Formal Verification of an Open-Source Secure Enclave

Pranav Gaddamadugu
pranavsaig@berkeley.edu
Problem Definition

- Verifying hyperproperties about the Keystone Security monitor
- Secure Remote Execution (SRE) : a 2-safety hyperproperty that can be decomposed into guarantees on:
  - Integrity
  - Confidentiality
  - Measurement

- Previous models assume a fixed implementation of a TEE, our work allows for easy compositional verification of various hardware components and Keystone plugins
Prior Work

- ‘A Formal Foundation for Secure Remote Execution of Enclaves’
  - Subramanyan, Sinha, et al. at CCS ‘17

- Introduces a model of a Trusted Abstract Platform (TAP)

- Defines three separate adversary models:
  - M, MC, MCP

- Proves SRE for Intel SGX and MIT Sanctum
Proof Methodology

- Show that TAP guarantees SRE under the three adversary models
- Show that models of SGX and Sanctum are refinements of the TAP model under specific adversarial parameters

\[ \sigma_i, \sigma_j \in \text{States of TAP Model} \]
\[ s_i, s_j \in \text{States of Keystone Model} \]
\[ \sim, \sim_L := \text{Transition Relations} \]
\[ R := \text{Refinement Relation} \]
Redesigning the Model for Modular Verification

● Translated TAP model from Boogie to UCLID5
  ○ Toolkit for formal specification and verification of compositional systems
  ○ Suited for reasoning about the composition of Keystone and additional plugins
  ○ Future work on automatic invariant generation

● Extensions to UCLID5
  ○ Support for modular procedure-level verification, additional features for easier programmability, modifications to proof techniques
Extending the Model

- Extension of the adversary model to physical attackers
  - Enclave platforms also provide guarantees ‘physical attackers’
  - We define a physical attackers as ‘an adversary with the capability to observe or tamper with any signal leaving the chip package’
  - Involves the addition of an abstract memory encryption engine, as well as a semantic embedding of ciphertext and plaintext
TAP Model Design

Platform

Abstract CPU

Abstract Cache

Physical Memory

Platform

Abstract MEE

Abstract Cache

Abstract CPU

Physical Memory
Keystone Model and Augmentation

Base SM

Abstract CPU

PMP

Abstract Cache

Physical Memory

Base SM

PMP

Cache Partitioning Plugin

Abstract CPU

FU540 Cache / Controller

Physical Memory
Future Work

- Write the Abstract MEE model and augment proofs to show that TAP+MEE provides SRE under a physical adversary
  - Refinement proof (once Memory Encryption is added to Keystone)

- Exploring automatic invariant generation
  - Implementing a native SyGuS solver in UCLID5
  - Generating invariants based off of TAP and Keystone model sketches
Thank you! Any questions?