough understanding of a TEE is key for securing it

• Available technology should be used as intended

• Processes should be certified, not only products

• Important lessons can be learned from other industries
Before we end...
Want to find out more?

More details about our research:
https://raelize.com/blog
Thank you! Any questions!?

Niek Timmers
niek@raelize.com
@tieknimmers

Cristofaro Mune
cristofaro@raelize.com
@pulsoid