Uncovering the Mysteries of Trusted Execution Environments: a Software System Perspective

Journées Sécurité - SiF
14 October 2021

Dr Valerio Schiavoni

valerio.schiavoni@unine.ch

University of Neuchâtel, Switzerland
Uncovering the Mysteries of Trusted Execution Environments: a Software System Perspective

Journées Sécurité - SiF
14 October 2021

Dr Valerio Schiavoni

valerio.schiavoni@unine.ch

University of Neuchâtel, Switzerland
(or, the main questions to answered by this presentation)

Agenda

1. **Why** Trusted Execution Environments are important?

2. **What** are TEEs after all?

3. **When** to use or not to use TEEs?

4. **Where** do we find TEEs nowadays?

5. **How** to use TEEs?
Motivating Scenario

- Suppose you want to develop an online service to handle very sensitive data
  - E.g., ECG logs
- Data **privacy** is paramount
  - Only for allowed stakeholders
- Data **integrity** is paramount
  - If data integrity is compromised, risks of false alerts
- The **code** being executed must also be confidential
  - E.g., **algorithms** to compute HR variations and detect health anomalies
What could go wrong?

Local host

- data
- code

host-os

off-chip hardware

CPU

Untrusted

Trusted
What could go wrong?

Lots of bad things!

Local host

- in-process attacks (memory corruption, ROP)
- OS attacks (rootkits,..)
- hardware attacks (cold boot,..)

Untrusted

Trusted

Untrusted

Trusted

Lots of bad things!
What could go wrong?

Local host

- in-process attacks (memory corruption, ROP)
- OS attacks (rootkits, ...)
- hardware attacks (cold boot, ...)

Untrusted

Trusted

data
code

CPU

TEEs to the rescue!

What could go wrong?

Intel SGX

AMD SEV

off-chip hardware

host-os

Untrusted

Trusted

data
code

CPU

TEEs to the rescue!
What could go wrong?

- Intel SGX
- AMD SEV
- off-chip hardware
- host-os
- in-process attacks (memory corruption, ROP)
- OS attacks (rootkits, ...)
- hardware attacks (cold boot, ...)

Untrusted ➔ Trusted

Local host

Enclave creation

enclave code

enclave data

TEE

CPU

TEE to the rescue!
What could go wrong?

in-process attacks (memory corruption, ROP)

OS attacks (rootkits,..)

hardware attacks (cold boot,..)

more about enclaves in a moment

TEEenclave data

enclave code

Enclave creation

TEE

TEE to the rescue!

Untrusted

Trusted

Local host

host-os

off-chip hardware

CPU

valerio.schiavoni@unine.ch - Journées Sécurité - Uncovering the TEEs - 14.10.21
What could go wrong?

more about enclaves in a moment

Intel SGX
AMD SEV
off-chip hardware
host-os

in-process attacks (memory corruption, ROP)
OS attacks (rootkits, ...)
hardware attacks (cold boot, ...)

enclave code
enclave data

Enclave creation

local host

Untrusted

Trusted

Attestation service
optional

mandatory

TEE

CPU

TEEs to the rescue!

valerio.schiavoni@unine.ch - Journées Sécurité - Uncovering the TEEs - 14.10.21
Agenda

1. Why Trusted Execution Environments are important?

   Protect code and data from powerful attackers

2. What are TEEs after all?

3. When to use or not to use TEEs?

4. Where do we find TEEs nowadays?

5. How to use TEEs?
What is a TEE?

- Hardware **protected** area against powerful attacks
- The **content** of the enclaves is **shielded** from:
  - Compromised operating system, compromised system libraries, attackers with physical access to a machine

![Diagram of TEE components]

- **data**
- **code**
- **enclave data**
- **enclave code**
- **Attestation service**
- **host-os**
- **off-chip**
- **Enclave creation**
- **CPU**

**Attestation service**
What is a TEE?

- Hardware **protected** area against powerful attacks
- The **content** of the enclaves is **shielded** from:
  - Compromised operating system, compromised system libraries, attackers with physical access to a machine

![Diagram of TEE architecture]

- **Confidentiality**
- **Integrity**

**Enclave creation**

**Data**

**Code**

**Host-OS**

**CPU**

**off-chip**

**Attestation service**
Confidentiality

- Code and data in the enclave never leave the CPU package unencrypted

➡ Outside the CPU, everything is encrypted

- When memory is read back into cache lines, the CPU decrypts
Confidentiality

- Code and data in the enclave never leave the CPU package unencrypted

→ Outside the CPU, everything is encrypted

- When memory is read back into cache lines, the CPU decrypts
Confidentiality

- Code and data in the enclave never leave the CPU package unencrypted

- Outside the CPU, everything is encrypted

- When memory is read back into cache lines, the CPU decrypts (with the help of the MME)
The **CPU** verify the **integrity** of cache lines

The **CPU** verify the **integrity** of virtual-to-physical addresses

- Intel SGX: MME maintains the root of a Merkle tree
- Arm TrustZone: vendor-specific.
  - Example: Samsung’s Knox uses passive and active counter-measures
- In the case of AMD SEV: no integrity
Intel SGX in a nutshell

- Available since 2015, SkyLake
- Hardware-protected area on die
- Support strong adversarial models
- Split the program in two parts:
  - Untrusted vs. trusted, enclaves
- Code integrity, genuine hardware
- Intel Attestation Service
- Memory limits, EPC, up to 512 MB in recent server-grade processors, up to 128 MB until recently
- Intel SDK, C/C++, Rust SDK, frameworks for legacy systems (Scone, SGX–LKL, graphene–sgx, etc.)
Intel SGX in a nutshell


- Available since 2015, Skylake

- Hardware-protected area on die

- Support strong adversarial models

- Split the program in two parts:
  - Untrusted vs. trusted, enclaves

- Code integrity, genuine hardware

- Intel Attestation Service

- Memory limits, EPC, up to 512 MB in recent server-grade processors, up to 128 MB until recently

- Intel SDK, C/C++, Rust SDK, frameworks for legacy systems (Scone, SGX–LKL, graphene–sgx, etc.)
AMD SEV in a nutshell

- Secure Encrypted Virtualization
- Secure Memory Encryption
- Designed for virtualized systems
- Lack of integrity protection
- SEV–SNMP fixing this
- Attestation
- To fix in hardware?

SEVered: Subverting AMD’s Virtual Machine Encryption
Mathias Morbitzer, Manuel Huber, Julian Horsch and Sascha Wessel
Fraunhofer AISEC
Garching near Munich, Germany
firstname.lastname@aisec.fraunhofer.de

Insecure Until Proven Updated:
Analyzing AMD SEV’s Remote Attestation
Robert Buhren
robert.buhren@sect.tu-berlin.de
Technische Universität Berlin
Security in Telecommunications
Christian Werling
christian.werling@student.hpi.de
Hasso Plattner Institute, Potsdam

EuroSec’18
CCS’19
# SGX vs. SEV

- Memory-bound stressors from `stress-ng`
- The **greener**, the better

<table>
<thead>
<tr>
<th>Memory [MiB]</th>
<th>read64</th>
<th>write64</th>
<th>flip</th>
<th>gray</th>
<th>incdec</th>
<th>inc-nybble</th>
<th>walk-0d</th>
<th>walk-1d</th>
<th>rand-set</th>
<th>ro</th>
<th>rand-sum</th>
<th>zero-one</th>
<th>galpat-0</th>
<th>galpat-1</th>
<th>swap</th>
<th>modulo-x</th>
<th>prime-gray-0</th>
<th>prime-gray-1</th>
<th>prime-incdec</th>
<th>walk-0a</th>
<th>walk-1a</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td><img src="image1.png" alt="Diagram" /></td>
<td><img src="image2.png" alt="Diagram" /></td>
<td><img src="image3.png" alt="Diagram" /></td>
<td><img src="image4.png" alt="Diagram" /></td>
<td><img src="image5.png" alt="Diagram" /></td>
<td><img src="image6.png" alt="Diagram" /></td>
<td><img src="image7.png" alt="Diagram" /></td>
<td><img src="image8.png" alt="Diagram" /></td>
<td><img src="image9.png" alt="Diagram" /></td>
<td><img src="image10.png" alt="Diagram" /></td>
<td><img src="image11.png" alt="Diagram" /></td>
<td><img src="image12.png" alt="Diagram" /></td>
<td><img src="image13.png" alt="Diagram" /></td>
<td><img src="image14.png" alt="Diagram" /></td>
<td><img src="image15.png" alt="Diagram" /></td>
<td><img src="image16.png" alt="Diagram" /></td>
<td><img src="image17.png" alt="Diagram" /></td>
<td><img src="image18.png" alt="Diagram" /></td>
<td><img src="image19.png" alt="Diagram" /></td>
<td><img src="image20.png" alt="Diagram" /></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td><img src="image21.png" alt="Diagram" /></td>
<td><img src="image22.png" alt="Diagram" /></td>
<td><img src="image23.png" alt="Diagram" /></td>
<td><img src="image24.png" alt="Diagram" /></td>
<td><img src="image25.png" alt="Diagram" /></td>
<td><img src="image26.png" alt="Diagram" /></td>
<td><img src="image27.png" alt="Diagram" /></td>
<td><img src="image28.png" alt="Diagram" /></td>
<td><img src="image29.png" alt="Diagram" /></td>
<td><img src="image30.png" alt="Diagram" /></td>
<td><img src="image31.png" alt="Diagram" /></td>
<td><img src="image32.png" alt="Diagram" /></td>
<td><img src="image33.png" alt="Diagram" /></td>
<td><img src="image34.png" alt="Diagram" /></td>
<td><img src="image35.png" alt="Diagram" /></td>
<td><img src="image36.png" alt="Diagram" /></td>
<td><img src="image37.png" alt="Diagram" /></td>
<td><img src="image38.png" alt="Diagram" /></td>
<td><img src="image39.png" alt="Diagram" /></td>
<td><img src="image40.png" alt="Diagram" /></td>
<td></td>
</tr>
<tr>
<td>16</td>
<td><img src="image41.png" alt="Diagram" /></td>
<td><img src="image42.png" alt="Diagram" /></td>
<td><img src="image43.png" alt="Diagram" /></td>
<td><img src="image44.png" alt="Diagram" /></td>
<td><img src="image45.png" alt="Diagram" /></td>
<td><img src="image46.png" alt="Diagram" /></td>
<td><img src="image47.png" alt="Diagram" /></td>
<td><img src="image48.png" alt="Diagram" /></td>
<td><img src="image49.png" alt="Diagram" /></td>
<td><img src="image50.png" alt="Diagram" /></td>
<td><img src="image51.png" alt="Diagram" /></td>
<td><img src="image52.png" alt="Diagram" /></td>
<td><img src="image53.png" alt="Diagram" /></td>
<td><img src="image54.png" alt="Diagram" /></td>
<td><img src="image55.png" alt="Diagram" /></td>
<td><img src="image56.png" alt="Diagram" /></td>
<td><img src="image57.png" alt="Diagram" /></td>
<td><img src="image58.png" alt="Diagram" /></td>
<td><img src="image59.png" alt="Diagram" /></td>
<td><img src="image60.png" alt="Diagram" /></td>
<td></td>
</tr>
<tr>
<td>64</td>
<td><img src="image61.png" alt="Diagram" /></td>
<td><img src="image62.png" alt="Diagram" /></td>
<td><img src="image63.png" alt="Diagram" /></td>
<td><img src="image64.png" alt="Diagram" /></td>
<td><img src="image65.png" alt="Diagram" /></td>
<td><img src="image66.png" alt="Diagram" /></td>
<td><img src="image67.png" alt="Diagram" /></td>
<td><img src="image68.png" alt="Diagram" /></td>
<td><img src="image69.png" alt="Diagram" /></td>
<td><img src="image70.png" alt="Diagram" /></td>
<td><img src="image71.png" alt="Diagram" /></td>
<td><img src="image72.png" alt="Diagram" /></td>
<td><img src="image73.png" alt="Diagram" /></td>
<td><img src="image74.png" alt="Diagram" /></td>
<td><img src="image75.png" alt="Diagram" /></td>
<td><img src="image76.png" alt="Diagram" /></td>
<td><img src="image77.png" alt="Diagram" /></td>
<td><img src="image78.png" alt="Diagram" /></td>
<td><img src="image79.png" alt="Diagram" /></td>
<td><img src="image80.png" alt="Diagram" /></td>
<td></td>
</tr>
<tr>
<td>256</td>
<td><img src="image81.png" alt="Diagram" /></td>
<td><img src="image82.png" alt="Diagram" /></td>
<td><img src="image83.png" alt="Diagram" /></td>
<td><img src="image84.png" alt="Diagram" /></td>
<td><img src="image85.png" alt="Diagram" /></td>
<td><img src="image86.png" alt="Diagram" /></td>
<td><img src="image87.png" alt="Diagram" /></td>
<td><img src="image88.png" alt="Diagram" /></td>
<td><img src="image89.png" alt="Diagram" /></td>
<td><img src="image90.png" alt="Diagram" /></td>
<td><img src="image91.png" alt="Diagram" /></td>
<td><img src="image92.png" alt="Diagram" /></td>
<td><img src="image93.png" alt="Diagram" /></td>
<td><img src="image94.png" alt="Diagram" /></td>
<td><img src="image95.png" alt="Diagram" /></td>
<td><img src="image96.png" alt="Diagram" /></td>
<td><img src="image97.png" alt="Diagram" /></td>
<td><img src="image98.png" alt="Diagram" /></td>
<td><img src="image99.png" alt="Diagram" /></td>
<td><img src="image100.png" alt="Diagram" /></td>
<td></td>
</tr>
</tbody>
</table>
TrustZone in a nutshell

- Two-world separation, one TA at the time
- Lack of built-in attestation service
- 2~5Mb per TA

Watch upcoming talk on OP-TEE for more!
Other TEEs?

- Risc-V:
  - MultiZone
  - KeyStone
  - Penglai

- Since 2017, Google’s Titan M on Android Pixel (since v3)
- IBM SecureBlue & SecureBlue+
- Upcoming new ARM Confidential Compute Architecture (CCA)
1. **Why** Trusted Execution Environments are important?

   *Protect code and data from powerful attackers*

2. **What** are TEEs after all?

   *HW-shielded areas to build stronger systems*

3. **When** to use or not to use TEEs?

4. **Where** do we find TEEs nowadays?

5. **How** to use TEEs?
Why TEEs are good?

- Operations **inside** TEEs run at bare-metal speed
- Strong adversarial models (i.e., compromised OS)
- Orders of magnitude faster than SotA homomorphic encryption

![Graph showing performance comparison between different operations and bit sizes.](image)

- **Microsoft SEAL**
- Google Private Join and Compute?
Why TEEs are not-so-good?

- At least in the current incarnations:
  1. Requires some **craft** from programmers
  2. Might lack fundamental properties
  3. Performances can be poor (goto 1)
  4. Requires good knowledge of system issues
  5. Continuous stream of side-channel attacks (and fancy logos)!

- Followed by a stream of mitigations, patches..

Can target several TEEs
From REE (untrusted) to TEE (trusted) on average more expensive

Energy side-effects, check CPU governors

Devs must know system details
1. **Why** Trusted Execution Environments are important?
   - Protect code and data from powerful attackers

2. **What** are TEEs after all?
   - HW-shielded areas to build stronger systems

3. **When** to use or not to use TEEs? **Not so easy!**
   - Requires awareness on the target scenarios

4. **Where** do we find TEEs nowadays?

5. **How** to use TEEs?
TEEs at the edge

- **Mobile** devices based on Arm processors might support TrustZone
  - Samsung Knox TEE
  - Google Trusty
- **Cheap off-the-shelf IoT devices** with Arm processors
  - Raspberry 3, 3B+, 4
- Intel compute-sticks (size of an USB stick)
Server-grade TEEs

• Intel SGX widely available on server-grade processors
  • EPC limits being lifted
    • With Total Memory Encryption (TME) and MKTME (multi-key), EPC might become obsolete
• AMD processors largely support SEV
  • Next iteration will ship integrity
Cloud Providers

#TEE-as-a-Service#

- **AWS TEE-enabled instances**
  - AWS Nitro Enclaves. Several limits: no persistent storage, no network, no human access
  - Communication via a trusted channel with the instance creating the nitro enclave
  - *(speculation)* Some flavour of TrustZone

- **Azure Confidential Computing instances**
  - Intel-SGX based VM
  - Support for attestation services (both Azure and Intel)
  - Open Enclave [https://openenclave.io/sdk/](https://openenclave.io/sdk/)

- **Google Shielded VMs**
  - Secure boot, vTPM, integrity monitoring
  - [https://cloud.google.com/shielded-vm/](https://cloud.google.com/shielded-vm/)
Agenda

1. **Why** Trusted Execution Environments are important?
   - Protect code and data from powerful attackers

2. **What** are TEEs after all?
   - HW-shielded areas to build stronger systems

3. **When** to use or not to use TEEs? **Not so easy!**
   - Requires awareness on the target scenarios

4. **Where** do we find TEEs nowadays?
   - Edge, Cloud, and everything in between

5. **How** to use TEEs?
How-to TEEs

• If you reach this point, you have solved already the majority of the mysteries surrounding TEEs (why, what, when, where)!
  • Well done, the hard part is beyond you!
• The next (big) task is common in the software-lifecycle:
  • design
  • implement
  • test
  • deploy
  • profile
  • debug
How-to TEEs

• If you reach this point, you have solved already the majority of the mysteries surrounding TEEs (why, what, when, where)

• The next (big) tasks are common in the software-lifecycle:
  • design
  • implement
  • test
  • deploy
  • profile
  • debug

What are the specific difficulties in executing these steps when dealing with TEEs?

will refer to SGX examples
How-to TEEs

- Dealing with legacy-software:
  - Limited choice of implementation languages
  - Rely on intermediate representations:
    - WebAssembly (check-out our Twine system, ICDE’21)
    - Java (check-out our Montsalvat, Middleware’21)
- If you are lucky and work on a new TEE-enabled project:
  - Start from scratch, native C/C++ SDKs, Rust
  - Key decision: split between untrusted/trusted
  - Important performance consequences

Does it have to be this way? No…but at which cost? open-source, contact me
How-to TEEs

• Suppose you ended up with a container-enabled solution
• How to schedule programs in TEE-enabled clusters?
• Prototype extends Kubernetes and Intel SGX driver

Open-source: https://github.com/sebva/sgx-orchestrator
How-to TEEs

• You might find out that your TEE-powered program is slow

• How do you profile it?

• Requires specialised tools

<table>
<thead>
<tr>
<th>Tool</th>
<th>Framework Agnostic</th>
<th>Paging</th>
<th>Enclave Transitions</th>
<th>Orchestrated Applications</th>
<th>Real-Time Reports</th>
</tr>
</thead>
<tbody>
<tr>
<td>LIKWID</td>
<td>✓</td>
<td></td>
<td></td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>perf</td>
<td>✓</td>
<td>x</td>
<td></td>
<td></td>
<td>x</td>
</tr>
<tr>
<td>MemProf</td>
<td>✓</td>
<td>x</td>
<td></td>
<td></td>
<td>x</td>
</tr>
<tr>
<td>TEE-Perf</td>
<td>✓</td>
<td>x</td>
<td></td>
<td></td>
<td>x</td>
</tr>
<tr>
<td>gprof</td>
<td>✓</td>
<td>x</td>
<td></td>
<td></td>
<td>x</td>
</tr>
<tr>
<td>VTune</td>
<td>✓</td>
<td>x</td>
<td></td>
<td></td>
<td>x</td>
</tr>
<tr>
<td>SGX-Perf</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td>x</td>
</tr>
<tr>
<td>SGXTOP</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>TEEMon</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td>✓</td>
</tr>
</tbody>
</table>

(see Middleware’20)

TEEMon
Agenda

1. Why Trusted Execution Environments are important?
   - Protect code and data from powerful attackers

2. What are TEEs after all?
   - HW-shielded areas to build stronger systems

3. When to use or not to use TEEs? Not so easy!
   - Requires awareness on the target scenarios

4. Where do we find TEEs nowadays?
   - Edge, Cloud, and everything in between

5. How to use TEEs?
   - Understand your requirements
Mysteries Solved

1. **Why** Trusted Execution Environments are important?
   - Protect code and data from powerful attackers

2. **What** are TEEs after all?
   - HW-shielded areas to build stronger systems

3. **When** to use or not to use TEEs? **Not so easy**!
   - Requires awareness on the target scenarios

4. **Where** do we find TEEs nowadays?
   - Edge, Cloud, and everything in between

5. **How** to use TEEs?
   - Understand your requirements
Mysteries Solved

Any questions (or mysteries) left?

1. **Why** Trusted Execution Environments are important?
   - Protect code and data from powerful attackers

2. **What** are TEEs after all?
   - HW-shielded areas to build stronger systems

3. **When** to use or not to use TEEs? **Not so easy!**
   - Requires awareness on the target scenarios

4. **Where** do we find TEEs nowadays?
   - Edge, Cloud, and everything in between

5. **How** to use TEEs?
   - Understand your requirements
One Open Challenge

**Mixed TEEs** communication

- In **heterogeneous** deployments, it is probably the norm
- How to build trusted channels between SGX and TZ enclaves?
- How about the next ones (**RISC-V Keystone**, …)
- What new systems can we build out of these primitives?