Recommendations for TEEP Support of Intel® SGX Technology

Overview of SGX & Selected TEEP Topics

David M. Wheeler
david.m.wheeler@intel.com
Apologies...

• If you are really interested in the details of SGX
  This Won’t Satisfy Your Curiosity

• The best public paper can be found at:
  • Stanford Seminar YouTube: [https://www.youtube.com/watch?v=mPT_vJrlHlg](https://www.youtube.com/watch?v=mPT_vJrlHlg)
  [https://software.intel.com/en-us/sgx/academic-research](https://software.intel.com/en-us/sgx/academic-research)

Please refrain from asking deep questions on SGX Architecture that are not relevant to TEEP 😊

We are under a Time Constraint
What We Will Cover

• Overview of Intel® Software Guard Extensions (SGX)
• SGX TCB (Trusted Computing Base)
• SGX Attestation
Overview of Intel® SGX

- SGX provides a protected area of memory (EPC Memory) where enclave code and data exist
- Enclave code is packaged with the Client Application, and loaded into EPC Memory by the Application
  - To the REE, both memory areas are within the same process
  - Enclave is prevented from access into Client process memory, AND Client prevented from access to the EPC (see flow chart)
- A special enclave (Launch Control) is used to load an enclave with code and data, and the Launch Control verifies the code during loading (authentication, authorization & integrity)
  - The Enclave must be signed by the Launch Key
- Entering and Exiting the enclave are done through processor instructions
  - EENTER and EEXIT

EPC = Enclave Page Cache
What is Relevant to TEEP

- **Trusted Application is not Separate from Client Application**
  - SGX Applications include both the trusted part (Enclave) and the untrusted part (Client Application)
  - This doesn’t prevent a Client Application from presenting all information needed to “authorize” an SGX application to a TEEP Agent
    - Some information is embedded in the enclave
      - Authorized TAM or Service Provider (Mr. Signer), Integrity Proof (Mr.Enclave), Other Rights
    - Other information can be provided by the Client App
      - TAM Identity & authorization signatures, Other stuff?

- **There is no Security Domain**
  - Only one “program” can be loaded a single enclave – multiple separate enclaves can exist simultaneously
  - One can consider an Enclave as a single domain for only one TA
  - Optionally, an implementation of a TEEP Agent can manage TA interactions “as if” they were in the same Security Domain (e.g. secret sharing, secure channels, etc.)

- **There is no internal Agent watching all Enclaves**
  - It isn’t possible to report on all the “installed” TAs – installed TA’s take no resources from the TEE until loaded
  - It isn’t possible to report on all the “running” TAs – as they do not know about each other
    - TEEP Agent could report on all TA’s that it loaded as running enclaves
    - Launching an application that contains an enclave does not mean the enclave gets loaded
How would SGX Use TEEP?

**Install / Uninstall**
- There is no *real* install/uninstall commands in SGX
  - Any application on the platform file system can carry SGX enclave code (a TA)
  - Same vector as any REE Application install (e.g. HDD, Flash, USB Stick, Network, etc.)
- One option could be signing the SGX enclave code (TA) so that it can be launched
  - For example:
    1. Service Provider requests TAM to prepare a particular Application for an SGX Platform (e.g. Install)
    2. The TAM holds the Enclave Signing Key for some platforms
    3. TAM authorizes SP, and if OK, then signs the requested Enclave & delivers it to the Platform
- Simplifies Application Developer deployment

**Start / Stop**
- An SGX enclave is launched (Started) by the application (not by the TEE)
- TEEP Start could be mapped to Client Application launch
  - However, the Client Application can delay the launch of the enclave to a later time
Intel® SGX Trusted Computing Base

- Security Perimeter is the CPU package boundary
  - Data/Code inside CPU is unencrypted
  - Data/Code outside CPU is encrypted/integrity protected

- BIOS is formally outside the TCB
  - BIOS controls how much memory is allocated to EPC, but cannot affect the security of EPC memory

- OS is formally outside the TCB
  - OS controls page tables, but does not control the security or attributes of the pages
  - Interrupts and certain OS features (files, network sockets) are still handled by OS, but considered in Application scope/control
    - State is saved in special EPC memory area for interrupts and context switches

Attack Surface under Regular REE

Attack Surface under SGX w/REE

Access by OS / VMM to Enclave is prevented
What is Relevant to TEEP

- SGX does not depend on Secure Boot
  - SGX has its own Roots of Trust for:
    - Measurement (RTM), Integrity (RTI), Verification (RTV),
    - Confidentiality (RTC), Reporting (RTR), Storage* (RTS)

- On an SGX platform, Secure Boot may NOT be turned on
  - Not possible to report from SBM

* The RTS is limited to providing sealing keys – actual storage is based on OS services
Intel® SGX Attestation

- **SGX Includes Two forms of Attestation**
  - Local Attestation – Hash Based
  - Remote Attestation – EPID Signature (Elliptic Curve Group Signature)

- **Local Attestation**
  - AES–CMAC Key Generated from Enclave & Platform
    - Attributes of the Enclave (Signer, Integrity Measurements, Version, etc.)
    - Platform Attributes (Fuses, Microcode Version, CPU Serial #, etc.)
  - Allows inclusion of other message via Hash
  - Can be sent to other enclaves on same platform

- **Remote Attestation**
  - Built from Local Attestation by SGX Signing Enclave
  - Only Signing Enclave has access to EPID key (the RTR)
  - Requires an External Verifier for EPID Signatures
What is Relevant to TEEP

- **Add EPID Digital Signature Algorithm as Optional to Support**
  - Will be supported by default on SGX-Enabled Platforms
  - Must be supported by TAMs to consume attestation from SGX Platforms
    - Or offload to an Intel® SGX Verifier
  - Needed to support SGX Attestation Signatures
  - The only way to verify trust in an SGX Enclave
  - Can use an Attestation to “certify” another RSA or ECDSA key pair
    - This would enable SP to have an Application/TA-specific RSA or ECDSA key pair

- **Local Attestation can be used to provide communication between the TA’s and a TEEP Agent**
  - Can be used to simulate Security Domains and “Universal TEE Knowledge” for reporting state
ISO/IEC 20008-2 Known Patent Rights

- The following are the known (to me on 7/16/2018) IPR claims on EPID
- I make no claim on the part of Intel or other parties that this list is complete or accurate

1. ISO/IEC 20008-2 (EPID Group Signature)
   - NEC corporation - RAND/reciprocal
   - Electronics and Telecommunications Research Institute (ETRI) - RAND/reciprocal

   - China IWNCOMM Co., LTD. - RAND
   - Electronics and Telecommunications Research Institute (ETRI) - RAND/reciprocal
Other Crypto Recommendations

- **NIST Recommends moving to larger Key Sizes**
  - NIST Recommendations
    - RSA-3072, RSA-4096, RSA-2048
    - ECDSA using NIST P-384, NIST P-256
    - ECDSA using Ed448-Goldilocks, Ed25519
    - EPID 2.0 Group Signature (Elliptic Curve w/ Bilinear Maps, TCG DAA group signature scheme)

- **Minimal to support should be**
  - RSA-3072-bit or larger
  - Diffie-Hellman (DH) 3072-bit or larger
  - ECDH with NIST P-384
  - ECDSA with NIST P-384
  - SHA-384
  - AES-256


What TEEP Services are Relevant to SGX?

- **TEEP on SGX will likely operate much differently than on a TZ platform**
  - The TEEP Agent’s counterpart “inside the TEE” will be an enclave just like every other enclave
    - The TEEP Service Enclave will perform services as if it were managing the whole TEE
    - But can only manage Enclaves that “cooperate” – have a TEEP Agent Helper library as part of their enclave/App
    - The TEEP Service Enclave will provide information on a “best effort” basis – may not know about all enclaves
  - TEEP will only “see” the applications installed/started/stopped through TEEP

- **Get Device State is a “Best Effort” Service**

- **Install/Uninstall a TA is equivalent to same operation on a Client Application**
  - TEEP Agent can report on TAs installed through TEEP, but not on ALL TAs/Applications
  - TEEP Agent cannot prevent TAs / Client Applications from being deleted (Denial of Service)
  - TEEP Agent may not be able to delete / remove a TA (depends on implementation)

- **TEEP Services are Useful in an SGX Environment, but will be limited**
A Standard SGX TA today is signed by the developer, but the developer’s key is register (signed by) the Launch key (as a CA) allowing the developer’s code to run as an enclave.

A TA can be signed by the developer, but not signed with a key that is recognized by the SGX Launch control on the platform. In this situation, a TAM can provide a valuable service to sign the TA so that it can be launched on the platform.
SGX App+TA Launch with TEEP Agent

- No need for TAM interaction
- TEEP is used as a “courtesy” registration service
- TA must be instantiated by the Client App, because it must be instantiated inside the Client App’s process
- There is no Security Domain – or SD of size One
- When Client App goes away, so does the TA’s Enclave
- There is no “Uninstall TA”, “Update TA” or “Delete TA”
What if TA’s weren’t Bound to the Client App

- We could use TEEP for a “TA Registry”
  - Client Apps could “ask for a particular type of TA in the platform’s TA registry”
  - But if this done by through an untrusted interface (TEEP Agent on REE side), then how does the App trust this service?
  - Remember, that the Client App must instantiate the TA, so it must be “given a bag of bits”
  - The “bag of bits” must be signed by the TAM, AND is must be verifiable in some other way (hash of the TA, manifest under signature with version #, etc.)
SGX TA Install with TEEP Agent

1. **Device**
   - Client App
   - Dependencies
   - Developer Signed

2. **Ask for TA's**
3. **TAM Interaction to get signed TA for the platform**
   - If it isn’t in Registry

4. **Enclave**
   - TA Instantiated as Enclave
   - TEEP Agent Enclave
   - Registry of TAs

- TAM Delivered to TAM
- TA Delivered to TAM
- TAM Interaction to get "missing TAs"
- TEEP is more than a "courtesy" registration service
- TA must still be instantiated by the Client App, because it must be instantiated inside the Client App's process
- There is no Security Domain – or SD of size One
- An "Uninstall", "Update TA" or "Delete TA" is used to control the registry of the TEEP Agent Enclave

**ISSUE** – How does the Client App really TRUST the TEEP Agent that it isn’t giving it a different “bag of bits” than what the TEEP Enclave actually provided? Client App must do secondary verification – TA hash, version #, developer signature, etc.
Summary Recommendations for TEEP SGX Support

- **TEEP MUST** support TA delivery within a Client Application
- **TEEP MUST** support EPID Signature Algorithm as Optional
- **TEEP SHOULD** look to support longer key sizes due to Post-Quantum recommendations
- **TEEP SHOULD NOT** require Secure Boot Attestation
  - SBM and TFW are not required of all platforms
  - Attestation Report should be flexible, allowing only required platform-specific elements
- **TEEP SHOULD** further explore the Security Domain Concept and only if valuable and necessary, then develop a crisp definition and model for Security Domains
  - This crisp model should encompass platforms that create SD’s of size One
- **TEEP MUST AVOID** definitions of operations that are very platform specific
  - Secure Boot, specific types of reporting, and platform state
  - Some reporting needs to be considered ‘Best-Effort’ or contain a quality-of-reporting