MesaTEE SGX: Redefining AI and Big Data Analysis with Intel SGX

Yu Ding
Staff Security Scientist, Baidu X-Lab

May-29-2019
About me

- Security Scientist @Baidu X-Lab
- Rust Fans
- Ph.D on Exploit/Mitigation
- Works on Rust-SGX projects

• https://dingelish.com
• https://github.com/dingelish
• https://github.com/baidu/rust-sgx-sdk
MesaTEE SGX
Redefining AI and Big Data Analysis with Intel SGX

Intel SGX for Privacy-Preserving Computation
• Background of Intel SGX
• Challenges on building a privacy-preserving software stack based on Intel SGX

Hybrid Memory Safety
• Rule-of-thumb
• Practice on Intel SGX

Towards a Secure and Trustworthy AI/Big Data Analysis framework
• What is trustworthiness?
• Achieving trustworthy AI/Big Data Analysis using Intel SGX
MesaTEE SGX

Redefining AI and Big Data Analysis with Intel SGX

Intel SGX for Privacy-Preserving Computation
- Background of Intel SGX
- Challenges on building a privacy-preserving software stack based on Intel SGX

Hybrid Memory Safety
- Rule-of-thumb
- Practice on Intel SGX

Towards a Secure and Trustworthy AI/Big Data Analysis framework
- What is trustworthiness?
- Achieving trustworthy AI/Big Data Analysis using Intel SGX
MesaTEE SGX
Redefining AI and Big Data Analysis with Intel SGX
MesaTEE SGX
Redefining AI and Big Data Analysis with Intel SGX
MesaTEE SGX
Redefining AI and Big Data Analysis with Intel SGX

- Cloud Provider
- Data Owner
- Algorithm Provider (can be data owner)

Don’t trust each other

Data leaves its owner but still guaranteed to be under control
MesaTEE SGX
Redefining AI and Big Data Analysis with Intel SGX

• Solution Overview
• Use Intel SGX to establish trust and TEE
  • Secure and Trusted Authentication/Authorization
  • Secure and Trusted Channel
  • Secure and Trusted Execution Environment
• Build system with hybrid memory safety
• Trustworthy AI/Big Data Analysis
MesaTEE SGX
Redefining AI and Big Data Analysis with Intel SGX
MesaTEE SGX
Redefining AI and Big Data Analysis with Intel SGX

Intel SGX for Privacy-Preserving Computation

- Background of Intel SGX
- Challenges on building a privacy-preserving software stack based on Intel SGX

Hybrid Memory Safety

- Rule-of-thumb
- Practice on Intel SGX

Towards a Secure and Trustworthy AI/Big Data Analysis framework

- What is trustworthiness?
- Achieving trustworthy AI/Big Data Analysis using Intel SGX
Background of Intel SGX

Apps not protected from privileged code attacks

Protected Mode (rings) protects OS from apps ...

... and apps from each other ...

... UNTIL a malicious app exploits a flaw to gain full privileges and then tampers with the OS or other apps

Intel® Software Guard Extensions (Intel® SGX)
Frank McKeen, Intel Labs, April 15, 2015
Background of Intel SGX

Attack surface without/with Intel SGX Enclaves

Intel® Software Guard Extensions (Intel® SGX)
Frank McKeen, Intel Labs, April 15, 2015
Background of Intel SGX
Memory access control during address translation

Intel® Software Guard Extensions (Intel® SGX)
Frank McKeen, Intel Labs, April 15, 2015
Background of Intel SGX

Confidentiality and Integrity guarantees

- With its own code and data
- Provide Confidentiality
- Provide integrity
- With controlled entry points
- Supporting multiple threads
- With full access to app memory

Intel® Software Guard Extensions (Intel® SGX)
Frank McKeen, Intel Labs, April 15, 2015
Background of Intel SGX
Measurement and Attestation

Verify the measurement/signer

Establish trust by Remote Attestation
Background of Intel SGX
Remote Attestation

Figure is from “A First Step Towards Leveraging Commodity Trusted Execution Environments for Network Applications”, Seongmin Kim et al.
Background of Intel SGX

Short Summary of Intel SGX

- Provides any application the ability to keep a secret
- Provide capability using new processor instructions
- Application can support multiple enclaves

Provides integrity and confidentiality
- Resists hardware attacks
- Prevent software access, including privileged software and SMM

Applications run within OS environment
- Low learning curve for application developers
- Open to all developers
Background of Intel SGX
Challenges on building a privacy-preserving software stack based on Intel SGX

- Hard Limitations of Intel SGX
  - No syscall
  - No RDTSC
  - No CPUID
  - 128 Mbytes of EPC memory. Slow page-fault driven memory swapping
  - No mprotect
Background of Intel SGX

Challenges on building a privacy-preserving software stack based on Intel SGX

• **Hard Limitations of Intel SGX => Challenges**
  • No syscall
    • No fs/net/env/proc/thread/...
  • No RDTSC
  • No trusted time. How to verify a TLS certificate?
  • No CPUID
    • Some crypto libraries needs it for better performance
  • 128 Mbytes of EPC memory. Slow page-fault driven memory swapping
  • AI? Big data analysis?
  • No mprotect: JIT? AOT?
Background of Intel SGX
Challenges on building a privacy-preserving software stack based on Intel SGX

- Hard Limitations of Intel SGX
  - No syscall
  - No fs/net/env/proc/thread/…
  - No RDTSC
  - No trusted time. How to verify a TLS certificate?
  - No CPUID
  - Some crypto libraries need it for better performance
  - 128 Mbytes of EPC memory. Slow page-fault driven memory
  - AI? Big data analysis?
  - No mprotect: JIT? AOT?
Background of Intel SGX

Challenges on building a privacy-preserving software stack based on Intel SGX

- Soft Limitations of Intel SGX
  - Suffers from memory bugs

- Memory Safety?
  - Overflow?
  - UAF?
  - Data Racing?
  - ROP?
Background of Intel SGX
Challenges on building a privacy-preserving software stack based on Intel SGX

- Soft Limitations of Intel SGX
  - Suffers from memory bugs

- Memory Safety?
  - Overflow?
  - UAF?
  - Data Racing?
  - ROP?
Background of Intel SGX
Challenges on building a privacy-preserving software stack based on Intel SGX

• Short Summary
• Challenges
  • Re-implement a software stack in Intel SGX environment on a limited foundation
  • Require memory safety guarantees
MesaTEE SGX
Redefining AI and Big Data Analysis with Intel SGX

Intel SGX for **Privacy-Preserving** Computation
- Background of Intel SGX
- Challenges on building a privacy-preserving software stack based on Intel SGX

**Hybrid Memory Safety**
- Rule-of-thumb
- Practice on Intel SGX

Towards a **Secure and Trustworthy** AI/Big Data Analysis framework
- What is trustworthiness?
- Achieving trustworthy AI/Big Data Analysis using Intel SGX
Hybrid Memory Safety
Programming Languages Guarantee Memory Safety
Hybrid Memory Safety
The Software Stack

• Kernel
• Syscall
• Libc, system libs
• Runtime libs
• Applications
Hybrid Memory Safety

The Software Stack

- Kernel
- Syscall
- Libc, system libs
- Runtime libs
- Applications
Hybrid Memory Safety

Hybrid Memory Safety – Rule-of-thumb

- Unsafe components must not taint safe components, especially for public APIs and data structures.
- Unsafe components should be as small as possible and decoupled from safe components.
- Unsafe components should be explicitly marked during deployment and ready to upgrade.
Hybrid Memory Safety

Hybrid Memory Safety – MesaPy as an Example

![Diagram showing Hybrid Memory Safety and MesaPy as an Example]
## Hybrid Memory Safety

Hybrid Memory Safety – Practice in SGX

<table>
<thead>
<tr>
<th></th>
<th>Linux</th>
<th>Rust-SGX</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kernel</td>
<td></td>
<td>N/A</td>
</tr>
<tr>
<td>Syscall</td>
<td></td>
<td>OCALL (statically controlled)</td>
</tr>
<tr>
<td>Libc</td>
<td></td>
<td>Intel – SGX tlibc</td>
</tr>
<tr>
<td>Runtime</td>
<td></td>
<td>Rust-SGX sgx_tstd/...</td>
</tr>
</tbody>
</table>
Hybrid Memory Safety

Hybrid Memory Safety – Practice in SGX

- Rusty-machine
- Remote attestation
- ring/rustls/webpki
- sgx_tstd
- sgx_tlibc
- sgx_trts

- gdbt-rs
- Data storage/trans
- crypto_helper
- sgx_trts
- sgx_tcrypto
- sgx_tservices

- tvm worker
- Interpreter
tvm-runtime
sgx_tservices

Enclave Boundary
Hybrid Memory Safety

Hybrid Memory Safety – Practice in SGX

#![no_std]

libpanic_abort → libunwind → librustc_demangle → compiler_builtins → libstd → liballoc → libc → glibc

#![no_core]

libcore
Hybrid Memory Safety
Hybrid Memory Safety – Practice in SGX
MesaTEE SGX

Redefining AI and Big Data Analysis with Intel SGX

Intel SGX for Privacy-Preserving Computation
- Background of Intel SGX
- Challenges on building a privacy-preserving software stack based on Intel SGX

Hybrid Memory Safety
- Rule-of-thumb
- Practice on Intel SGX

Towards a Secure and Trustworthy AI/Big Data Analysis framework
- What is trustworthiness?
- Achieving trustworthy AI/Big Data Analysis using Intel SGX
Towards a Secure and Trustworthy AI/Big Data Analysis framework

What is trustworthiness?
What is trustworthiness?
What is trustworthiness?

The term **Trustworthy Computing** (TwC) has been applied to computing systems that are inherently secure, available, and reliable. It is particularly associated with the **Microsoft** initiative of the same name, launched in 2002.
Towards a Secure and Trustworthy AI/Big Data Analysis framework

What is trustworthiness?

Trusted computing

The term is taken from the field of trusted systems and has a specialized meaning. With Trusted Computing, the computer will consistently behave in *expected* ways, and those behaviors will be enforced by computer hardware and software.
Towards a Secure and Trustworthy AI/Big Data Analysis framework

Achieving trustworthy AI/Big Data Analysis using Intel SGX

Gradient-Boosting decision tree

How to achieve trustworthy?

• The running instance started with the static binary I wanted to run
• The static binary is generated from the codes I want to use
• The code I use implements the algorithm honestly
• The compiler is not doing evil
• Data transfer is secure
Towards a Secure and Trustworthy AI/Big Data Analysis framework

Achieving trustworthy AI/Big Data Analysis using Intel SGX

Gradient-Boosting decision tree

**gbdt-rs**

- ~2000 sloc of Rust – Self explain
- Well commented/documented
- 7x faster than XGBoost on 1 thread
- Works seamlessly in SGX
- Clean and clear software stack!

![Memory Usage and Training Time Graphs](image-url)
Towards a Secure and Trustworthy AI/Big Data Analysis framework

Achieving trustworthy AI/Big Data Analysis using Intel SGX

MesaPy SGX

- Ported PyPy with strong bound check
- Disabled all syscalls
- Customized runtime – limited ocall
- Eliminate indeterminism
- Formal verification
- Replace unsafe libraries with Rust crates
Towards a Secure and Trustworthy AI/Big Data Analysis framework

Achieving trustworthy AI/Big Data Analysis using Intel SGX

<table>
<thead>
<tr>
<th>Solutions</th>
<th>Spark</th>
<th>MesaTEE Spark</th>
<th>GraphSC</th>
<th>OblIVM</th>
<th>Homomorphic Encryption</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Encryption</td>
<td>x</td>
<td>✓</td>
<td>x</td>
<td>x</td>
<td>✓</td>
</tr>
<tr>
<td>Oblivious</td>
<td>x</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>x</td>
</tr>
<tr>
<td>Turnaround</td>
<td>1 sec</td>
<td>4-20 sec</td>
<td>2-6 days</td>
<td>&gt;100 days</td>
<td>∞</td>
</tr>
</tbody>
</table>
Towards a Secure and Trustworthy AI/Big Data Analysis framework
Achieving trustworthy AI/Big Data Analysis using Intel SGX

We are working with Baidu XuperData for applications
Towards a Secure and Trustworthy AI/Big Data Analysis framework

Anakin-SGX

---

![Graph showing comparison between X86-64 and SGX for NIN_ImageNet (1000 images)]
MesaTEE SGX: Redefining AI and Big Data Analysis with Intel SGX

Yu Ding
Security Scientist, Baidu X-Lab

Q&A