StrongBox: A GPU TEE on Arm Endpoints

Yunjie Deng*, Chenxu Wang*, Shunchang Yu, Shiqing Liu, Zhenyu Ning, Kevin Leach, Jin Li, Shoumeng Yan, Zhengyu He, Jiannong Cao, Fengwei Zhang
Wide Application of GPU

Virtual Reality
Video Processing
Neural Network
3D Games

Server

Endpoint

Video Processing
Neural Network
3D Games

Server Endpoint 
Virtual
Reality
Neural 
Network
3D 
Games
Video 
 Processing
GPU Security

- Varied **sensitive data** are processed on GPU
  - face, fingerprints, voice ...
- The vulnerable host OS severely threats GPU computing
  - Privileged attackers can directly access the data, or
  - Break the page table isolation between GPU computation

![Diagram of GPU Security](image)
Trusted Execution Environments

- Processor IP developers introduce hardware-assisted Trusted Execution Environment (TEE) for secure data storage and computation
  - Arm TrustZone
  - Intel Software Guard Extensions (SGX)
  - AMD Secure Encrypted Virtualization (SEV)
GPU TEEs

- Secure data transmission between OS and GPU
- Isolate GPU memory and GPU computation
GPU Trusted Execution Environments

- TEEs have participated in secure GPU computing
  - **Graviton**: Trusted Execution Environments on GPUs (OSDI’18)
  - **HIX**: Heterogeneous isolated execution for commodity gpus (ASPLOS’19)
  - **HETEE**: Enabling Rack-scale Confidential Computing using Heterogeneous Trusted Execution Environment (S&P’20)
  - **LITE**: A Low-Cost Practical Inter-Operable GPU TEE (ICS’22)
  - **Secdeep** (IoTDI’21): Secure and Performant On-device Deep Learning Inference Framework for Mobile and IoT Devices
  - ...
Challenges of Adapting Existing Works to Arm Endpoints

- **Architecture**
  - CPU Architecture: Intel vs. Arm
  - GPU Architecture: Dedicated-memory GPU vs. Shared-memory GPU

![Diagram of CPU and GPU architectures]
Challenges of Adapting Existing Works to Arm Endpoints

- Architecture
  - A typical workflow on Arm endpoint GPUs

![Workflow Diagram]

1. Allocate Memory
2. Load Data
3. Load Code
4. Task Submit
5. GPU Execution
6. Task Finish
7. Load Result
Challenges of Adapting Existing Works to Arm Endpoints

- **Compatibility**
  - Hardware modification on GPU chips or system architecture
- **TCB size**
  - Directly porting the vulnerable GPU software stacks into enclave

---

**Diagram:**
- Extra Hardware
- Enclave
- GPU
- Low Compatibility
- Large TCB

---

Yunjie et al. StrongBox
StrongBox Overview

- Architecture
  - Arm hardware features
    - TrustZone Address Space Controller (TZASC)
    - Stage-2 translation
  - Shared-memory GPU
    - Reserve a memory region for sensitive GPU tasks
    - Protect GPU memory by TZASC and Stage-2 translation
StrongBox Overview: Threat Model and Assumptions

- Compromised GPU software stacks
  - GPU runtime
  - GPU driver
  - Other peripheral drivers
  - System OS

- No hypervisor on Arm endpoints

*Trusted secure OS and applications

Out of scope: side-channel attacks, physical attacks, Denial-of-Service

*: Addressed in future works
StrongBox Overview

- **High Compatibility**
  - No hardware modification

- **Minimal TCB**
  - Reuse GPU software to fulfill functionality
  - Deploy lightweight StrongBox runtime to perform security check for sensitive computation tasks

![StrongBox Diagram]

- Host OS
- GPU Software
- smc
- TEE
- StrongBox Runtime
- GPU

Yunjie et al. StrongBox
StrongBox Overview

Software

Normal World
- GPU Application

Secure World
- Secure App

Hardware

CPU
- App
- NS Task
- S Task

GPU
- MMU
- MMIO

Secure Monitor
- Strongbox Runtime
- GPU Guard
- Task Protector
- Trust Modules

Host OS
- GPU Driver
- Memory Manager
- DMA Controller

SEL0
- CPU
- MMU
- TZASC

SEL1
- Normal World

SEL2
- Secure World

SEL3
- Added/Modified
- Trusted
- Untrusted

Added/Modified
- GPU Access

Secure App
- Trusted

Secure OS
- Normal RAM

SEL0
- CPU Access
- GPU Access

SEL1
- CPU Access

SEL2
- CPU Access

SEL3
- CPU Access
StrongBox Overview: Secure Tasks and Non-secure Tasks
Design Details

- Isolated Execution Environment
  - Prohibit the attackers access GPU and GPU memory when executing sensitive tasks
- Dynamic and fine-grained GPU memory access control
  - Prohibit the attackers access scattered sensitive data and code
- Reduce performance overhead
  - Optimize the protection overhead on multi-tasks GPU applications
Isolated Execution Environment

- **Restrict two modes of data access**
  - Host OS to GPU
  - Host OS to shared memory

- **Approach**
  - Route the control from GPU driver to StrongBox runtime inside TrustZone
  - Manage the access to the shared memory

- **Other requirements**
  - Small TCB
  - No hardware modification
Isolated Execution Environment: Submission

- ①: Route control to StrongBox runtime
- ②: Forbid the Host OS to access GPU
- ③: Protect the sensitive data and code
- ④: Submit computation task to GPU
Isolated Execution Environment: Termination

- ⑤: Capture task finish interrupt
- ⑥: Restore the access permission to sensitive data and code
- ⑦: Allow Host OS to access GPU
- ⑧: Route the control to GPU driver
Dynamic and Fine-grained Memory Access Control

- Dynamic access control
  - Apply the protection to different GPU memory content
- Fine-grained protection
  - Combine Stage-2 translation (page-grained) and TZASC (slot-grained)
  - Prohibit the attackers access scattered sensitive data and code
  - Allow the GPU driver access the remaining non-sensitive region to fulfill functionality
Example of Memory Access Control

Application Preparation

<table>
<thead>
<tr>
<th>GPU Page Table Region</th>
<th>Task Region</th>
</tr>
</thead>
<tbody>
<tr>
<td>Buf. 1</td>
<td>Buf. 1</td>
</tr>
<tr>
<td>Buf. 2</td>
<td>Buf. 2</td>
</tr>
<tr>
<td>C1</td>
<td>C1</td>
</tr>
<tr>
<td>Buf. 3</td>
<td>Buf. 3</td>
</tr>
<tr>
<td>Buf. 4</td>
<td>Buf. 4</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>PTEs</th>
<th>Ciphertext</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ciphertext</td>
<td>Code</td>
</tr>
<tr>
<td>Ciphertext</td>
<td>Ciphertext</td>
</tr>
<tr>
<td>Code</td>
<td>Code</td>
</tr>
</tbody>
</table>

Full Accessible  
Write Protected  
DMA Prohibited  
OS-DMA Prohibited  
GPU-DMA Prohibited  
OS-GPU-DMA Prohibited

Task 1

- IN
- Code
- OUT

Buf. 1  
Buf. 2  
C1  
Buf. 2

Task 2

- IN
- Code
- OUT

Buf. 2  
Buf. 3  
C2  
Buf. 4
Example of Memory Access Control

Application Preparation

Task Region

Task 1 Execution

Full Accessible
Write Protected
DMA Prohibited
OS-DMA Prohibited
GPU-DMA Prohibited
OS-GPU-DMA Prohibited
Example of Memory Access Control

Application Preparation

GPU Page Table Region

<table>
<thead>
<tr>
<th>PTEs</th>
<th>Ciphertext</th>
<th>Ciphertext</th>
<th>Code</th>
<th>Ciphertext</th>
<th>Ciphertext</th>
<th>Code</th>
</tr>
</thead>
</table>

Task Region

<table>
<thead>
<tr>
<th>Buf. 1</th>
<th>Buf. 2</th>
<th>C1</th>
<th>Buf. 3</th>
<th>Buf. 4</th>
<th>C2</th>
</tr>
</thead>
<tbody>
<tr>
<td>PTEs</td>
<td>Plaintext</td>
<td>Plaintext</td>
<td>Code</td>
<td>Ciphertext</td>
<td>Ciphertext</td>
</tr>
</tbody>
</table>

Task 1 Execution

<table>
<thead>
<tr>
<th>PTEs</th>
<th>Plaintext</th>
<th>Plaintext</th>
<th>Code</th>
<th>Ciphertext</th>
<th>Ciphertext</th>
<th>Code</th>
</tr>
</thead>
</table>

Task 1 Switch Out

<table>
<thead>
<tr>
<th>PTEs</th>
<th>Ciphertext</th>
<th>Plaintext</th>
<th>Code</th>
<th>Ciphertext</th>
<th>Ciphertext</th>
<th>Code</th>
</tr>
</thead>
</table>

Full Accessible

Write Protected

DMA Prohibited

OS-DMA Prohibited

GPU-DMA Prohibited

OS-GPU-DMA Prohibited
Example of Memory Access Control

Application Preparation

<table>
<thead>
<tr>
<th>GPU Page Table Region</th>
<th>Task Region</th>
</tr>
</thead>
<tbody>
<tr>
<td>Buf. 1</td>
<td>Buf. 2</td>
</tr>
<tr>
<td>PTEs</td>
<td>Ciphertext</td>
</tr>
</tbody>
</table>

Task 1 Execution

<table>
<thead>
<tr>
<th>PTEs</th>
<th>Plaintext</th>
<th>Plaintext</th>
<th>Code</th>
<th>Ciphertext</th>
<th>Ciphertext</th>
<th>Code</th>
</tr>
</thead>
</table>

Task 1 Switch Out

<table>
<thead>
<tr>
<th>PTEs</th>
<th>Ciphertext</th>
<th>Plaintext</th>
<th>Code</th>
<th>Ciphertext</th>
<th>Ciphertext</th>
<th>Code</th>
</tr>
</thead>
</table>

Task 2 Execution

<table>
<thead>
<tr>
<th>PTEs</th>
<th>Ciphertext</th>
<th>Plaintext</th>
<th>Code</th>
<th>Plaintext</th>
<th>Plaintext</th>
<th>Code</th>
</tr>
</thead>
</table>

- Full Accessible
- Write Protected
- DMA Prohibited
- OS-DMA Prohibited
- GPU-DMA Prohibited
- OS-GPU-DMA Prohibited

IN

OUT

Task 1

Task 2

Yunjie et al. StrongBox 23 / 31
## Example of Memory Access Control

<table>
<thead>
<tr>
<th>Application Preparation</th>
<th>Task Region</th>
</tr>
</thead>
<tbody>
<tr>
<td>Buf. 1</td>
<td>Buf. 2</td>
</tr>
<tr>
<td>PTEs</td>
<td>Ciphertext</td>
</tr>
</tbody>
</table>

### Task 1 Execution

- **Buf. 1**: Plaintext
- **Buf. 2**: Plaintext
- **C1**: Code
- **Buf. 3**: Ciphertext
- **Buf. 4**: Ciphertext
- **C2**: Code

### Task 1 Switch Out

- **Buf. 1**: Ciphertext
- **Buf. 2**: Plaintext
- **C1**: Code
- **Buf. 3**: Ciphertext
- **Buf. 4**: Ciphertext
- **C2**: Code

### Task 2 Execution

- **Buf. 1**: Ciphertext
- **Buf. 2**: Plaintext
- **C1**: Code
- **Buf. 3**: Plaintext
- **Buf. 4**: Plaintext
- **C2**: Code

### Task 2 Switch Out

- **Buf. 1**: Ciphertext
- **Buf. 2**: Ciphertext
- **C1**: Code
- **Buf. 3**: Ciphertext
- **Buf. 4**: Ciphertext
- **C2**: Code

**Legend:**
- Full Accessible
- Write Protected
- DMA Prohibited
- OS-DMA Prohibited
- GPU-DMA Prohibited
- OS-GPU-DMA Prohibited
### Example of Memory Access Control

<table>
<thead>
<tr>
<th>GPU Page Table Region</th>
<th>Task Region</th>
</tr>
</thead>
<tbody>
<tr>
<td>Buf. 1</td>
<td>Buf. 2</td>
</tr>
<tr>
<td>PTEs</td>
<td>Ciphertext</td>
</tr>
</tbody>
</table>

- **Application Preparation**
  - Task 1
    - Execution
    - Switch Out
  - Task 2
    - Execution
    - Switch Out

- **All Tasks Finished**
  - PTEs
  - Ciphertext
  - Ciphertext
  - Code
  - Ciphertext
  - Ciphertext
  - Code

- **Access Control**
  - Full Accessible
  - Write Protected
  - DMA Prohibited
  - OS-DMA Prohibited
  - GPU-DMA Prohibited
  - OS-GPU-DMA Prohibited

- **Task 1**
  - IN
  - Buf. 1
  - Buf. 2
  - C1
  - OUT
  - Buf. 2

- **Task 2**
  - IN
  - Buf. 2
  - Buf. 3
  - C2
  - OUT
  - Buf. 4
Reduce Performance Overhead

- In multi-task applications, the output of one task can be used as the input of the next task.
- Eliminate redundant operations to reduce performance overhead.

![Diagram showing redundant operation optimization policy](image-url)
Evaluation: Security Analysis

- ①: Directly access the sensitive data and code ×
- ②: Attack with malicious tasks ×
- ③: Attack with fake GPU ×
- ④: Attack with compromised GPU software ×
Evaluation: Rodinia Benchmark

Figure: Evaluation on Rodinia benchmarks (overhead 4.70% - 15.26%).
### Optimization on redundant protection

<table>
<thead>
<tr>
<th>Benchmark</th>
<th>Task</th>
<th>No Optimization</th>
<th>StrongBox</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>TProtect</td>
<td>Total</td>
</tr>
<tr>
<td>Single</td>
<td>KNN</td>
<td>7.31 (11.55%)</td>
<td>63.30</td>
</tr>
<tr>
<td>Task</td>
<td>LMD</td>
<td>1,227.88 (8.27%)</td>
<td>14,854.08</td>
</tr>
<tr>
<td>Multi</td>
<td>PF</td>
<td>3,495.99 (54.50%)</td>
<td>6,414.31</td>
</tr>
<tr>
<td>Task</td>
<td>LUD</td>
<td>97,179.42 (95.24%)</td>
<td>102,032.57</td>
</tr>
<tr>
<td></td>
<td>H3D</td>
<td>196,457.42 (96.87%)</td>
<td>202,797.82</td>
</tr>
<tr>
<td></td>
<td>GS</td>
<td>2,149,460.48 (97.40%)</td>
<td>2,206,881.00</td>
</tr>
</tbody>
</table>
Conclusion on StrongBox

- First GPU TEE on Arm Endpoints
  - Ensure secure and isolated computation on Arm endpoint GPUs
  - Entail a minimal TCB to reduce potential attack surface
  - Maintain high compatibility

- Source code
  - https://github.com/Compass-All/CCS22-StrongBox
Thank You