Towards Distributed and Virtualized Trusted Execution Environments

Zhiqiang Lin
Distinguished Professor of Engineering
zlin@cse.ohio-state.edu

12/06/2022
Trusted Execution Environments (TEEs)
Trusted Execution Environments (TEEs)

First TEE by Trusted Logic and TI

Trusted Execution Environments (TEEs)

First TEE by Trusted Logic and TI

ARM TrustZone

2004 2005 2006 ...
Trusted Execution Environments (TEEs)
Trusted Execution Environments (TEEs)
Trusted Execution Environments (TEEs)

- First TEE by Trusted Logic and TI
- ARM TrustZone
- Intel SGX
- AMD SEV
- First SGX cloud instance by Aliyun
- SEV in Google Cloud Engine
- Google Asylo
- AWS Nitro Enclave

Timeline:
- 2004
- 2005
- 2006
- 2012
- 2013
- 2014
- 2015
- 2016
- 2017
- 2018
- 2019
- 2020
- 2021
Numerous Applications w/ TEEs

1. SGX-based password manager [KKP+18]
2. SGX-based anonymity network [KHH+17]
3. Privacy-preserving data analytics (e.g., [SCF+15]) and machine learning (e.g., [KPM+16, OSF+16]),
4. SGX-based game protection [PAL20]
5. Privacy-preserving contact-tracing (e.g., SafeTrace [eni]) and blockchains [CXZW21]
Our Efforts in TEE Research

3. “Securing Data Analytics on SGX With Randomization”. ESORICS 2017
11. “vSGX: Virtualizing SGX on AMD SEV”. Oakland 2022
13. Controlled Data Races in Enclaves: Attacks and Detection”. USENIX Security 2023
Our Efforts in TEE Research

2. "SGX-Log: Securing System Logs With SGX". ASIACCS 2017
3. "Securing Data Analytics on SGX With Randomization". ESORICS 2017
8. "Towards Memory Safe Enclave Programming with Rust-SGX". CCS 2019
11. "vSGX: Virtualizing SGX on AMD SEV". Oakland 2022
12. "Towards A TEE-based V2V Protocol For Connected And Autonomous Vehicles". AutoSec 2022
13. Controlled Data Races in Enclaves: Attacks and Detection. USENIX Security 2023
Problems in Current TEEs: (1) Vendor lock-in and (2) Slow evolution

1. SGX is an ISA extension → apps have to be written specifically for SGX and can’t run else where (e.g., you can’t run it on AMD chips)

2. Enumerous side channel vulnerabilities and attacks (e.g., [XCP15, SCNS16, SWG+17, BMD+17, HCP17, GESM17, LSG+17, VBMW+18, vSMÖ+19, SLM+19, CGG+19, CCX+19]) → The defense requires faster patching
The Need to Decouple TEEs from the Hardware (Virtualized TEE)

Recent Trend

1. Unifying TEE SDKs
   - Asoly by Google [Goo], by Vmware [VMw]

2. Software defined TEE
   - Komodo [FBHP17], but not binary compatible
The Need to Decouple TEEs from the Hardware (Virtualized TEE)

Recent Trend
1. Unifying TEE SDKs
   - Asoly by Google [Goo], by Vmware [VMw]
2. Software defined TEE
   - Komodo [FBHP17], but not binary compatible

Benefits of Software Defined TEE
1. No vendor lock-in
2. Flexibility on deployment
3. Faster feature evolution
4. Faster bug fixes
5. Migration
The Need to Decouple TEEs from the Hardware (Virtualized TEE)

Recent Trend

1. Unifying TEE SDKs
   - Asoly by Google [Goo], by Vmware [VMw]
2. Software defined TEE
   - Komodo [FBHP17], but not binary compatible

Benefits of Software Defined TEE

1. No vendor lock-in
2. Flexibility on deployment
3. Faster feature evolution
4. Faster bug fixes
5. Migration

Our goal: Can we design a binary compatible, software defined TEE? If so, how?
### SGX 101: SGX Instructions

<table>
<thead>
<tr>
<th>SGX Version</th>
<th>User Space enclu</th>
<th>Kernel Space encls</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>SGX-v1</td>
<td>5</td>
<td>13</td>
<td>18</td>
</tr>
<tr>
<td>SGX-v2</td>
<td>5+3</td>
<td>13+3</td>
<td>18+6</td>
</tr>
</tbody>
</table>
SGX 101: SGX Instructions

<table>
<thead>
<tr>
<th>Privilege</th>
<th>Type</th>
<th>Instruction</th>
<th>Description</th>
<th>Version</th>
</tr>
</thead>
<tbody>
<tr>
<td>P</td>
<td>MEM</td>
<td>EADD</td>
<td>Add a page</td>
<td>v1</td>
</tr>
<tr>
<td>P</td>
<td>MEM</td>
<td>EBLOCK</td>
<td>Block an EPC page</td>
<td>v1</td>
</tr>
<tr>
<td>P</td>
<td>EXE</td>
<td>ECREATE</td>
<td>Create an enclave</td>
<td>v1</td>
</tr>
<tr>
<td>P</td>
<td>DBG</td>
<td>EDBGRD</td>
<td>Read data by debugger</td>
<td>v1</td>
</tr>
<tr>
<td>P</td>
<td>DBG</td>
<td>EDBGWR</td>
<td>Write data by debugger</td>
<td>v1</td>
</tr>
<tr>
<td>P</td>
<td>MEM</td>
<td>EEXTEND</td>
<td>Extend EPC page measurement</td>
<td>v1</td>
</tr>
<tr>
<td>P</td>
<td>EXE</td>
<td>EINIT</td>
<td>Initialize an enclave</td>
<td>v1</td>
</tr>
<tr>
<td>P</td>
<td>MEM</td>
<td>ELDB</td>
<td>Load an EPC page as blocked</td>
<td>v1</td>
</tr>
<tr>
<td>P</td>
<td>MEM</td>
<td>ELDU</td>
<td>Load an EPC page as unblocked</td>
<td>v1</td>
</tr>
<tr>
<td>P</td>
<td>SEC</td>
<td>EPA</td>
<td>Add version array</td>
<td>v1</td>
</tr>
<tr>
<td>P</td>
<td>MEM</td>
<td>EREMOVE</td>
<td>Remove a page from EPC</td>
<td>v1</td>
</tr>
<tr>
<td>P</td>
<td>MEM</td>
<td>ETACK</td>
<td>Activate EBLOCK checks</td>
<td>v1</td>
</tr>
<tr>
<td>P</td>
<td>MEM</td>
<td>EWB</td>
<td>Write back/invalidate an EPC page</td>
<td>v1</td>
</tr>
<tr>
<td>P</td>
<td>MEM</td>
<td>EAUG</td>
<td>Allocate page to an existing enclave</td>
<td>v2</td>
</tr>
<tr>
<td>P</td>
<td>SEC</td>
<td>EMODPR</td>
<td>Restrict page permissions</td>
<td>v2</td>
</tr>
<tr>
<td>P</td>
<td>EXE</td>
<td>EMODT</td>
<td>Change the type of an EPC page</td>
<td>v2</td>
</tr>
<tr>
<td>U</td>
<td>EXE</td>
<td>EENTER</td>
<td>Enter an enclave</td>
<td>v1</td>
</tr>
<tr>
<td>U</td>
<td>EXE</td>
<td>EEXIT</td>
<td>Exit an enclave</td>
<td>v1</td>
</tr>
<tr>
<td>U</td>
<td>SEC</td>
<td>EGETKEY</td>
<td>Create a cryptographic key</td>
<td>v1</td>
</tr>
<tr>
<td>U</td>
<td>SEC</td>
<td>EREPORT</td>
<td>Create a cryptographic report</td>
<td>v1</td>
</tr>
<tr>
<td>U</td>
<td>EXE</td>
<td>ERESUME</td>
<td>Re-enter an enclave</td>
<td>v1</td>
</tr>
<tr>
<td>U</td>
<td>MEM</td>
<td>EACCEPT</td>
<td>Accept changes to a page</td>
<td>v2</td>
</tr>
<tr>
<td>U</td>
<td>SEC</td>
<td>EMODPE</td>
<td>Enhance access rights</td>
<td>v2</td>
</tr>
<tr>
<td>U</td>
<td>MEM</td>
<td>EACCEPTCOPY</td>
<td>Copy page to a new location</td>
<td>v2</td>
</tr>
</tbody>
</table>
SGX 101: ECREATE, EADD; EEXTEND, EINIT
SGX 101: ECREATE, EADD; EEXTEND, EINIT
SGX 101: ECREATE, EADD; EEXTEND, EINIT
**SGX 101: ECREATE, EADD; EEXTEND, EINIT**

- **Intel SGX CPU**
- **OS**
- **Untrusted App**
- **Enclave**
- **Hash**
- **Signed Hash**

Diagram illustrating the interaction between the Intel SGX CPU, OS, Untrusted App, Enclave, and Signed Hash.
SGX 101: ECREATE, EADD; EEXTEND, EINIT
SGX 101: EENTER, EEXIT; AEX, ERESUME

Enclave

entry_point_1:
...
...
entry_point_2:
...
...
...

Untrusted App

do_ecall:
EENTER
post_ecall:
...
...

OS

Intel SGX CPU
SGX 101: EENTER, EEXIT; AEX, ERESUME
SGX 101: Enclave Memory in SGX

Enclave

... 
mov var, %rax
...

tvar:
.byte 00 00

Untrusted App

var:
.byte 00 00

OS

Intel SGX CPU
SGX 101: Attestation in SGX

My Server

Proof

Internet

My Enclave

Untrusted App

OS

Intel SGX CPU
AMD Secure Encrypted Virtualization (SEV)
vSGX: Virtualizing SGX Enclaves on AMD SEV
vSGX: Virtualizing SGX Enclaves on AMD SEV

Design Goals

1. Binary compatibility
2. Comparable security guarantee with both SGX and SEV
3. Reasonable performance
Design Goals

1. Binary compatibility
2. Comparable security guarantee with both SGX and SEV
3. Reasonable performance

vSGX should work like an SGX module plugged onto an SEV machine
Challenges

1. How to isolate the enclave from others components?
2. How to execute SGX instructions on SEV?
3. How to handle memory access inside an enclave?
4. How to connect between an enclave and other components?
vSGX Architecture

TCB

- EVM
  - Enclave Manager
  - Enclave
  - Enclave Kernel
    - Instruction Emulation
    - Memory Management
    - Cross-VM Communication

Untrusted

- AVM
  - App
  - App VM Module
    - Instruction Emulation
    - Memory Management
    - Cross-VM Communication

Hypervisor

- vSGX Hub
- KVM Module

Root of Trust

AMD SEV Hardware

Motivations
Background
Virtualized TEE
TEE-based V2V Protocol
Related Work
Takeaway
References
# SGX Instructions Emulation in vSGX

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>EADD</td>
<td>Add an page to an uninitialized enclave</td>
<td>✓</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>● ← ○</td>
<td>4185</td>
<td>19</td>
</tr>
<tr>
<td>EAUG</td>
<td>Add an page to an initialized enclave</td>
<td>✓</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
<td>● &lt; ○</td>
<td>25</td>
<td>19</td>
<td></td>
</tr>
<tr>
<td>EBLOCK</td>
<td>Block an EPC page</td>
<td>✓</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
<td>● ← ○</td>
<td>9</td>
<td>19</td>
<td></td>
</tr>
<tr>
<td>ECREATE</td>
<td>Create a SECS page in EPC</td>
<td>✓</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
<td>● ← ○</td>
<td>4105</td>
<td>19</td>
<td></td>
</tr>
<tr>
<td>EDBGRD</td>
<td>Read from a debug enclave</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>EDBGWR</td>
<td>Write to a debug enclave</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>EEEXTEND</td>
<td>Extend uninitialized enclave's measurement</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>• ← ○</td>
<td>9</td>
<td>19</td>
<td></td>
</tr>
<tr>
<td>EINIT</td>
<td>Initialize an enclave</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>● ← ○</td>
<td>2137</td>
<td>19</td>
<td></td>
</tr>
<tr>
<td>ELDL/ELDU</td>
<td>Load a page to enclave</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>● ← ○</td>
<td>8370</td>
<td>4131</td>
<td></td>
</tr>
<tr>
<td>EMODPR</td>
<td>Restrict an EPC page's permission</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>● ← ○</td>
<td>12</td>
<td>19</td>
<td></td>
</tr>
<tr>
<td>EMODT</td>
<td>Change an EPC page's type</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>● ← ○</td>
<td>12</td>
<td>19</td>
<td></td>
</tr>
<tr>
<td>EPA</td>
<td>Add version array</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>● ← ○</td>
<td>9</td>
<td>4131</td>
<td></td>
</tr>
<tr>
<td>EREMOVE</td>
<td>Remove a page from EPC</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>✓</td>
<td>● ← ○</td>
<td>9</td>
<td>19</td>
<td></td>
</tr>
<tr>
<td>ETRACK</td>
<td>Block until EBLOCK is done</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>● ← ○</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>EWB</td>
<td>Write an EPC page to main memory</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>● ← ○</td>
<td>4137</td>
<td>8355</td>
<td></td>
</tr>
<tr>
<td>EACCEPT</td>
<td>Accept changes to an EPC page</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>● ← ○</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>EACCEPTCOPY</td>
<td>Copy a page to a new EPC page</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>● ← ○</td>
<td>177</td>
<td>19</td>
<td></td>
</tr>
<tr>
<td>ENTER</td>
<td>Enter an enclave</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>● ← ○</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>EEXIT</td>
<td>Exit an enclave</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>● ← ○</td>
<td>153</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>EGKEY</td>
<td>Derive a key</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>● ← ○</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>EMODPE</td>
<td>Extend permission of an EPC page</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>● ← ○</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>EREPORT</td>
<td>Create a cryptographic report</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>● ← ○</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>ERESUME</td>
<td>Resume an enclave</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>● ← ○</td>
<td>33</td>
<td>19</td>
<td></td>
</tr>
</tbody>
</table>

## Behavior

| AEX               | Exit an enclave due to interrupt or fault | ✓* | ✗ | ✗ | ✗ | ✗ | ● ← ○ | 166 | - |

---

16 / 37
Cross VM Communication

VM2 – Cross-VM Communication
- Crypto Engine
- Dispatcher
- Dispatch Queue
- IRQ Handler
- Destination

VM1 – Cross-VM Communication
- Source
- Crypto Engine
- Sender
- Data Packer

Hypervisor – vSGX Hub
- CPUID Handler
- Send Worker
- Send Queue
Cross VM Communication

**VM2 – Cross-VM Communication**
- Crypto Engine
- Dispatcher
- Dispatch Queue

**VM1 – Cross-VM Communication**
- Source
- Sender
- Data Packer

**Hypervisor – vSGX Hub**
- CPUIID Handler
- Send Worker
- Send Queue
Control Flow Transferring: EENTER, EEXIT
Control Flow Transferring: EENTER, EEXIT
Performance - BYTEmark

(a) CPU Intensive Test

(b) Memory Intensive Test

(c) FP Intensive Test
Performance - Graphene

(a) Time Consumption Launching Graphene SGX on vSGX

(b) cURL Execution Time

(c) GMPbench 0.2 Score
Connected and Automotive Vehicles (CAVs)

- Vehicle To Infrastructure (V2I)
- Vehicle To Vehicle (V2V)
- Benefits
  1. Traffic Safety
  2. Traffic Efficiency
  3. Energy Savings
V2V Protocol Requirements

1. (Instant) Authentication
2. Message Integrity (Tamper Resiliency)
3. Message Confidentiality (Privacy)
4. Anti-tracking (Privacy)
5. Revocation and Accountability
6. Scalability
PKI-based V2V Encrypted Broadcast
PKI-based V2V Encrypted Broadcast

Vehicle V1

Cert

Vehicle V2

G

V1

V1 || G

V1

V2
PKI-based V2V Encrypted Broadcast

Vehicle V1

Cert\_V1 || G\_V1

Vehicle V2

\textbf{Verify} Cert\_V1
\textbf{Check} Cert\_V1 = \text{? CRL\_REVOKED}
PKI-based V2V Encrypted Broadcast
PKI-based V2V Encrypted Broadcast

Vehicle V1

Cert_{V1} \| G^{V1}

Cert_{V2} \| G^{V2}

Verify Cert_{V1}
Check Cert_{V1} \neq \text{CRL}_{\text{REVOKED}}
Derive K_{DH} \leftarrow G^{V1*V2}

Vehicle V2

Verify Cert_{V1}
Check Cert_{V1} \neq \text{CRL}_{\text{REVOKED}}
Derive K_{DH} \leftarrow G^{V2*V1}
PKI-based V2V Encrypted Broadcast

- **Vehicle V1**
  - Send: $\text{Cert}_{V1} \parallel G^{V1}$
  - Verify: $\text{Cert}_{V1}$
  - Check: $\text{Cert}_{V1} \neq \text{CRL}_{REVOKED}$
  - Derive: $K_{DH} \leftarrow G^{V1*V2}$

- **Vehicle V2**
  - Send: $\text{Cert}_{V2} \parallel G^{V2}$
  - Verify: $\text{Cert}_{V2}$
  - Check: $\text{Cert}_{V2} \neq \text{CRL}_{REVOKED}$
  - Derive: $K_{DH} \leftarrow G^{V2*V1}$

- Message: $\text{BSM}^K_{DH}, \text{CMAC}$

- **Key Points**:
  - Use PKI for certificate validation.
  - Implement key derivation for secure communication.
  - Use CMAC for message integrity and authenticity.
PKI-based V2V Encrypted Broadcast

![Diagram of PKI-based V2V Encrypted Broadcast](image)

**Verification**
- Verify $\text{Cert}_{V1}$
- Check $\text{Cert}_{V1} = \text{CRL}_{\text{REVOKED}}$

**Key Derivation**
- Derive $K_{DH} \leftarrow G^{V1*V2}$
- Derive $K_{DH} \leftarrow G^{V2*V1}$

**Message Processing**
- $\{\text{BSM}\}K_{DH}, \text{CMAC}$

**Decryption and Process**
- Decrypt BSM
- Process BSM
Our Approach

Key Objective

1. (Instant) Authentication
2. Message Integrity (Tamper Resiliency)
3. Message Confidenality (Privacy)
4. Anti-tracking (Privacy)
5. Revocation and Accountability
6. Scalability
### Our Approach

#### Key Objective

1. **(Instant) Authentication**
2. Message Integrity (Tamper Resiliency)
3. Message Confidentiality (Privacy)
4. **Anti-tracking** (Privacy)
5. Revocation and **Accountability**
6. Scalability

#### Our Approach

- Leveraging TEEs to protect secrets
- Using a shared Daily Symmetric (DS) Key among vehicles
- Storing DS key in the enclave
- Using Remote Attestation to ensure the right environment
- Generating temporal IDs for privacy
**Our Approach**

### Key Objective

1. **(Instant) Authentication**
2. Message Integrity (Tamper Resiliency)
3. Message Confidentiality (Privacy)
4. **Anti-tracking** (Privacy)
5. Revocation and **Accountability**
6. Scalability

---

**Our Approach**

- Leveraging TEEs to protect secrets
- Using a shared Daily Symmetric (DS) Key among vehicles
- Storing DS key in the enclave
- Using Remote Attestation to ensure the right environment
- Generating temporal IDs for privacy

---

The Protocol: Manufacture Phase (One Time)

Collaborate and Integrate into Vehicle On-Board unit
The Protocol: Registration Phase (One Time)

Signature Keys: $s_{BMV}, p_{BMV}$

Fused Keys: $K_{Seal}, K_{PV}$

ID: VIN

TEE Designer Server
The Protocol: Registration Phase (One Time)

- **Signature Keys:** $s_{BMV}, p_{BMV}$
- **TEE Designer Server**
- **Fused Keys:** $K_{Seal}, K_{PV}$
- **ID:** VIN
- **Enclave Initialize Request**
The Protocol: Registration Phase (One Time)

Fused Keys: $K_{Seal}, K_{PV}$

ID: VIN

Signature Keys: $sk_{BMV}, pk_{BMV}$

TEE Designer Server

Enclave Initialize Request

TLS Connect
The Protocol: Registration Phase (One Time)
The Protocol: Registration Phase (One Time)

Fused Keys: $K_{Seal}, K_{PV}$

Signature Keys: $sk_{BMV}, pk_{BMV}$

ID: VIN

Session Key: $K$

{Enclave Program}$K$

Enclave Initialize Request

TLS Connect

TEE Designer Server

K
The Protocol: Registration Phase (One Time)

- **Signature Keys:** $s_{BMV}$, $p_{BMV}$
- **Fused Keys:** $K_{Seal}$, $K_{PV}$
- **ID:** VIN
- **Session Key:** $K$

1. Enclave Initialize Request
2. TLS Connect
3. {Enclave Program}$K$
4. Enclave Install
5. Boot-Up Application
6. Check Enclave Integrity
The Protocol: Registration Phase (One Time)

**Fused Keys:** $K_{\text{Seal}}, K_{\text{PV}}$

**ID:** VIN

**Signature Keys:** $sk_{\text{BMV}}, pk_{\text{BMV}}$

**Enclave Initialize Request**

**Enclave Install**

**Boot-Up Application**

**Check Enclave Integrity**

**Remote Attest SGX Platform**
The Protocol: Registration Phase (One Time)
The Protocol: Registration Phase (One Time)

Fused Keys: $K_{Seal}, K_{PV}$

ID: VIN

Signature Keys: $sk_{BMV}, pk_{BMV}$

Enclave Initialize Request

TLS Connect

Session Key: $K$

{Enclave Program}$K$

Enclave Install

Boot-Up Application

Check Enclave Integrity

Remote Attest SGX Platform

{Enclave Approve}$K$

{VIN, Owner Info}$K$
The Protocol: Registration Phase (One Time)

Signature Keys: \( s_{BMV}, p_{BMV} \)

Enclave Initialize Request

TLS Connect

Session Key: \( K \)

Enclave Install

Boot-Up Application

Check Enclave Integrity

Remote Attest SGX Platform

Fused Keys: \( K_{\text{Seal}}, K_{\text{PV}} \)

ID: VIN

Enclave Program
The Protocol: Registration Phase (One Time)

**Fused Keys:** $K_{\text{Seal}}, K_{\text{PV}}$

**ID:** VIN

**Signature Keys:** $sk_{\text{BMV}}, pk_{\text{BMV}}$

**Session Key:** $K$

**Enclave Initialize Request**

**TLS Connect**

**Enclave Install**

**Boot-Up Application**

**Check Enclave Integrity**

**Remote Attest SGX Platform**

**VIN Database**

ADD (VIN, VRK, ...)

**Seal (VRK)**

**{Enclave Approve}$K$**

**{VIN, Owner Info}$K$**

**{VRK}$K$**

**TEE Designer Server**

$K_{\text{PV}}$
The Protocol: DS Key Renewal (On Demand)

**Fused Keys:**
- $K_{Seal}$
- $K_{PV}$

**ID:** VIN

**Signature Keys:**
- $sk_{BMV}$
- $pk_{BMV}$

**TEE Designer Server:**
- $K_{PV}$
The Protocol: DS Key Renewal (On Demand)

- **Fused Keys:** $K_{Seal}$, $K_{PV}$
  - **ID:** VIN

- **Signature Keys:** $sk_{BMV}$, $pk_{BMV}$

- **DS Key Database ENTRY:** ($K_{DS}$, ActiveGeo, ActiveDate)

- **TEE Designer Server**
  - $K_{PV}$
The Protocol: DS Key Renewal (On Demand)

**Motivations**

**Background**

**Virtualized TEE**

**TEE-based V2V Protocol**

**Related Work**

**Takeaway**

**References**

Fused Keys: $K_{Seal}, K_{PV}$

ID: VIN

Signature Keys: $sk_{BMV}, pk_{BMV}$

**TEE Designer Server**

$K_{PV}$

**DS Key Database**

ENTRY ($K_{DS}$, ActiveGeo, ActiveDate)

DS Key Request
The Protocol: DS Key Renewal (On Demand)

Fused Keys: $K_{Seal}, K_{PV}$
ID: VIN

Signature Keys: $sk_{BMV}, pk_{BMV}$

TEE Designer Server

DS Key Database
ENTRY ($K_{DS}, ActiveGeo, ActiveDate$)

DS Key Request

Attestation: Verify SGX Platform
The Protocol: DS Key Renewal (On Demand)

Fused Keys: $K_{\text{Seal}}, K_{PV}$
ID: VIN

Signature Keys: $sk_{BMV}, pk_{BMV}$

TEE Designer Server

Derive Session Key $K_{RA}$

Attestation: Verify SGX Platform

DS Key Database
ENTRY ($K_{DS}$, ActiveGeo, ActiveDate)

DS Key Request
The Protocol: DS Key Renewal (On Demand)

Fused Keys: $K_{\text{Seal}}, K_{\text{PV}}$
ID: VIN

Signature Keys: $sk_{\text{BMV}}, pk_{\text{BMV}}$

TEE Designer Server $K_{PV}$

DS Key Database
ENTRY ($K_{DS}, \text{ActiveGeo, ActiveDate}$)

Attestation: Verify SGX Platform

Derive Session Key $K_{RA}$

{$K_{DS}, \text{VRL, Config}$}$K_{RA}$
The Protocol: DS Key Renewal (On Demand)

**Fused Keys:** $K_{\text{Seal}}, K_{PV}$
**ID:** VIN

**Signature Keys:** $sk_{BMV}, pk_{BMV}$

**TEE Designer Server**

$K_{PV}$

**DS Key Database**
ENTRY ($K_{DS}$, ActiveGeo, ActiveDate)

**DS Key Request**

**Attestation:** Verify SGX Platform

**Derive Session Key $K_{RA}$**

$\{K_{DS}, VRL, \text{Config}\}_{K_{RA}}$

$TS \leftarrow \text{TrustedTime()}$

$Loc \leftarrow \text{TrustedLoc()}$

$\text{Seal} \left( K_{DS}, TS, Loc, VRL, \text{Config} )$
The Protocol: V2V (On Demand)
The Protocol: V2V (On Demand)

Vehicle V1

Derive TID ← HKDF(VRK, RIID)

Vehicle V2
The Protocol: V2V (On Demand)

Vehicle V1

Derive $TID \leftarrow HKDF(VRK, RIID)$

$\{TS, loc, TID, BSM\}_K_{DS, CMAC}$

Vehicle V2
The Protocol: V2V (On Demand)

Vehicle V1

Derive $\text{TID} \leftarrow \text{HKDF}(\text{VRK}, \text{RIID})$

Vehicle V2

$\{\text{TS, loc, TID, BSM}\}_K_{DS, \text{CMAC}}$

Verify CMAC
Decrypt TS, loc, TID, BSM
Verify TS with error tolerance
Verify Loc with error tolerance
Check $\text{TID} =? \text{HKDF}(\text{VRK}_{\text{VRL}}, \text{RIID})$
Process BSM
Preliminary Results: TEE vs. PKI

(a) Key Exchange Latency

(b) BSM Broadcast Latency

(c) End To End Latency
Next Step: Formal Verification of the TEE-based V2V Protocol

Modeling New Threats from TEEs

► New TCB modules in TEEs:
  1. Enclave memory (local/global variables)
  2. Non-volatile memory (monotonic counters)
  3. Persistent storage (sealed data)

► New threat model in TEEs:
  1. Controls the privileged code (e.g., OS, or hypervisors)
  2. Arbitrary thread and process instantiation
  3. Permute, reorder enclave calls
  4. Access to ecall/ocall arguments and returns
  5. Replay, modify of data in untrusted code
**Next Step: Formal Verification of the TEE-based V2V Protocol**

### Modeling New Threats from TEEs

**New TCB modules in TEEs:**
1. Enclave memory (local/global variables)
2. Non-volatile memory (monotonic counters)
3. Persistent storage (sealed data)

**New threat model in TEEs:**
1. Controls the privileged code (e.g., OS, or hypervisors)
2. Arbitrary thread and process instantiation
3. Permute, reorder enclave calls
4. Access to ecall/ocall arguments and returns
5. Replay, modify of data in untrusted code

---

“Towards **Formal Verification** of State Continuity for **Enclave Programs**”. USENIX Security 2021
Next Step: Formal Verification of the TEE-based V2V Protocol

Verifying SGX-V2V Protocol (Tamarin)

1. Absence of state continuity or reentrancy vulnerability
2. Absence of non-repudiation attacks (identity of a vehicle cannot be forged)
3. Absence of replay attacks against key distribution
4. Absence of replay attacks against V2V messages
5. Consensus on the same secret key for all the members
Key Intuition: Cryptographic Protocols and SGX Environment Share Similarities
Key Intuition: Cryptographic Protocols and SGX Environment Share Similarities

Key Exchange Protocols

Tamarin MSR and query language

SGX Environment

State Cont. Properties

SGX Thread Model
Komodo [FBHP17]: An enclave implementation using ARM’s TrustZone. It is implemented using software with formal verifications to provide support for feature evolving.

- Comparing to ours: Komodo has its own spec, and it does not have the binary compatibility issue with legacy SGX programs. There are also additional challenges due to the difference between ARM-TrustZone (already enclave-like) and SEV.

OpenSGX

- OpenSGX [JDKH16] is a QEMU implementation of SGX spec without any security promise. It has no security features and not strictly compatible with Intel SGX.
- It is a project when Intel has not released its SDKs, and useful for SGX programs understanding, debugging.
Exploring a framework for **unifying TEEs in virtual environments** [VMw]

“The scheme is designed with **virtualization** in mind and offers capabilities that simplify the use of TEEs for guest VM environments.”

Our project is **inspired** and closely aligns with this vision, but with an emphasis on **binary compatibility**
vSGX: Virtualizing SGX Enclaves on AMD SEV [S&P22]
vSGX: Virtualizing SGX Enclaves on AMD SEV [S&P22]

<table>
<thead>
<tr>
<th>Motivations</th>
<th>Background</th>
<th>Virtualized TEE</th>
<th>TEE-based V2V Protocol</th>
<th>Related Work</th>
<th>Takeaway</th>
<th>References</th>
</tr>
</thead>
<tbody>
<tr>
<td>Binary compatibility</td>
<td>Corresponding security guarantee with both SGX and SEV</td>
<td>Reasonable performance</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

vSGX works like an SGX module plugged onto an SEV machine
vSGX: Virtualizing SGX Enclaves on AMD SEV [S&P22]

- Binary compatibility
- Comparable security guarantee with both SGX and SEV
- Reasonable performance

vSGX works like an SGX module plugged onto an SEV machine

The source code has been made available at https://github.com/OSUSecLab/vSGX
A TEE-based V2V Protocol [AutoSec’22]

Objectives

1. **(Instant) Authentication**
2. Message Integrity (Tamper Resiliency)
3. Message Confidenality (Privacy)
4. **Anti-tracking** (Privacy)
5. Revocation and **Accountability**
6. Scalability
Objectives

1. **(Instant) Authentication**
2. Message Integrity (Tamper Resiliency)
3. Message Confidentiality (Privacy)
4. **Anti-tracking** (Privacy)
5. Revocation and **Accountability**
6. Scalability
A TEE-based V2V Protocol [AutoSec’22]

Our Approach

▶ Leveraging TEEs to protect secrets
▶ Using a shared Daily Symmetric (DS) Key among vehicles
▶ Storing DS key in the enclave
▶ Using Remote Attestation to ensure the right environment
▶ Generating temporal IDs for privacy

Vehicle V1

Derive $\text{TID} \leftarrow \text{HKDF}(	ext{VRK}, \text{RIID})$

Vehicle V2

$\{\text{TS, loc, TID, BSM}\}_{\text{KDS, CMAC}}$

Verify CMAC

Decrypt TS, loc, TID, BSM

Verify TS with error tolerance

Verify Loc with error tolerance

Check $\text{TID} =? \text{HKDF}(	ext{VRK}_{\text{VRL}}, \text{RIID})$

Process BSM
A TEE-based V2V Protocol [AutoSec’22]

Our Approach

▶ Leveraging TEEs to protect secrets
▶ Using a shared Daily Symmetric (DS) Key among vehicles
▶ Storing DS key in the enclave
▶ Using Remote Attestation to ensure the right environment
▶ Generating temporal IDs for privacy

https://github.com/OSUSecLab/tee-v2v
Towards Distributed and Virtualized Trusted Execution Environments

Zhiqiang Lin
Distinguished Professor of Engineering
zlin@cse.ohio-state.edu

12/06/2022
References I


References II


