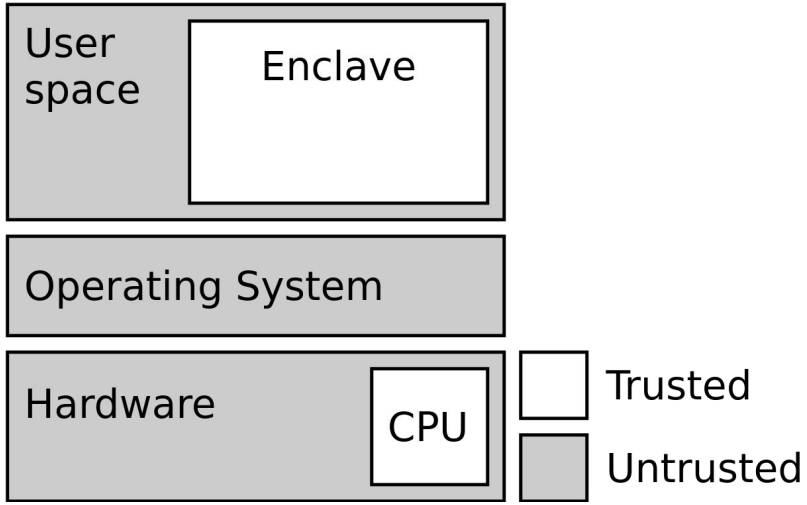
# Practical implications of Intel SGX with Graphene

July 4th, 2019

Derk Barten
Robin Klusman

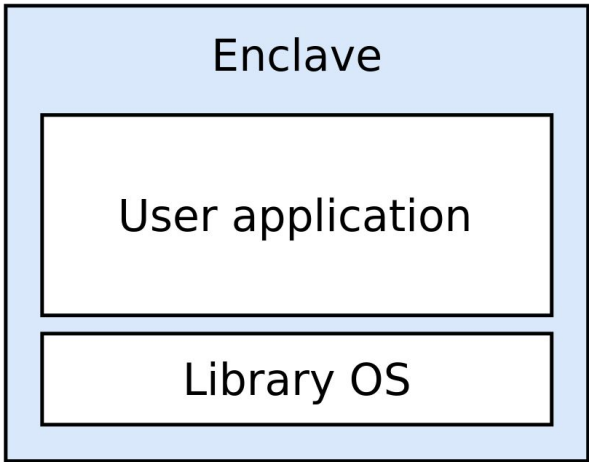
# Software Guard Extensions (SGX)

- Untrusted system
- Trusted enclave
- Attestation
- Encrypted & isolated memory
- Integrity, confidentiality, isolation



# Graphene-SGX

- Library OS
- Standard C library
- Unmodified applications
- Multi-process support
- Dynamic shared libraries
- Manifest



# Related work
# Use-cases

- SGX
  - DRM, Anti-cheat
  - Compilers
  - TLS termination
  - Databases
  - System logs
  - Middleboxes
- Graphene
  - No modifications required
  - Reduced development effort
  - Facilitate SGX research

# Related work
# Existing attacks on SGX

- Cache side channel attacks
  - Foreshadow
  - SgxPectre
  - BranchScope
  - CacheZoom
- Asyncshock
- Controlled channel

*What are the practical implications of running arbitrary applications in Intel SGX using Graphene-SGX?*

# Security implications

# Misaligned threat model

- Intel SGX
  - Operating system = untrusted
- Most applications
  - Operating system = trusted

Arbitrary applications are often not designed to guard against a malicious operating system.

# Iago attacks

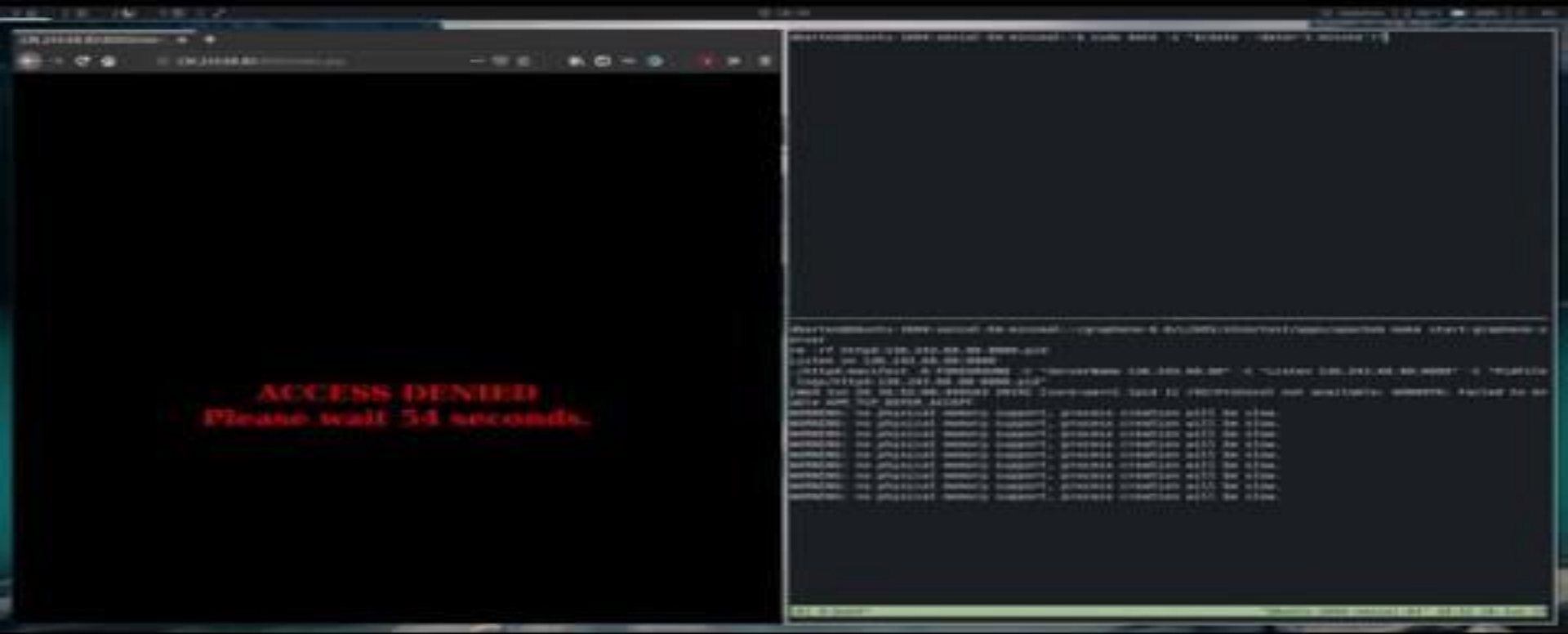
- Attacks by malicious kernel
- System calls

- Mitigation
  - Verification

# Date / time manipulation

- *gettimeofday()*
- Reliant on OS supplied vDSO
- Not verified by Graphene

- Implications
  - Transaction order
  - Kerberos
  - 2FA token validity
  - Rate limiting




Date / time manipulation demo

# Environment variable manipulation

- Arbitrary environment vars
- Not present in manifest
- Not checked by Graphene
- Easily overlooked

- Implications
  - Influence execution
  - GCC Epoch

# Framework maturity

# Running applications in Graphene

- OS version support
- Framework bugs
- Disk writes
- Non trivial to port complex applications

# Discussion & conclusions

# Discussion

- Security may be compromised
- Can be mitigated
- Graphene as research project
- Not ready for production

*Developers should take care when running arbitrary applications in SGX using Graphene, as there may be non-trivial security implications and framework bugs.*

# Future work

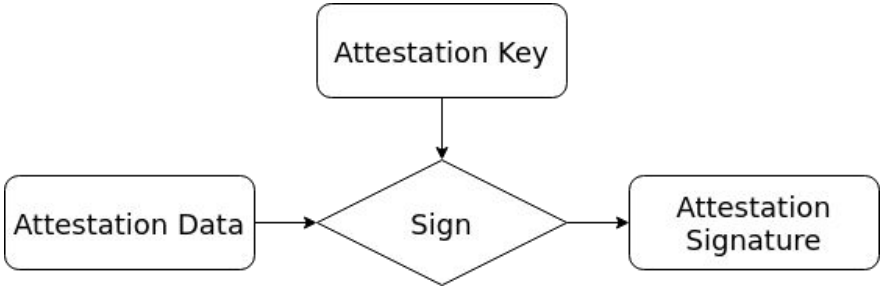
- Explore additional system calls
- Environment variable dependent applications
- Investigate SCONE/Panoply

# Sources


- Victor Costan and Srinivas Devadas. “Intel SGX Explained.” In: IACR Cryptology ePrint Archive 2016.086 (2016), pp. 1–118.
- Chia-Che Tsai, Donald E Porter, and Mona Vij. “Graphene-SGX: A Practical Library {OS} for Unmodified Applications on {SGX}”. In: 2017 {USENIX} Annual Technical Conference ({USENIX} {ATC} 17). 2017, pp. 645–658.
- Stephen Checkoway and Hovav Shacham. “Iago attacks: Why the system call api is a bad untrusted rpc interface”. In: ASPLOS. Vol. 13. 2013, pp. 253–264.
- Ofir Weisse et al. Foreshadow-NG: Breaking the virtual memory abstraction with transient out-of-order execution. Tech. rep. Technical report, 2018.
- Guoxing Chen et al. “Sgxpectre attacks: Stealing intel secrets from sgx enclaves via speculative execution”. In: arXiv preprint arXiv:1802.09085 (2018).
- Nico Weichbrodt et al. “AsyncShock: Exploiting synchronisation bugs in Intel SGX enclaves”. In: European Symposium on Research in Computer Security. Springer. 2016, pp. 440–457.
- Ahmad Moghimi, Gorka Irazoqui, and Thomas Eisenbarth. “Cachezoom: How SGX amplifies the power of cache attacks”. In: International Conference on Cryptographic Hardware and Embedded Systems. Springer. 2017, pp. 69–90.
- Yuanzhong Xu, Weidong Cui, and Marcus Peinado. “Controlled-channel attacks: Deterministic side channels for untrusted operating systems”. In: 2015 IEEE Symposium on Security and Privacy. IEEE. 2015, pp. 640–656.
- Dmitry Evtyushkin et al. “BranchScope: A new side-channel attack on directional branch predictor”. In: ACM SIGPLAN Notices. Vol. 53. 2. ACM. 2018, pp. 693–707.

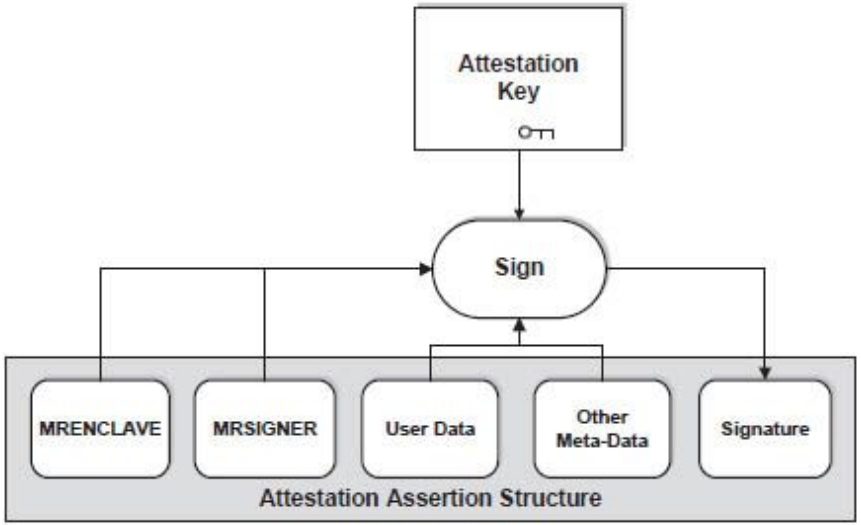
# Software Attestation

- Attestation data
- Attestation key
- Attestation signature



# Software Attestation SGX

- MRENCLAVE - Enclave Identity
- MRSIGNER - Sealing Authority Public key hash
- Attestation Key in µcode




Source: Intel documentation