Graphene v1.0: Toward A Reliable, Open-Source Library OS for SGX

CHIA-CHE TSAI
TEXAS A&M /
GRAPHENE PROJECT
This Talk Includes:

1. WHAT’s Graphene, WHY, and HOW
2. Latest updates
3. Our future roadmap
SGX Is Not For “Dummies”

Developing a SGX application is hard

- Setup and configuration
- Porting legacy code
- OS interfaces and libraries
- Security implications
- Debug and performance tuning

You have to be a:
- system admin
- Linux/libc maintainer
- security expert
- computer architect?
What IS Graphene?

A system that runs unmodified* Linux apps on platforms like SGX.

* Native binaries, no code change, no recompile
Can We Improve the Dev Cycles?

Dev Cycle w/ SDK
- Deployment
- Design OCALL / ECALL
- Design Defense
- Integration
- Port Library
- Testing OCALL / ECALL
- Testing

Dev Cycle w/ Graphene
- Deployment
- Design / Adopt Native Program
- Testing on SGX
- Testing on Native
- Enclave Impl.
The LibOS Approach

Enclave

Unmodified program

Jailed environment without OS and legacy library routines
The LibOS Approach

Broadly speaking, lots of SGX frameworks qualify for this definition (Haven, SCONE, SGX-LKL, …)
How Are “LibOSes” Generally Built?

**Haven / SGX-LKL**
- Linux/POSIX/Windows APIs
- Legacy OS code
- Hypervisor-like interface
- Untrusted Host OS

**SCONE**
- Linux/POSIX/Windows APIs
- Redirection Layer
- Untrusted Host OS
How is Graphene Different?

1. Host interface defined with portability and security in mind
2. Rich functionality for Linux apps
Graphene: A Tailored LibOS

Step 1.
Define a host interface

Step 2.
Reimplement Linux APIs from scratch

300+ System Calls
(140 so far)

36 OCalls
+ 2 for Debugging

Untrusted Host OS
Why Host Interface Matters?

- Enclave
  - Unmodified program
    - System APIs
      - Malicious return values
  - Untrusted Host OS
Iago Attacks [ASPLOS 2013]

By SGX / Inktag / Overshadow / Nexus / Virtual Ghost / ...

App uses malicious results for sensitive operations:
Example:
Seed the RNG with (PID|TIME)
In A Nutshell

Iago attacks are semantics vulnerabilities caused by mistrusting legacy APIs
More Examples

Iago attacks are pervasive and often hard to mitigate

- File system metadata
- System time
- IPC (signals, message queues, shared memory)
- Scheduling
- System info (/proc, /sys, getrusage)
- Exception handling
Myths About Iago Attacks

1. Only system calls can suffer Iago attacks - wrong
2. Just leave to app developers - wrong
3. Orthogonal to system/runtime design - wrong
Decouple Shielding & Linux APIs

Enclave
LibOS

Platform Adaption Layer + Shield

36 OCalls
+ 2 for Debugging

Untrusted Host OS

Linux functionality

Portability & shielding
(Explicit semantics + crypto checks)
Multi-Process Applications

Distributed OS coordination

Supported:
- fork/exec
- signals
- message queue
- Semaphore

Attestation by CPU

Exchange Encryption Key

OCalls

Apache(P1)

kill
RpcSend()
RpcConnect()
LibOS

Apache(P2)

RpcListen()
More LibOS Features

140 / 318 system calls
Implemented (core features)

63 KLOC
Source code

1.4 MB
Library size
Tested Applications

See examples on: https://github.com/oscarlab/graphene
Graphene Project Updates

1. Open-source workgroup

2. Stable release plan (mid-August)
   - Reliability & Security improvements
   - New features
Graphene Workgroup

https://grapheneproject.io/
Stable Release (Mid-August)

Reliability improvements:

- 324 pull requests since Nov 2018
- Rewritten: memory mgmt., signal handling, IPC, and file system
- Data races and deadlock issues
- Better OCall interface and enclave initialization
- Documentation & UX improvements
Stable Release (Mid-August)

Vulnerability fixes:

- Received multiple reports from KU Leuven and U. Birmingham
  - Untrusted argv/envp
  - Untrusted memory allocation
  - Untrusted argument copy for OCalls
  - TOCTOU for untrusted copy
  - Incorrect pointer validation
- All fixed within 5 days
Stable Release (Mid-August)

New features:

- Support for GLIBC 2.23 / 2.27 and Ubuntu 18.04
- Static binary support for Golang support
- Simple remote attestation
- EXPERIMENTAL: Docker integration (Graphene Secure Container)
- EXPERIMENTAL: File system & network shield (RA-TLS) plugins
Remote Attestation

Enclave

Graphene LibOS

PAL

Only start if successfully attested

Intel Attestation Service

Retrieve attestation

Retrieve quote

Quoting Enclave
Docker Integration (EXPERIMENTAL)

Dockerfile ➔ Graphene Configuration ➔ Docker Engine ➔ Docker container

Generate

Enclave

Graphene LibOS

PAL
Future Roadmap

Periodic stable releases
File system shield and network shield (RA-TLS)
SGX2 (EDMM) support
Exitless enclave interface (optimization)
Support for upstream Linux driver
Conclusion

Why you should consider Graphene:
- Open-sourced (LGPL), good for customization and exploration
- Actively maintained by workgroup & community
- Tailored for small host interface and rich Linux functionality

https://grapheneproject.io
support@graphene-project.io