SGXELIDE: Enabling Enclave Code Secrecy via Self-Modification

Erick Bauman\textsuperscript{1}, Huibo Wang\textsuperscript{1}, Mingwei Zhang\textsuperscript{2}, Zhiqiang Lin\textsuperscript{1,3}

\textsuperscript{1}University of Texas at Dallas
\textsuperscript{2}Intel Labs
\textsuperscript{3}The Ohio State University

CGO 2018
Intel SGX

Skylake
Intel SGX

- Provides secure *enclaves*
Intel SGX

- Provides secure *enclaves*
- Memory regions isolated from all other code
Intel SGX

- Provides secure *enclaves*
- Memory regions isolated from all other code
- Cannot be accessed by OS or hypervisor
Intel SGX

Diagram:

- **Operating System**
- **Hypervisor**
- **Hardware**
- **Trusted**

- **App**
- **App**
- **App**

[Diagram showing layers of the Intel SGX trusted execution environment]
Intel SGX

Client
Application

Disk
Enclave
Code
Data
Intel SGX
Intel SGX

Client

Application

Enclave

Code Data

Disk

Enclave

Code Data

Attest
Intel SGX

Data Integrity

Client

Application

Enclave

Code

Data

Disk

Enclave

Code

Data

Attest
Intel SGX

Client
Enclave
Application
Disk

Data Integrity
Code Integrity
Intel SGX

- **Data Integrity**
- **Code Integrity**

Client

**Application**

**Enclave**

- Code
- Data

**Disk**

**Enclave**

- Code
- Data

Attest

Secret Data

- Disc

- Client

- Enclave

- Code

- Data
Intel SGX

- Data Integrity
- Code Integrity
- Data Confidentiality
Intel SGX

Data Integrity
Code Integrity
Data Confidentiality
Intel SGX

- Data Integrity
- Code Integrity
- Data Confidentiality
- Code Confidentiality
“The enclave file can be disassembled, so the algorithms used by the enclave developer will not remain secret.”
–SGX SDK Manual
SGXELIDE

Definition

Elide: To leave out or omit
Challenges

- Enclaves must be signed and unmodified until initialization
Challenges

- Enclaves must be signed and unmodified until initialization
- The entire enclave cannot be encrypted
Challenges

- Enclaves must be signed and unmodified until initialization
- The entire enclave cannot be encrypted
- Any secrets cannot be stored in the enclave
Challenges

- Enclaves must be signed and unmodified until initialization
- The entire enclave cannot be encrypted
- Any secrets cannot be stored in the enclave
- There should be minimal toolchain changes
Main Idea

Redact (or sanitize) secrets and restore at runtime
Blacklist vs. Whitelist

Blacklist

- User specifies secrets (e.g. annotations)
- Minimizes code that must be encrypted
- Burden of annotating secrets on developer

Whitelist

- Only specify code that must not be redacted
- Applicable to any enclave
- No need for developer to mark secrets
- More code must be encrypted
Blacklist vs. Whitelist

**Blacklist**
- User specifies secrets (e.g. annotations)
Blacklist vs. Whitelist

**Blacklist**
- User specifies secrets (e.g. annotations)
- Minimizes code that must be encrypted
Blacklist vs. Whitelist

**Blacklist**

- User specifies secrets (e.g. annotations)
- Minimizes code that must be encrypted
- Burden of annotating secrets on developer

**Whitelist**

- Only specify code that must not be redacted
- Applicable to any enclave
- No need for developer to mark secrets
- More code must be encrypted
Blacklist vs. Whitelist

**Blacklist**
- User specifies secrets (e.g. annotations)
- Minimizes code that must be encrypted
- Burden of annotating secrets on developer
- Risk of mistakes

**Whitelist**
- Only specify code that must not be redacted
- Applicable to any enclave
- No need for developer to mark secrets
- More code must be encrypted
Blacklist vs. Whitelist

**Blacklist**
- User specifies secrets (e.g. annotations)
- Minimizes code that must be encrypted
- Burden of annotating secrets on developer
- Risk of mistakes

**Whitelist**
- Only specify code that must not be redacted
- Applicable to any enclave
- No need for developer to mark secrets
- More code must be encrypted
Blacklist vs. Whitelist

**Blacklist**
- User specifies secrets (e.g. annotations)
- Minimizes code that must be encrypted
- Burden of annotating secrets on developer
- Risk of mistakes

**Whitelist**
- Only specify code that must not be redacted
Blacklist vs. Whitelist

**Blacklist**

- User specifies secrets (e.g. annotations)
- Minimizes code that must be encrypted
- Burden of annotating secrets on developer
- Risk of mistakes

**Whitelist**

- Only specify code that must not be redacted
- Applicable to any enclave
Blacklist vs. Whitelist

**Blacklist**
- User specifies secrets (e.g. annotations)
- Minimizes code that must be encrypted
- Burden of annotating secrets on developer
- Risk of mistakes

**Whitelist**
- Only specify code that must not be redacted
- Applicable to any enclave
- No need for developer to mark secrets
Blacklist vs. Whitelist

**Blacklist**
- User specifies secrets (e.g. annotations)
- Minimizes code that must be encrypted
- Burden of annotating secrets on developer
- Risk of mistakes

**Whitelist**
- Only specify code that must not be redacted
- Applicable to any enclave
- No need for developer to mark secrets
- More code must be encrypted
Our Solution

- Sign sanitized enclave and restore secrets after initializing
Our Solution

- Sign sanitized enclave and restore secrets after initializing
- Encrypt all nonessential functions
Our Solution

- Sign sanitized enclave and restore secrets after initializing
- Encrypt all nonessential functions
- Use remote attestation
Our Solution

- Sign sanitized enclave and restore secrets after initializing
- Encrypt all nonessential functions
- Use remote attestation
- Use both local and remote storage
SGXELIDE Overview

**Dummy Enclave Code Generation**
- dummy enclave code
- Compiler, Linker
- dummy.so

**Normal Enclave Code Generation**
- secret enclave code
- Compiler, Linker
- secret.so

**Sanitizer**
- secret.so
- secret.data

**Runtime Secret Enclave Code Restoration**
- Enclave
- Runtime Restorer
- secret.so
Remote vs. Local Data

- Secret Data
- Local Data
Remote vs. Local Data
Remote vs. Local Data

- Remote Data: Secret Key
- Local Data: Secret Data
Remote vs. Local Data
Remote vs. Local Data

- Secret Key
- Secret Data
SGXELIDE Design - Remote Data Data
SGXELIDE Design - Remote Data

User Platform

Application

Untrusted Code

Enclave

File System

Authentication Server

meta data

secret data
**SGXELIDE Design - Remote Data**

![Diagram of SGXELIDE Design - Remote Data](image-url)
SGXELIDE Design - Remote Data Data

User Platform

Application

Untrusted Code

Enclave

meta data

File System

Authentication Server

meta data
secret data
**SGXELIDE Design - Remote Data**

- **User Platform**
  - Application
    - Untrusted Code
  - Enclave
    - meta data
- **File System**
- **Authentication Server**
  - meta data
  - secret data

Flow:
1. Untrusted Code
2. Enclave
3. Authentication Server
4. meta data
SGXELIDE Design - Remote Data

User Platform

Application

Untrusted Code

Enclave

secret data

meta data

File System

Authentication Server

meta data

secret data

1. Authentication Server

2. User Platform

3. Application

4. Enclave

5. File System
SGXELIDE Design - Remote Data

**User Platform**

**Application**

**Enclave**

**Authentication Server**

1. Untrusted Code
2. secret data
3. meta data
4. secret data
5. meta data
6. sealed secret data
7. sealed secret data
SGXELIDE Design - Local Data
SGXELIDE Design - Local Data

User Platform

Application

Untrusted Code

Enclave

1

File System

encrypted secret data

Authentication Server

meta data
SGXELIDE Design - Local Data

User Platform

Application

Untrusted Code

Enclave

File System

encrypted secret data

Authentication Server

meta data
SGXELIDE Design - Local Data

User Platform

Application

Untrusted Code

Enclave

meta data

File System

encrypted secret data

Authentication Server

meta data

1

2

3
SGXELIDE Design - Local Data

User Platform

Application

Untrusted Code

1

Enclave

meta
data

2 3

File
System

encrypted
secret data

4

Authentication
Server

meta
data
SGXELIDE Design - Local Data

User Platform

Application

Untrusted Code

Enclave

secret data

meta data

File System

encrypted secret data

Authentication Server

secret data

meta data
SGXELIDE Design - Local Data

User Platform

Application

Untrusted Code

Enclave

secret data

meta data

File System

encrypted secret data

Authentication Server

secret data

meta data
SGXELIDE Design - Local Data
## Benchmarks

<table>
<thead>
<tr>
<th>Benchmarks</th>
<th>Original LOC</th>
<th>LOC w/ SGX</th>
<th>LOC w/ SGXELIDE</th>
<th>TC Functions</th>
<th>TC Bytes</th>
<th>Sanitized Functions</th>
<th>Sanitized Bytes</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>UC</td>
<td>TC</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AES</td>
<td>802</td>
<td>472</td>
<td>427</td>
<td>522</td>
<td>540</td>
<td>185</td>
<td>75999</td>
</tr>
<tr>
<td>DES</td>
<td>473</td>
<td>463</td>
<td>372</td>
<td>513</td>
<td>485</td>
<td>179</td>
<td>75455</td>
</tr>
<tr>
<td>Sha1</td>
<td>315</td>
<td>423</td>
<td>251</td>
<td>473</td>
<td>364</td>
<td>179</td>
<td>73791</td>
</tr>
<tr>
<td>Shas</td>
<td>2417</td>
<td>1529</td>
<td>1240</td>
<td>1579</td>
<td>1353</td>
<td>224</td>
<td>80127</td>
</tr>
<tr>
<td>2048</td>
<td>413</td>
<td>551</td>
<td>192</td>
<td>601</td>
<td>305</td>
<td>208</td>
<td>76351</td>
</tr>
<tr>
<td>Biniax</td>
<td>3523</td>
<td>3582</td>
<td>193</td>
<td>3632</td>
<td>306</td>
<td>208</td>
<td>76351</td>
</tr>
<tr>
<td>Crackme</td>
<td>48</td>
<td>316</td>
<td>93</td>
<td>366</td>
<td>206</td>
<td>182</td>
<td>73711</td>
</tr>
</tbody>
</table>
## Sanitization/Restoration Time

<table>
<thead>
<tr>
<th>Benchmarks</th>
<th>Remote Data</th>
<th></th>
<th>Local Data</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Sanitize</td>
<td>Stand.</td>
<td>Restore</td>
<td>Stand.</td>
<td>Sanitize</td>
</tr>
<tr>
<td>AES</td>
<td>0.09</td>
<td>0.01</td>
<td>4.06</td>
<td>0.54</td>
<td>0.15</td>
</tr>
<tr>
<td>DES</td>
<td>0.09</td>
<td>0.01</td>
<td>3.99</td>
<td>0.52</td>
<td>0.14</td>
</tr>
<tr>
<td>Sha1</td>
<td>0.09</td>
<td>0.01</td>
<td>3.67</td>
<td>0.35</td>
<td>0.14</td>
</tr>
<tr>
<td>Shas</td>
<td>0.09</td>
<td>0.00</td>
<td>4.06</td>
<td>0.53</td>
<td>0.15</td>
</tr>
<tr>
<td>2048</td>
<td>0.09</td>
<td>0.01</td>
<td>3.78</td>
<td>0.52</td>
<td>0.15</td>
</tr>
<tr>
<td>Biniax</td>
<td>0.09</td>
<td>0.00</td>
<td>4.44</td>
<td>0.61</td>
<td>0.15</td>
</tr>
<tr>
<td>Crackme</td>
<td>0.09</td>
<td>0.01</td>
<td>3.53</td>
<td>0.28</td>
<td>0.15</td>
</tr>
</tbody>
</table>
**SGXELIDE Overhead - Remote Data**

The chart above illustrates the relative performance comparison between two methods: one with SGX and another with SGXELIDE. The metric measured is the overhead introduced by these methods for different data sets and cryptographic algorithms.

- **AES**
  - w/ SGX: 100%
  - w/ SGXELIDE: 105%

- **DES**
  - w/ SGX: 101%
  - w/ SGXELIDE: 102%

- **Sha1**
  - w/ SGX: 102%
  - w/ SGXELIDE: 103%

- **Shas**
  - w/ SGX: 103%
  - w/ SGXELIDE: 104%

- **Crackme**
  - w/ SGX: 104%
  - w/ SGXELIDE: 105%

The chart shows a notable increase in overhead with SGXELIDE compared to SGX, especially noticeable in the **Crackme** category.
SGXELIDE Overhead - Local Data

The diagram illustrates the relative performance of various encryption and hashing algorithms with and without the SGXELIDE overhead. The x-axis represents the algorithms: AES, DES, Sha1, Shas, Crackme. The y-axis shows the relative performance percentage.

- AES: 99%
- DES: 100%
- Sha1: 101%
- Shas: 102%
- Crackme: 103%

The chart compares the performance with and without SGXELIDE, showing a slight increase in overhead for all algorithms except AES.
Discussions

SGXELIDE enclaves are self-modifying!
SGXELIDE enclaves are self-modifying!

- How do we defend against malicious enclaves?
Discussions

SGXElide enclaves are self-modifying!

- How do we defend against malicious enclaves?
- How do we protect vulnerable enclaves?
Discussions

**SGXELIDE enclaves are self-modifying!**

- How do we defend against malicious enclaves?
- How do we protect vulnerable enclaves?
- How does this influence side-channel attacks?
Discussions

**SGXELIDE enclaves are self-modifying!**
- How do we defend against malicious enclaves?
- How do we protect vulnerable enclaves?
- How does this influence side-channel attacks?

Limitations and future work
Discussions

**SGXELIDE enclaves are self-modifying!**
- How do we defend against malicious enclaves?
- How do we protect vulnerable enclaves?
- How does this influence side-channel attacks?

**Limitations and future work**
- Framework not completely transparent
Discussions

**SGXELIDE enclaves are self-modifying!**
- How do we defend against malicious enclaves?
- How do we protect vulnerable enclaves?
- How does this influence side-channel attacks?

**Limitations and future work**
- Framework not completely transparent
- Would be useful to test SGXELIDE with large-scale software
Discussions

**SGXELIDE enclaves are self-modifying!**
- How do we defend against malicious enclaves?
- How do we protect vulnerable enclaves?
- How does this influence side-channel attacks?

**Limitations and future work**
- Framework not completely transparent
- Would be useful to test SGXELIDE with large-scale software
- Framework is proof-of-concept and not production ready
Conclusion

**SGXELIDE**

- Presented framework for SGX that ensures code confidentiality
- Sanitize enclave and dynamically restore code at runtime
- Evaluated SGXELIDE’s performance with SGX benchmarks we developed
- Showed SGXELIDE has very little overhead with no performance penalty after restoration
Conclusion

**SGXELIDE**
- Presented framework for SGX that ensures code confidentiality
- Sanitize enclave and dynamically restore code at runtime
- Evaluated SGXELIDE’s performance with SGX benchmarks we developed
- Showed SGXELIDE has very little overhead with no performance penalty after restoration

**SGXELIDE Source**
github.com/utds3lab/sgxelide
Thank You