BLUEHAT
SHANGHAI 2019

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

Yu Ding

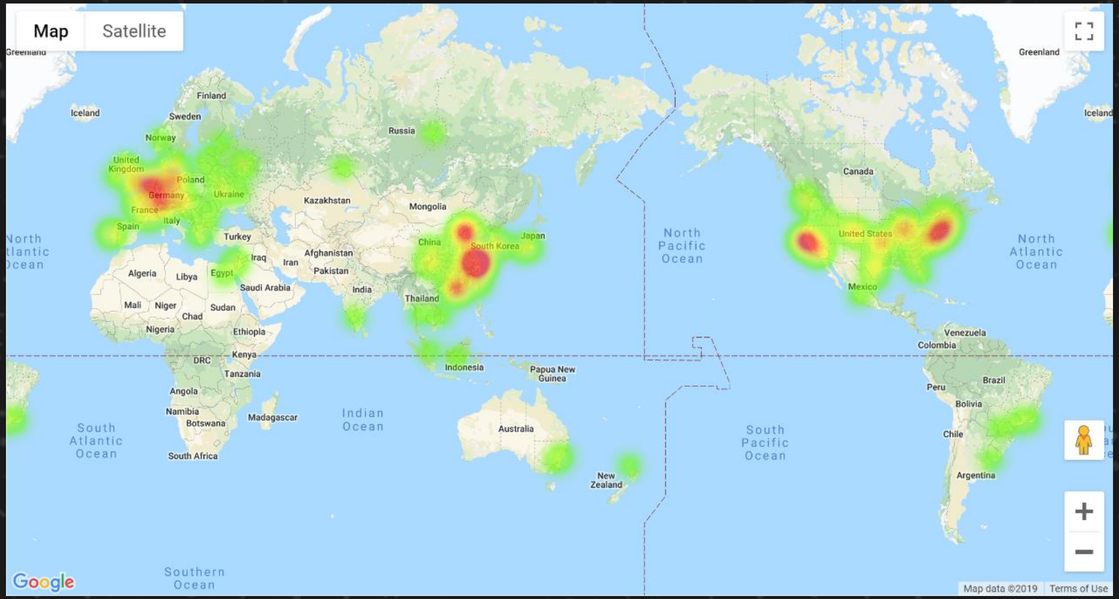
Staff Security Scientist, Baidu X-Lab

May-29-2019

About me

- https://dingelish.com
- https://github.com/dingelish
- https://github.com/baidu/rust-sgx-sdk

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



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

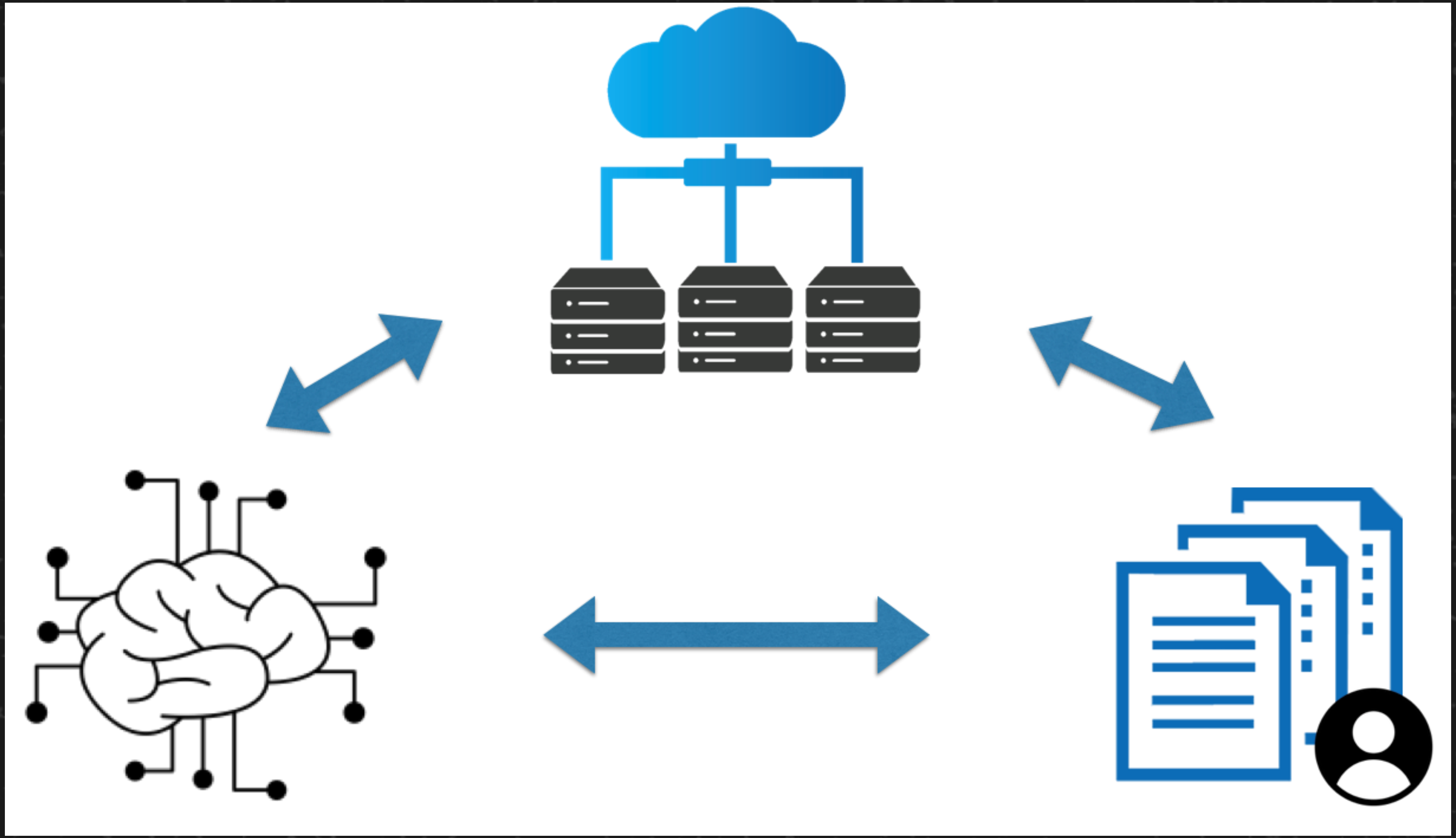
- Rule-of-thumb
- Practice on Intel SGX

Towards a Secure and Trustworthy AI/Big Data Analysis framework

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

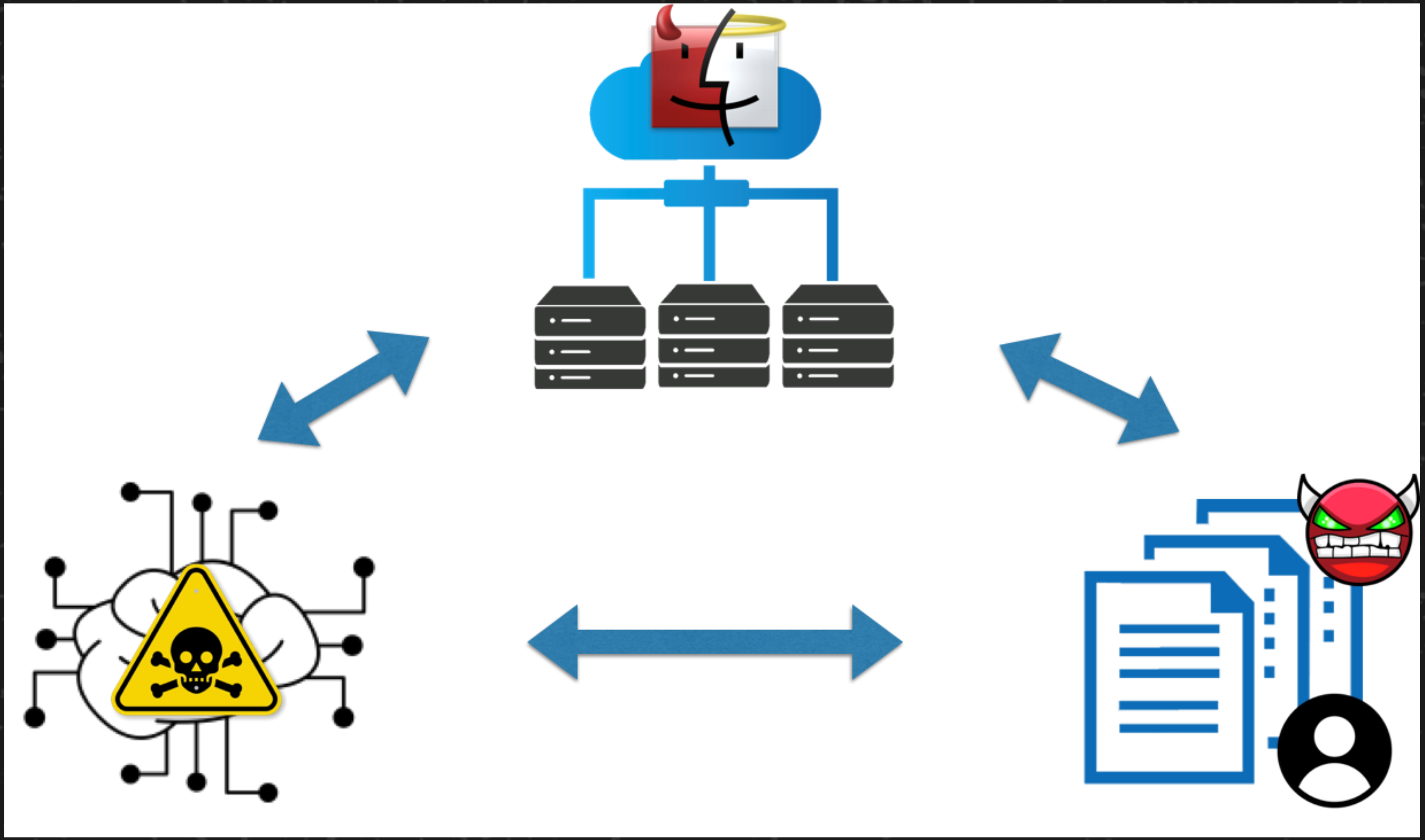
MesaTEE SGX

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

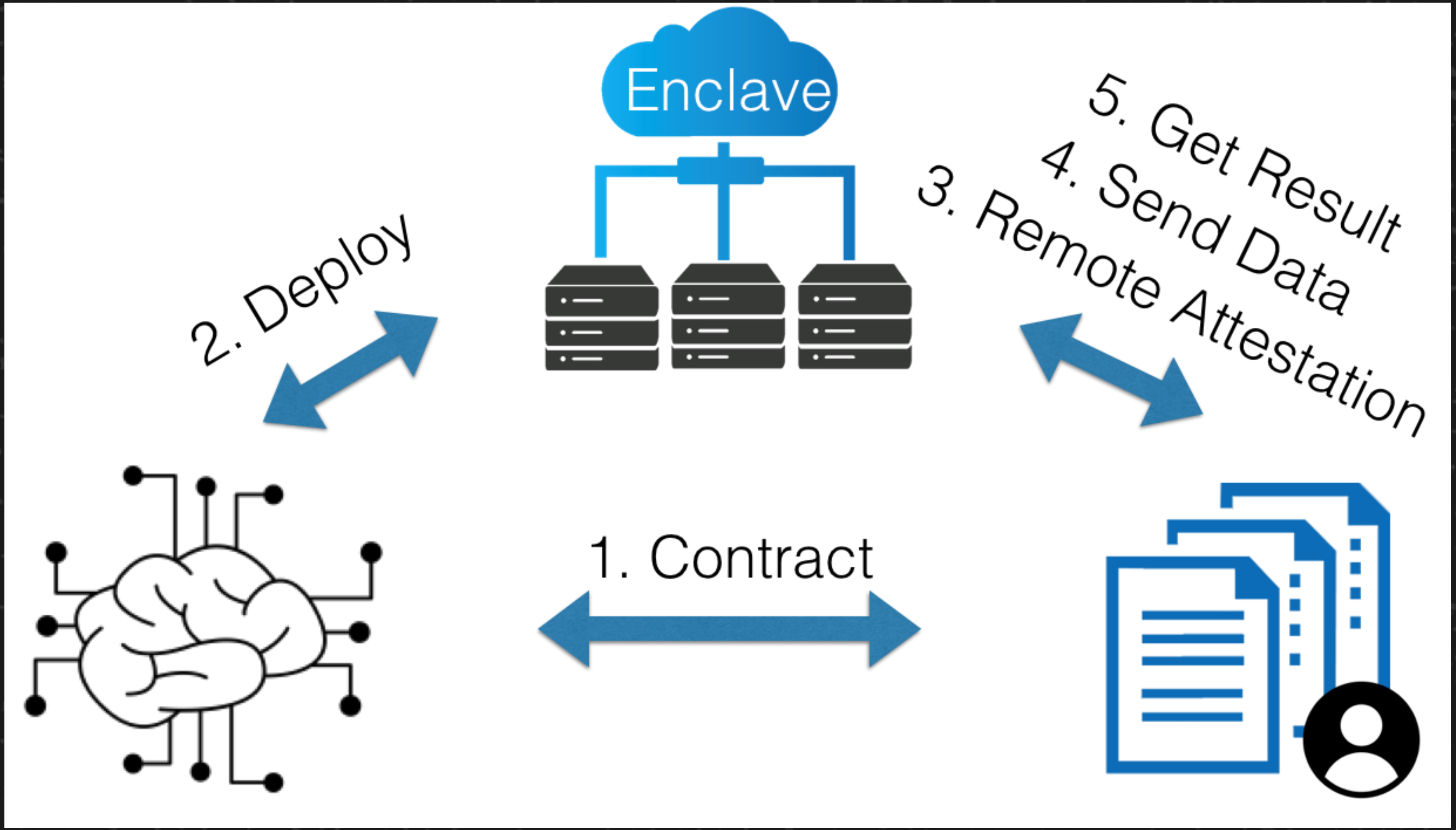
MesaTEE SGX

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

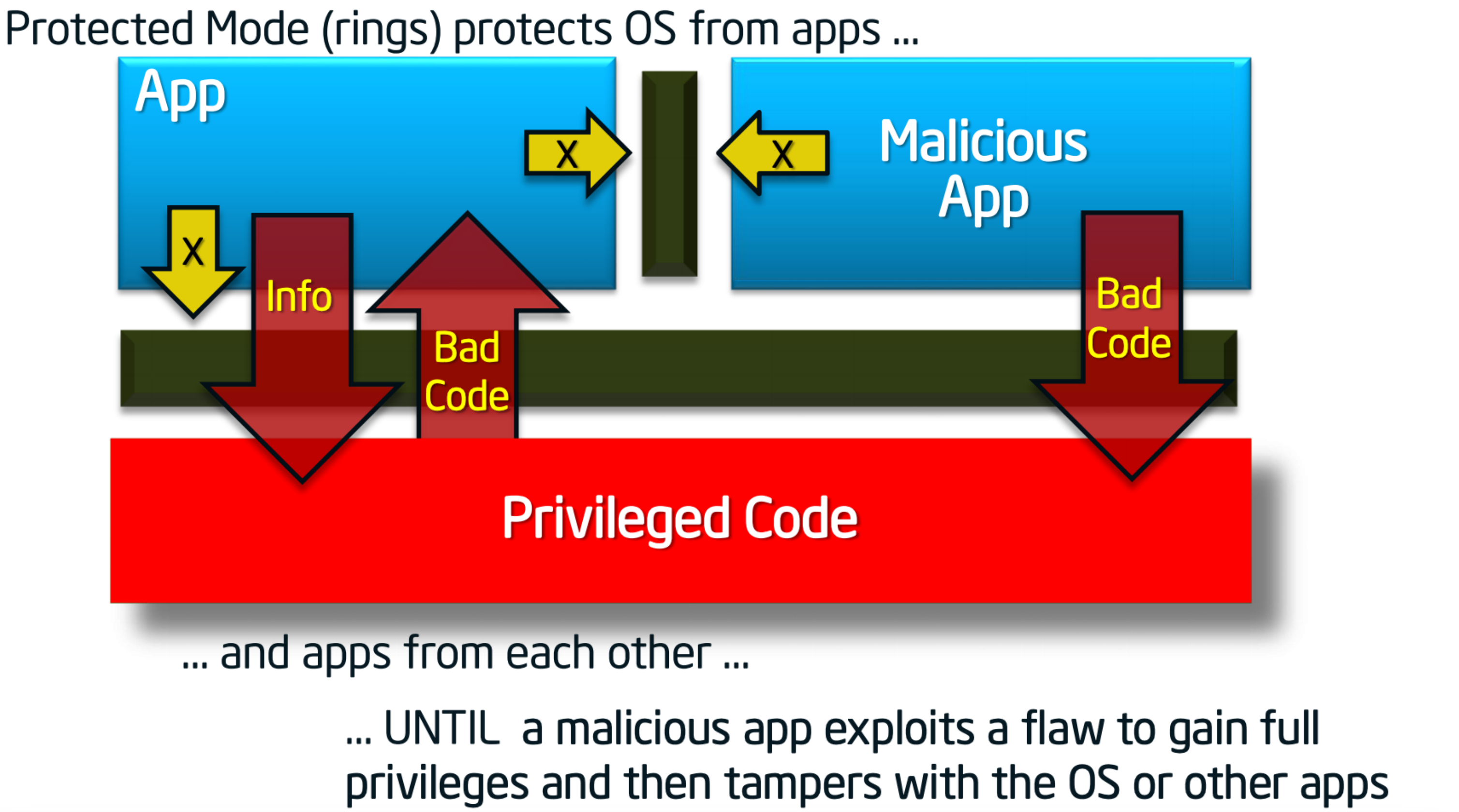
- Rule-of-thumb
- Practice on Intel SGX

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 …

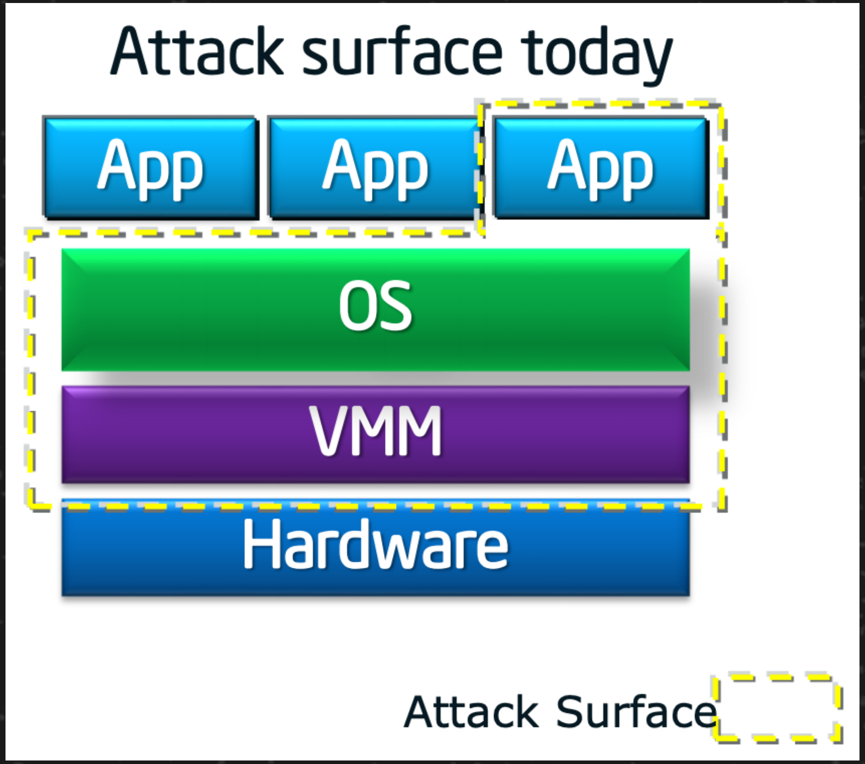
… and apps from each other …

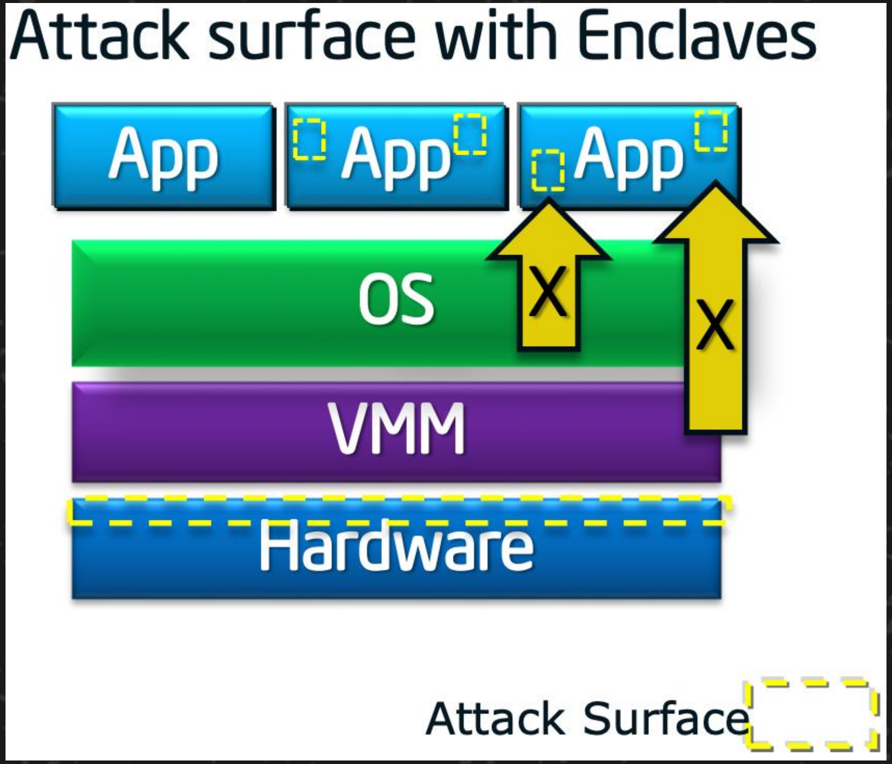
… UNTIL a malicious app exploits a flaw to gain full privileges and then tampers with the OS or other apps

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

Background of Intel SGX

Attack surface without/with Intel SGX Enclaves

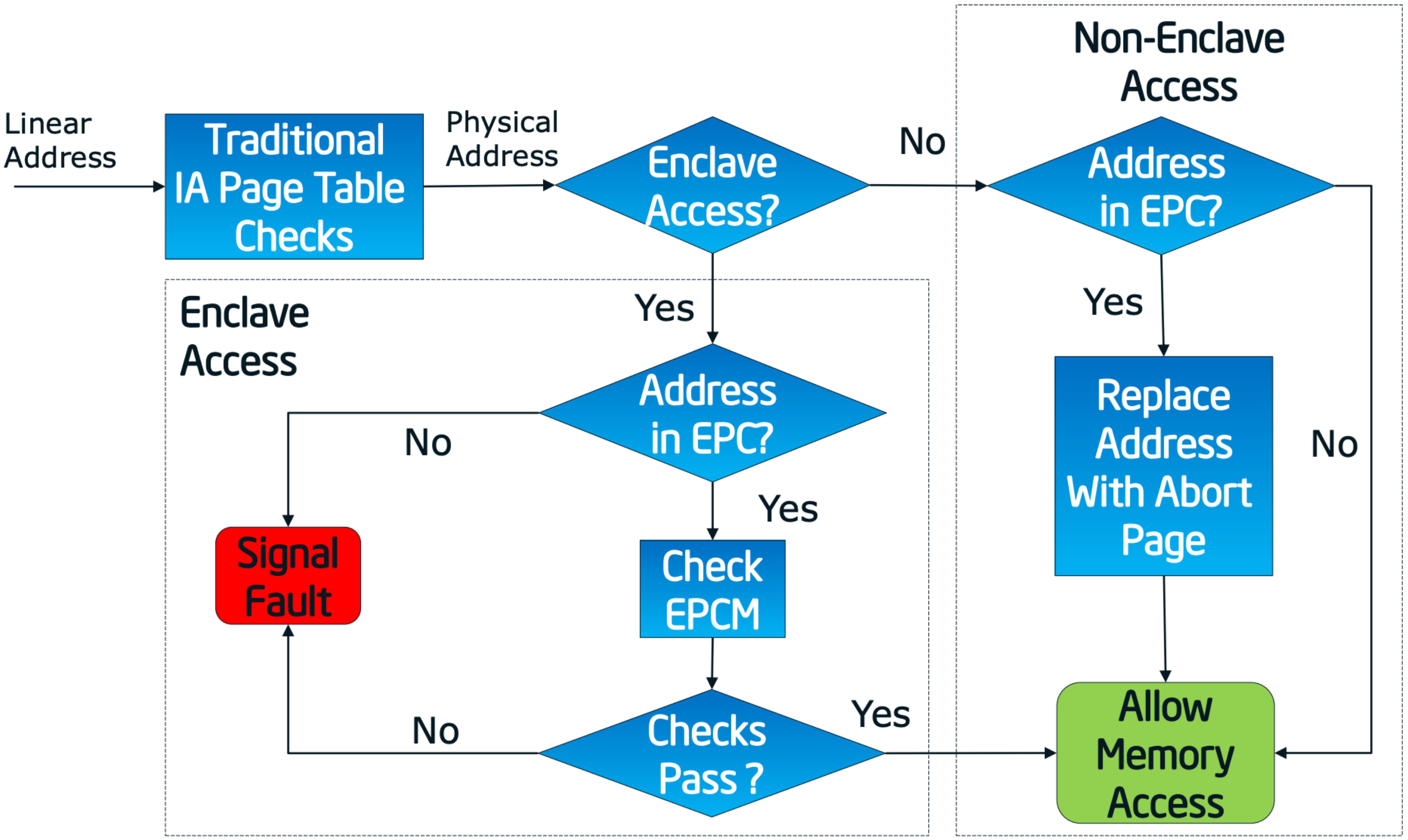




Intel® Software Guard Extensions(Intel® SGX)
Frank McKeen, Intel Labs, April 15, 2015

Background of Intel SGX

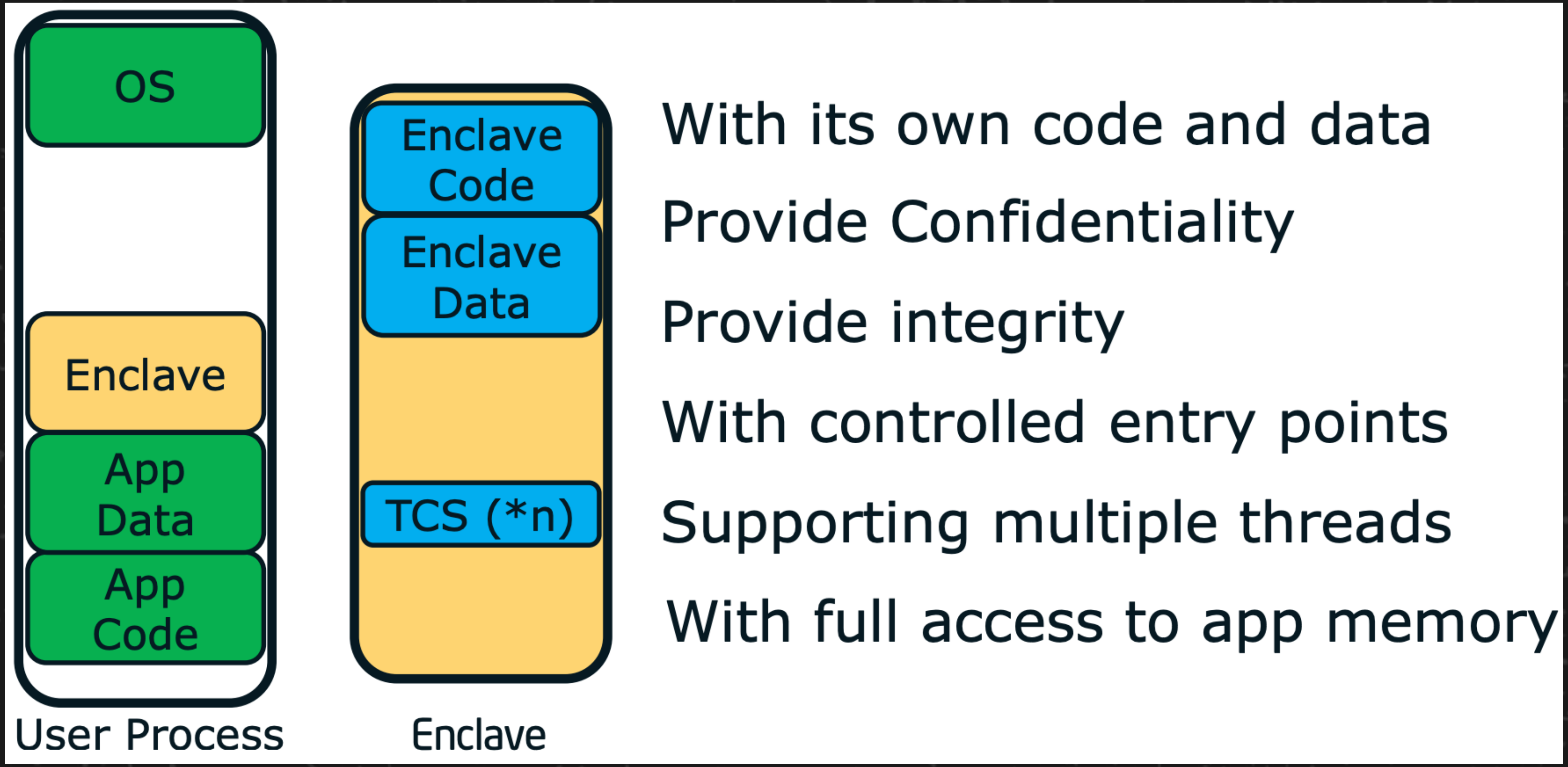
Memory access control during address translation



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

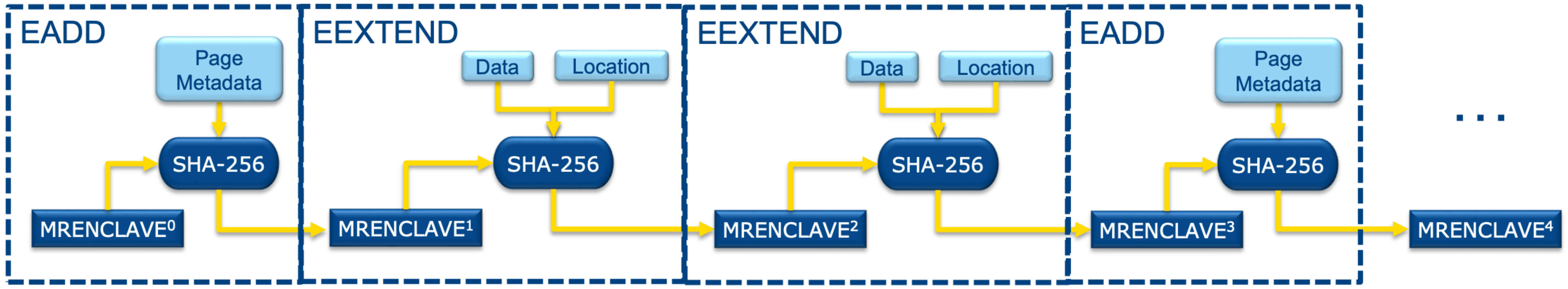
Intel® Software Guard Extensions(Intel® SGX)
Frank McKeen, Intel Labs, April 15, 2015

Background of Intel SGX

Measurement and Attestation

Verify the measurement/signer

Establish trust by Remote Attestation



Sealing and Attestation in Intel® Software Guard Extensions (SGX)
Rebekah Leslie-Hurd, Intel® Corporation, January 8th, 2016

Background of Intel SGX

Remote Attestation

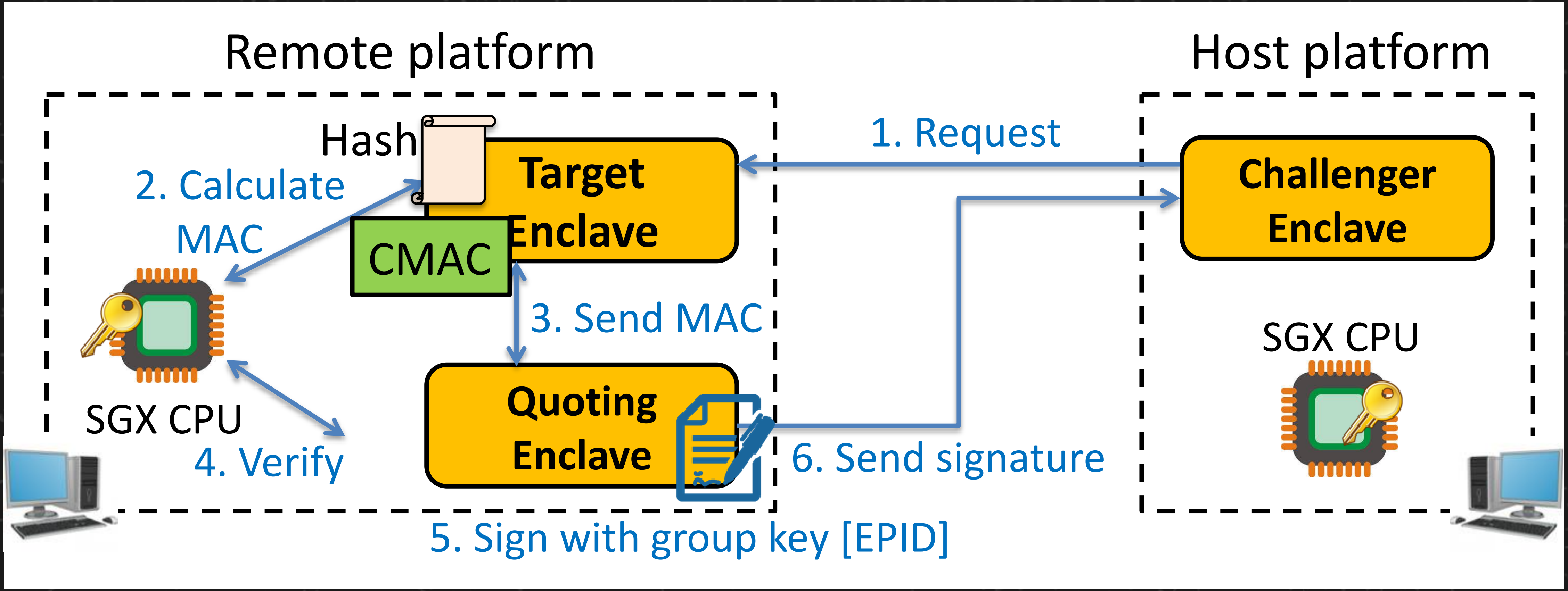


Figure is from "A First Step Towards Leveraging Commodity Trusted Execution Environments for Network Applications", Seongmin Kim et al.

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

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

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

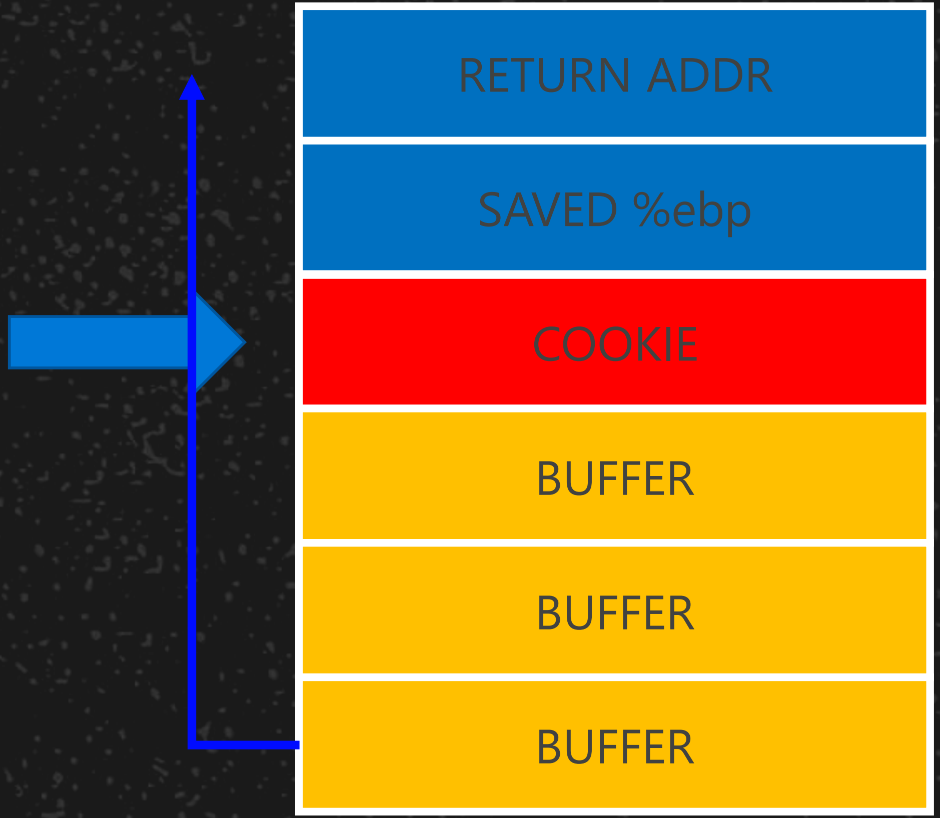
Challenges on building a privacy-preserving software stack based on Intel SGX

- Ha > Cha
- No
  - N
- No
  - N icate?
- No
  - S erforman
- 128 driven m
  - A
- 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

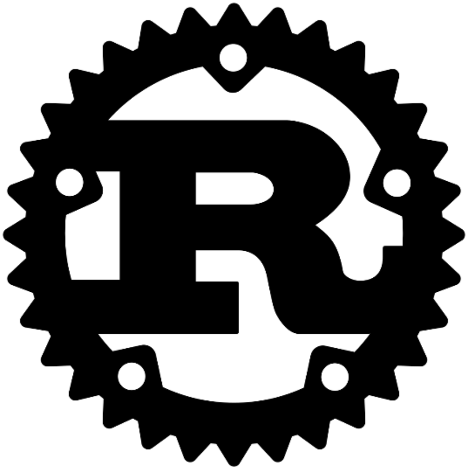
- Rule-of-thumb
- Practice on Intel SGX

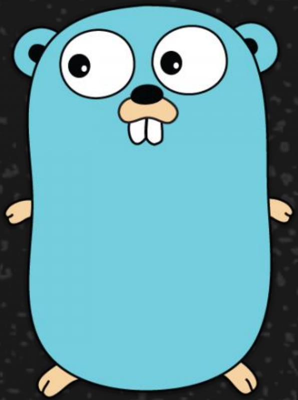
Towards a Secure and Trustworthy AI/Big Data Analysis framework

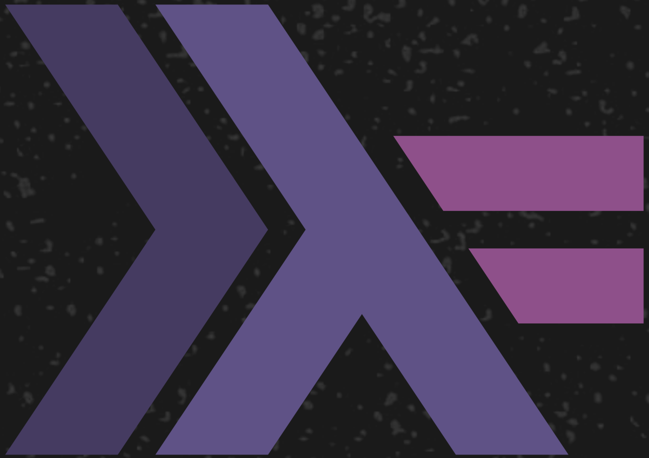
- What is trustworthiness?
- Achieving trustworthy AI/Big Data Analysis using Intel SGX

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

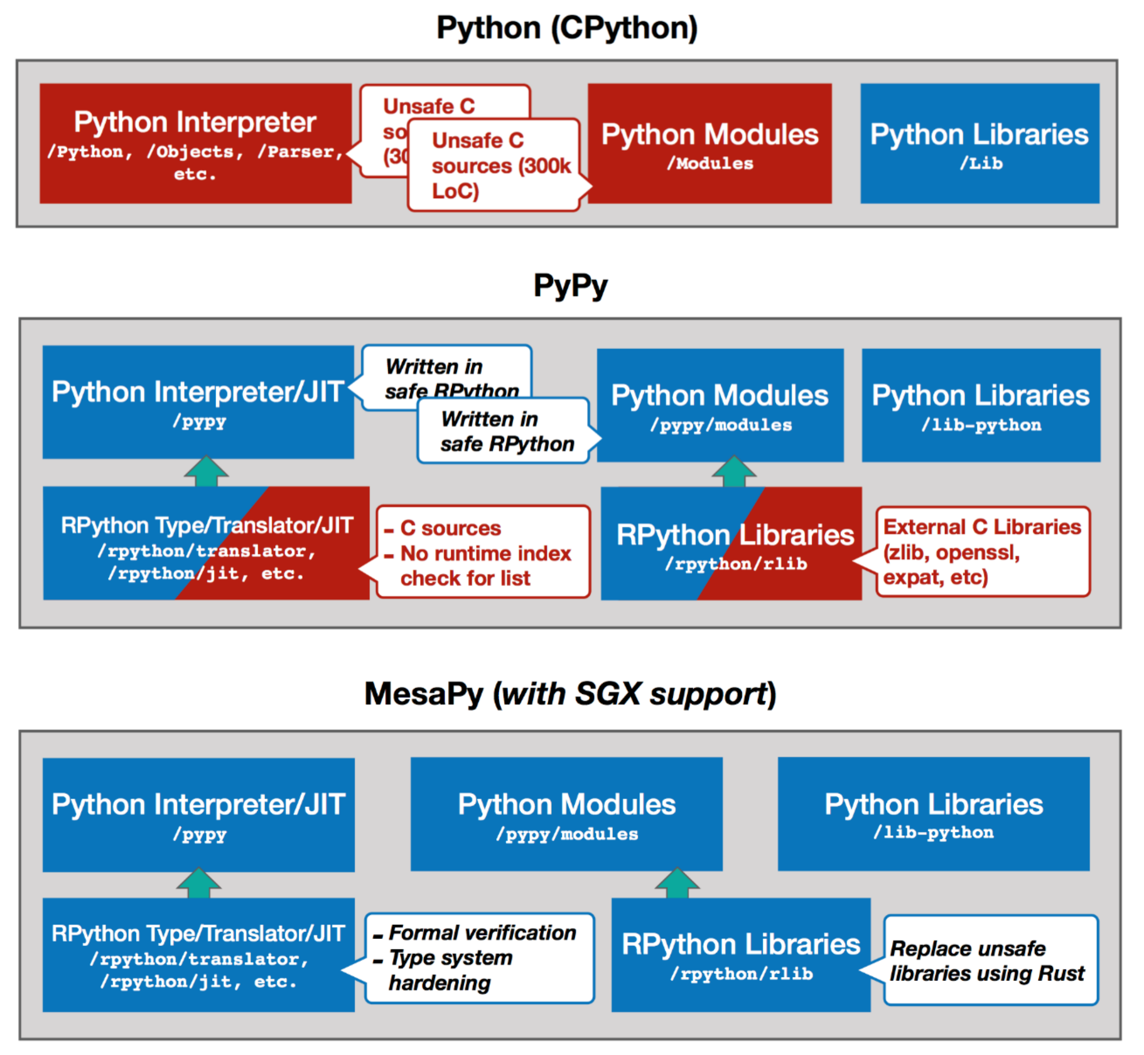
Hybrid Memory Safety

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



Python (CPython)

Python Interpreter
/Python, /Objects, /Parser, etc.

Unsafe C so (30

Unsafe C sources (300k LoC)

Python Modules
/Modules

Python Libraries
/Lib

PyPy

Python Interpreter/JIT
/pypy

Written in safe RPython

Written in safe RPython

Python Modules
/pypy/modules

Python Libraries
/lib-python

RPython Type/Translator/JIT
/rpython/translator, /rpython/jit, etc.

- C sources
- No runtime index check for list

RPython Libraries
/rpython/rlib

External C Libraries (zlib, openssl, expat, etc)

MesaPy (*with SGX support*)

Python Interpreter/JIT
/pypy

Python Modules
/pypy/modules

Python Libraries
/lib-python

RPython Type/Translator/JIT
/rpython/translator, /rpython/jit, etc.

- *Formal verification*
- *Type system hardening*

RPython Libraries
/rpython/rlib

Replace unsafe libraries using Rust

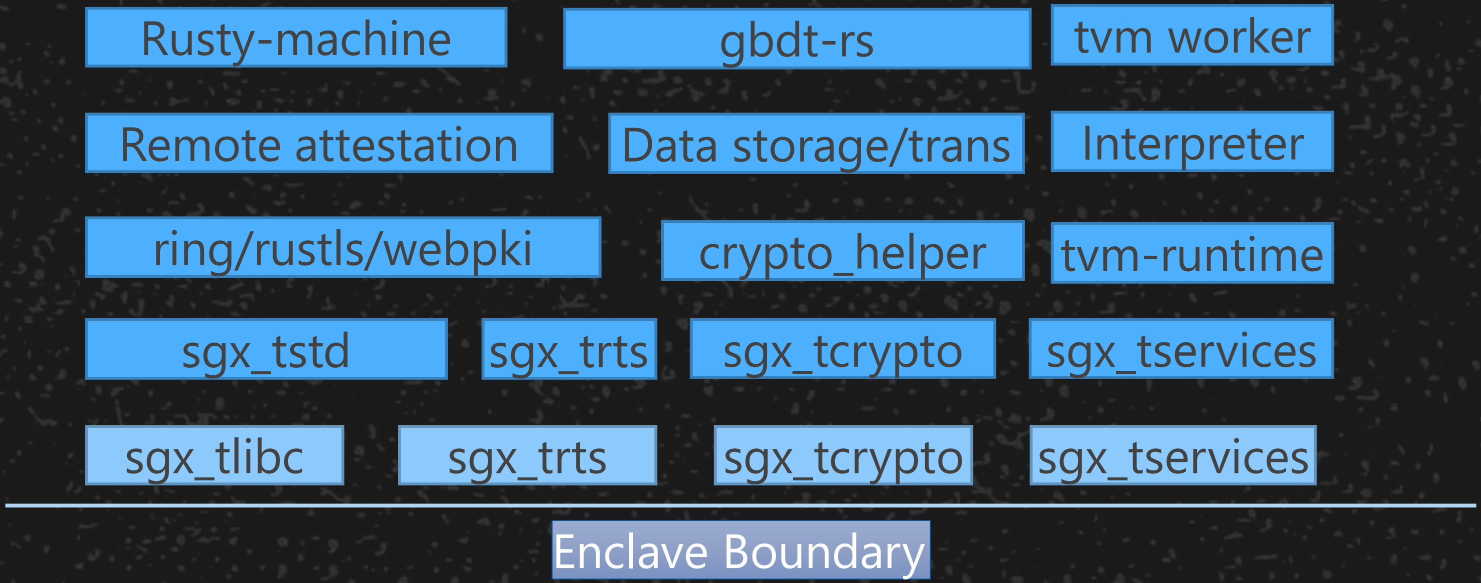
Hybrid Memory Safety

Hybrid Memory Safety – Practice in SGX

Linux	Rust-SGX
Kernel	N/A
Syscall	OCALL (statically controlled)
Libc	Intel – SGX tlibc
Runtime	Rust-SGX sgx_tstd/…

Hybrid Memory Safety

Hybrid Memory Safety – Practice in SGX



Rusty-machine gbdt-rs tvm worker

Remote attestation Data storage/trans Interpreter

ring/rustls/webpki crypto_helper tvm-runtime

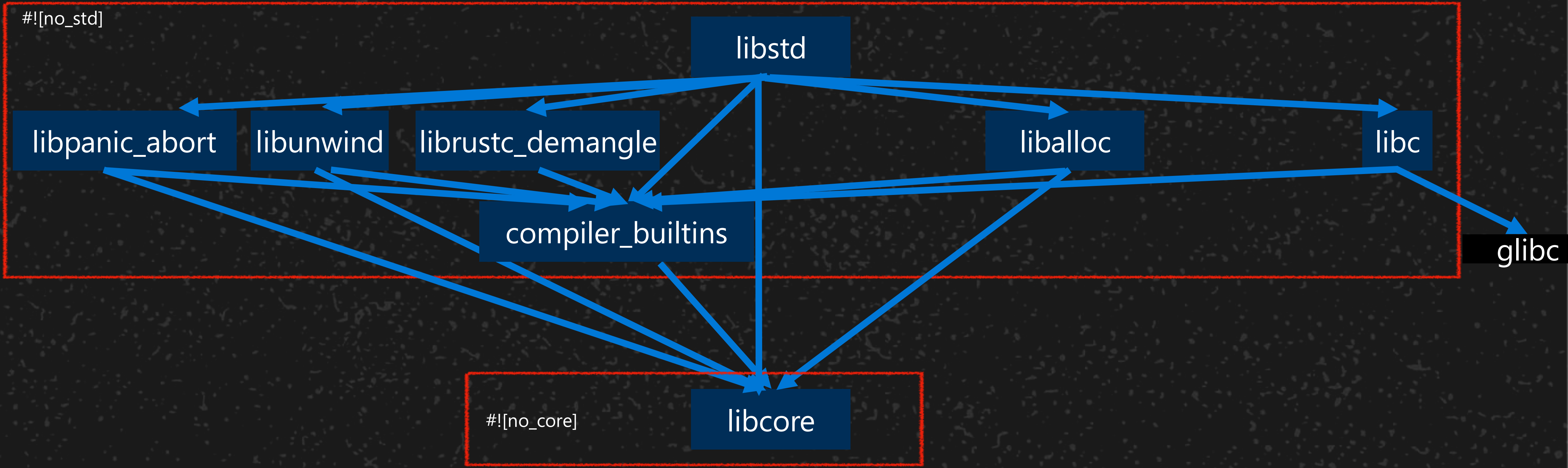
sgx_tstd sgx_trts sgx_tcrypto sgx_tservices

sgx_tlibc sgx_trts sgx_tcrypto sgx_tservices

Enclave Boundary

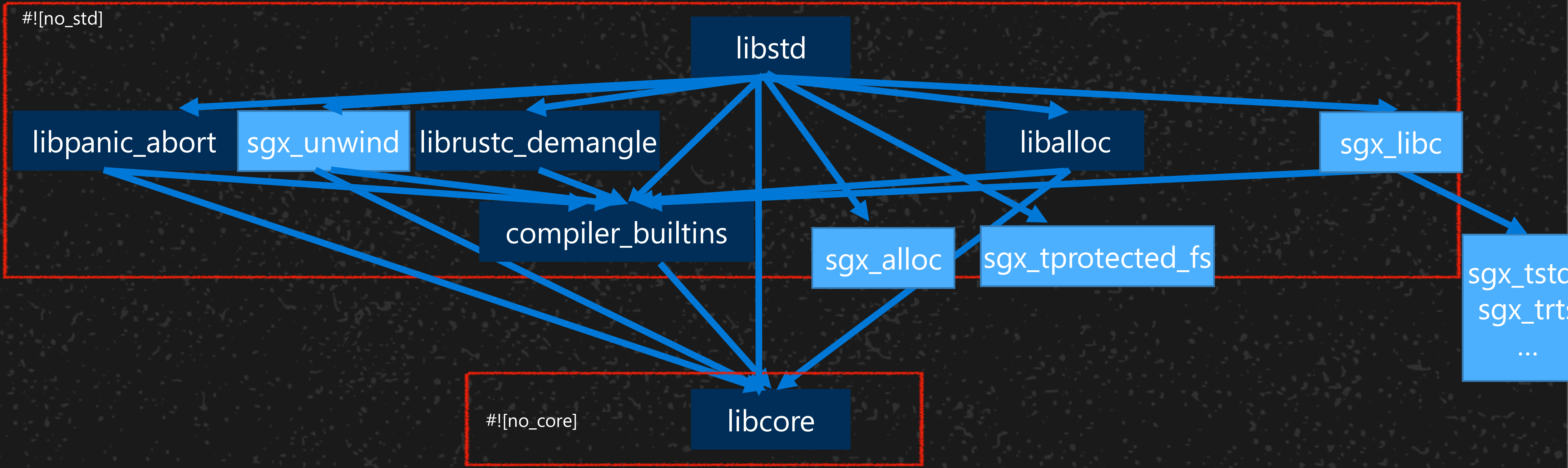
Hybrid Memory Safety

Hybrid Memory Safety – Practice in SGX



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?

Towards a Secure and Trustworthy AI/Big Data Analysis framework

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

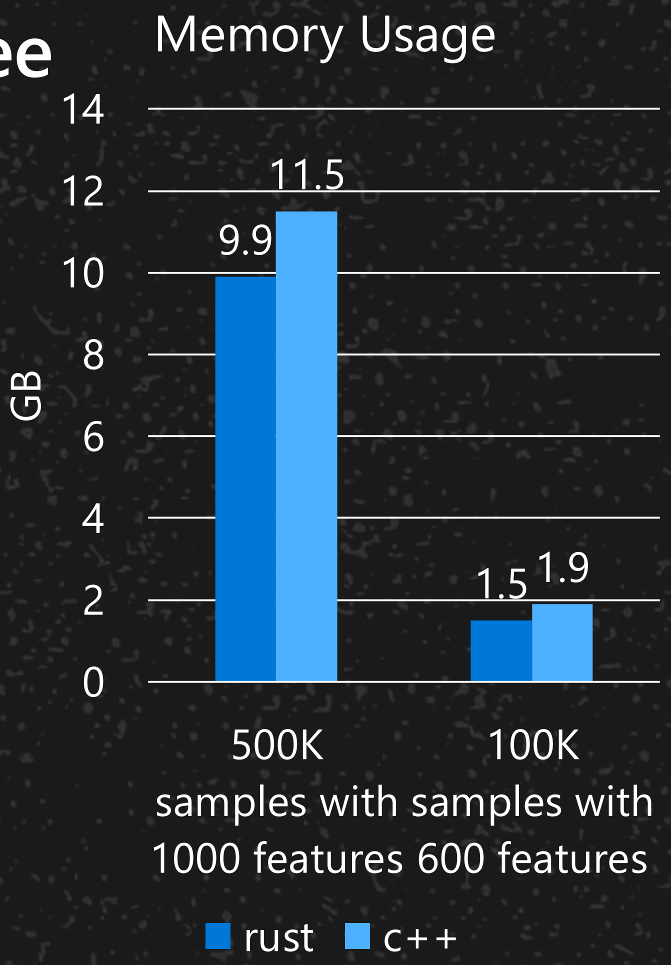
Towards a Secure and Trustworthy AI/Big Data Analysis framework

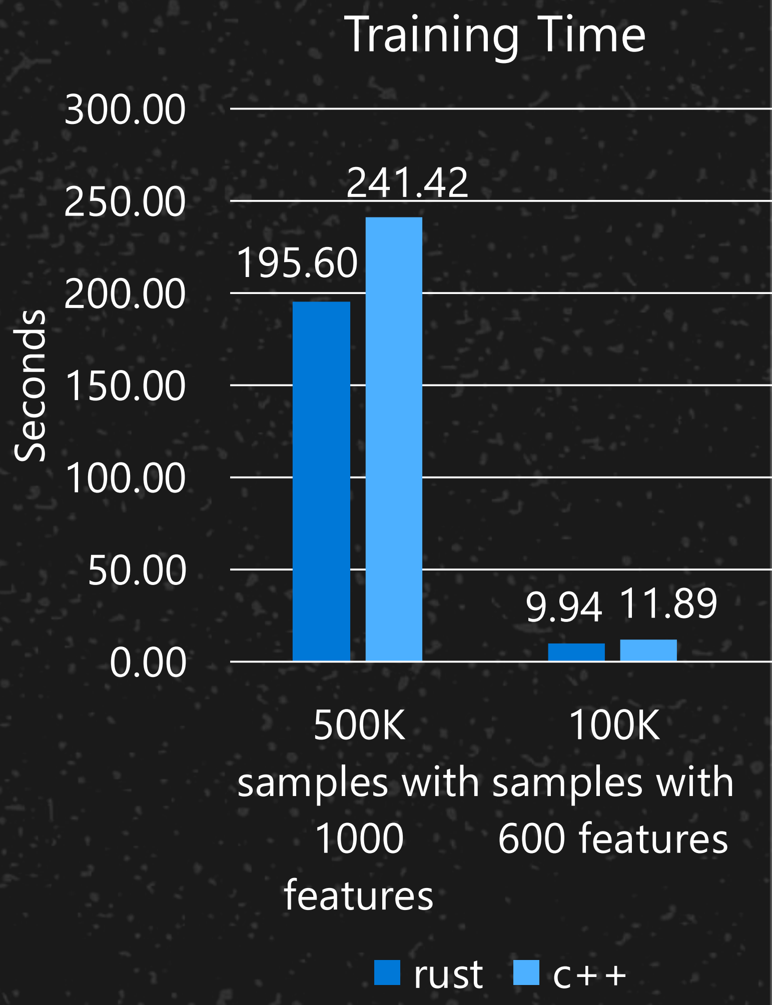
Achieving trustworthy AI/Big Data Analysis using Intel SGX

Gradient-Boosting decision tree

gbdt-rs

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



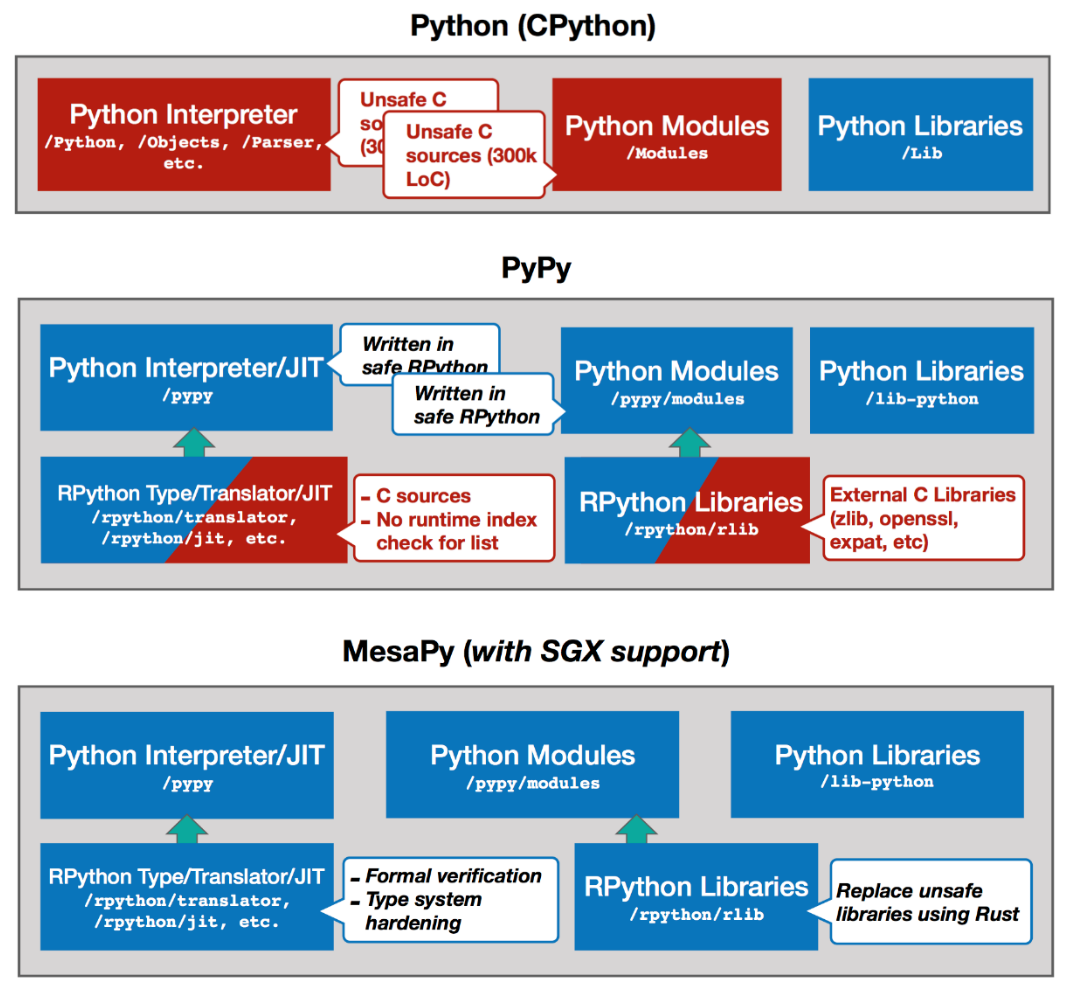


Towards a Secure and Trustworthy AI/Big Data Analysis framework

Achieving trustworthy AI/Big Data Analysis using Intel SGX

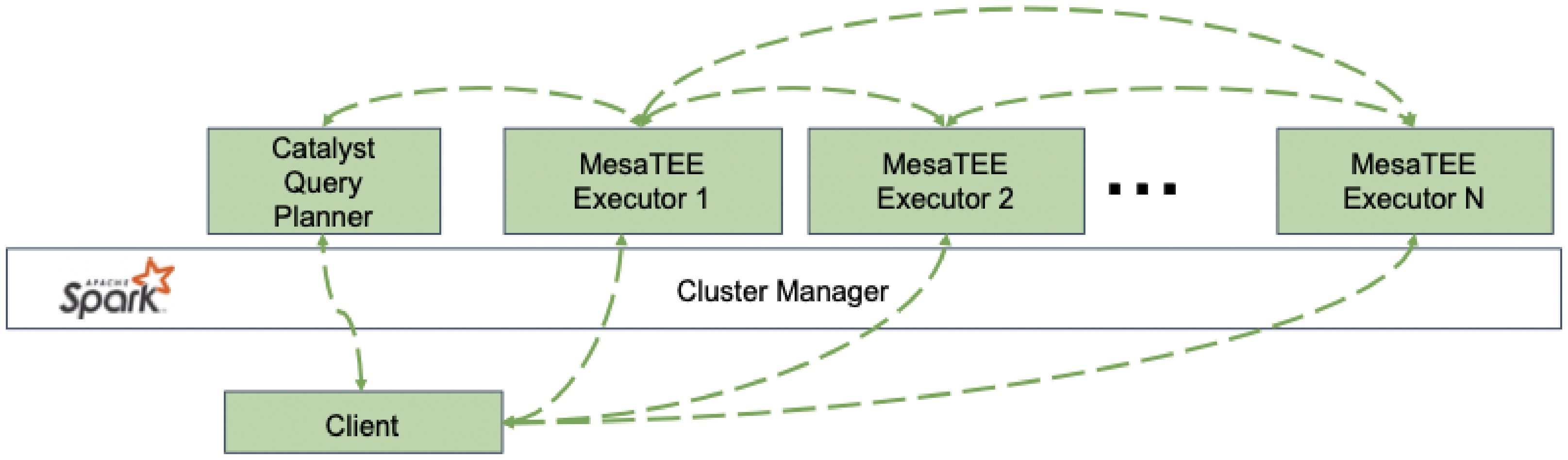
MesaPy SGX

- 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

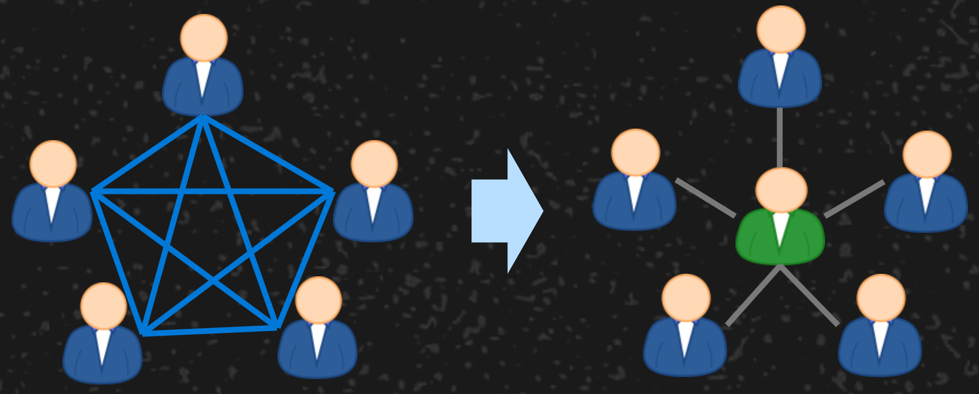
Achieving trustworthy AI/Big Data Analysis using Intel SGX

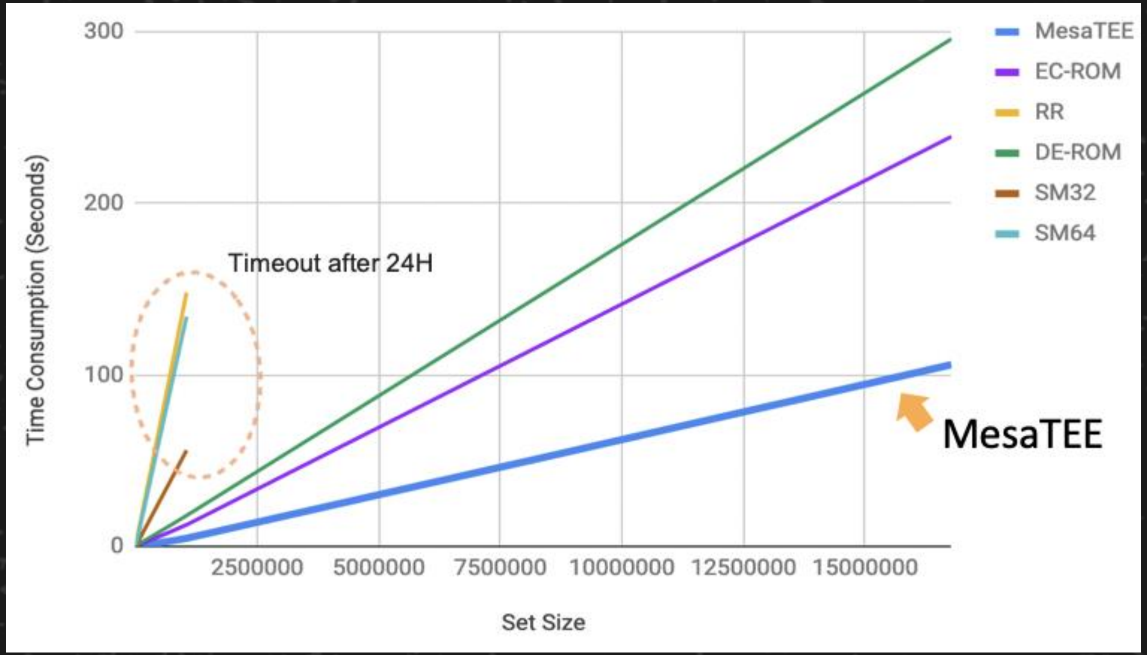


Solutions	Spark	MesaTEE Spark	GraphSC	ObliVM	Homomorphic Encryption
Data Encryption	x	√	x	x	√
Oblivious	x	√	√	√	x
Turnaround	1 sec	4-20 sec	2-6 days	>100 days	∞

Towards a Secure and Trustworthy AI/Big Data Analysis framework

Achieving trustworthy AI/Big Data Analysis using Intel SGX

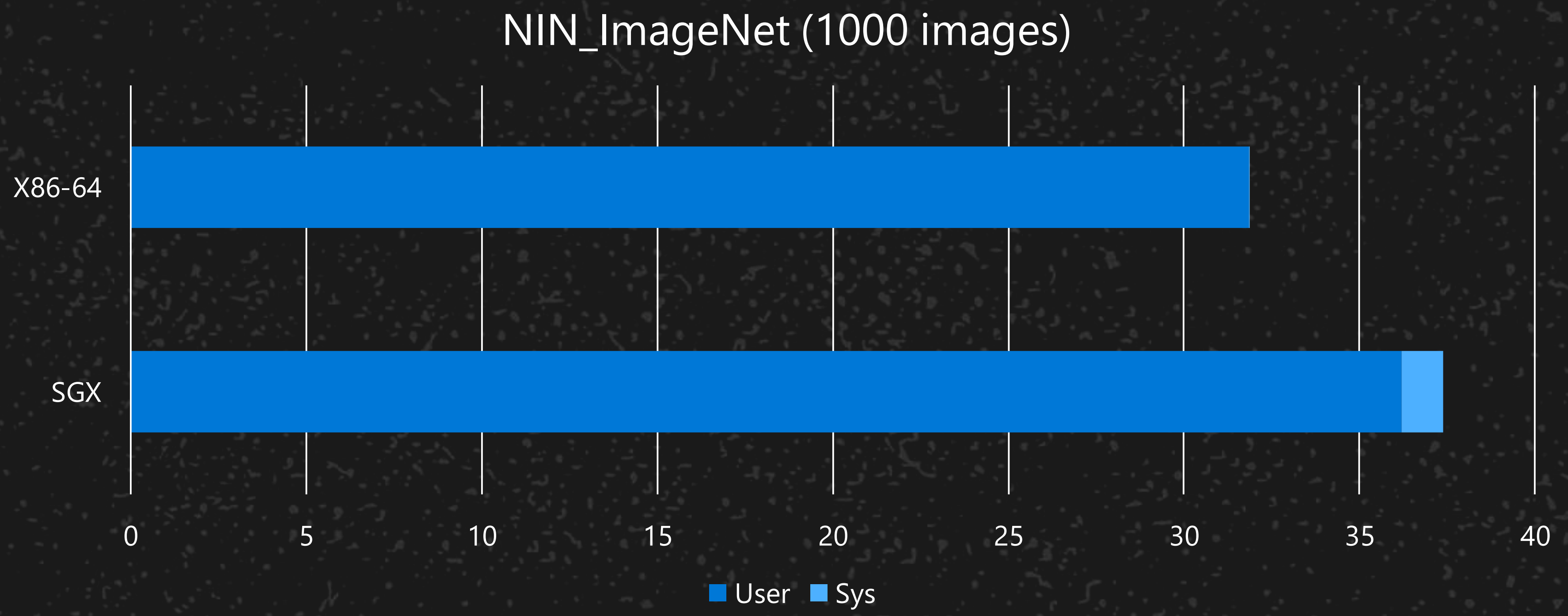




We are working with Baidu XuperData for applications

Towards a Secure and Trustworthy AI/Big Data Analysis framework

Anakin-SGX



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

Q&A

Yu Ding

Security Scientist, Baidu X-Lab