A Cryptographic Treatment of Software Guard Extensions

M. Barbosa **B. Portela** G. Scerri B. Warinschi

InfoBlender – 1st of July 2015

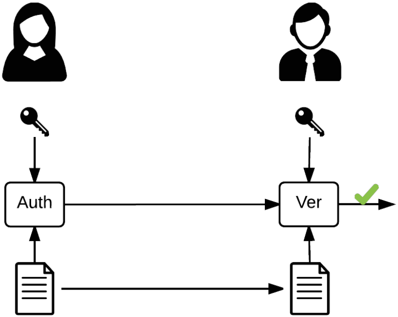
HASLab INESCTEC

Overview

Message Authentication Codes

The Message Authentication Code (MAC) is a cryptographic primitive that handles message integrity in a symmetric setting:

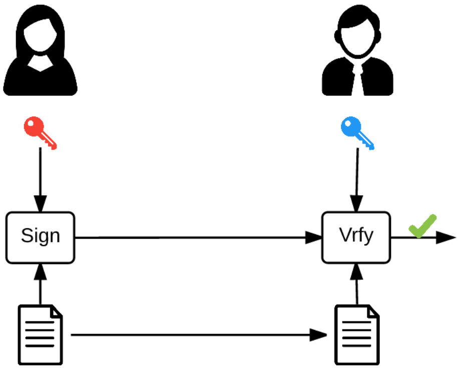
- **Auth** generates a MAC code given a *symmetric* key and some data.
- **Ver** takes a MAC and *the same* key, and verifies the integrity of the received content.



Digital Signatures

The Digital Signature is a cryptographic primitive that handles message integrity in an asymmetric setting:

- **Sign** generates a signature given a *private* key and some data.
- **Vrfy** takes a signature and the associated *public* key, and verifies the contents of the signature.



Intel's SGX

Ideas

- Enables applications to run with confidentiality and integrity in the native OS environment.
- Reduces amount of trust application developers have to place on client platforms.

Intel's SGX

Ideas

- Enables applications to run with confidentiality and integrity in the native OS environment.
- Reduces amount of trust application developers have to place on client platforms.

Mechanisms

- Allows the creation of isolated containers for code execution (enclaves).
- Contents cannot change after initialization.
- Achieved through hardware-specific instructions.
- Messages produced within an enclave are authenticated and bound to its contents.

SGX operation	Purpose
ECREATE	Initialize
EADD	Initialize
EXTEND	Initialize
EINIT	Initialize
EENTER	Execute
ERESUME	Execute
EEXIT	Execute
EGETKEY	Crypto
EREPORT	Crypto
EBLOCK	Management
EREMOVE	Management
ETRACK	Management
ELDB	Management
ELDU	Management
EPA	Management
EWB	Management

SGX - Security

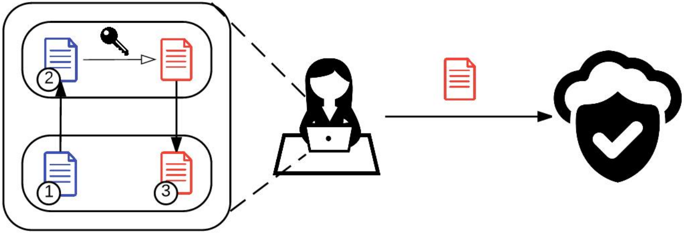
Authentication mechanism

- SGX provides code inside enclave with authenticity “proofs”.
- Micro-processor maintains one cryptographic key for each enclave.
- Requests for authentication “proofs” are performed using hardware specific instructions.
- Only a legitimate enclave can request a message authenticated with the key of another legitimate enclave.
- Authentication is performed using a cryptographic MAC, and can be used for intra-platform authentication.

SGX - From local to remote

A bit tricky

1. The enclave generates a cryptographic MAC.
2. Then sends its information with the MAC to a special enclave, to verify and produce a quote.
3. This quote contains a digital signature produced by a key only accessible via the special enclave. It can now be used for inter-platform authentication.



SGX - Applications

- White paper proposing solutions for one-time passwords, rights management and secure video conferencing [HLP+13].
- A distributed framework for map-reduce [SCF+14].
- The whole OS as an enclave [BPH14].

Motivation

Context

- Promising results arise from using SGX in practical applications.
- However, security implications are either unclear, or very specific to the different proposals.
- Isolated execution environments (IEE) are[1] not yet formalized from a cryptographic perspective.

[1] To the best of our knowledge

Motivation

Context

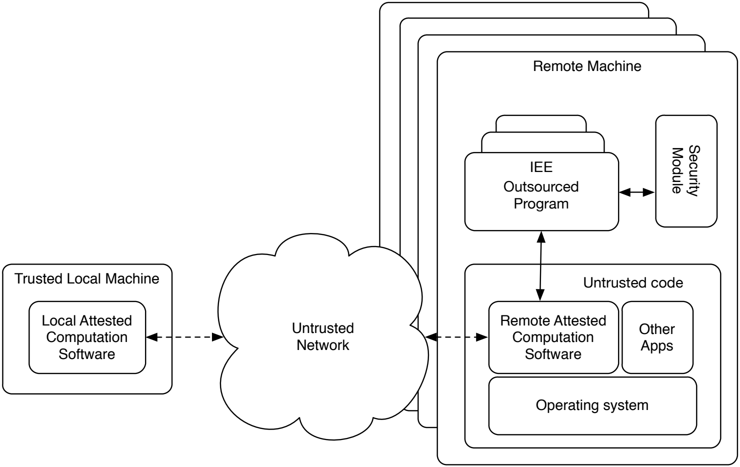
- Promising results arise from using SGX in practical applications.
- However, security implications are either unclear, or very specific to the different proposals.
- Isolated execution environments (IEE) are[1] not yet formalized from a cryptographic perspective.

Objectives

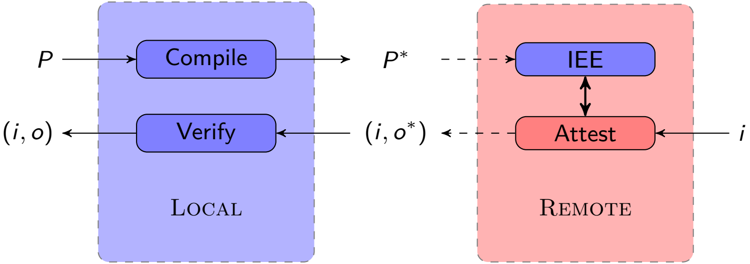
- Formalize the usage of IEEs: Attested Computation (AC).
- Propose a notion of key exchange for/over AC.
- Use this to get Secure Outsourced Computation.

[1] To the best of our knowledge

Modeling IEEs



Attested Computation



Security

1. Local view of trace is a trace of P
2. There exists an IEE executing P^* that has this trace

Implementing AC

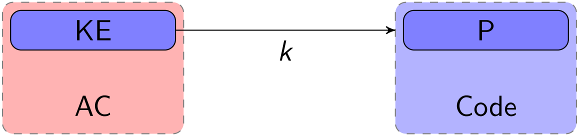
IEE provides: P^* is executing in an IEE and produced output x

P^*: adds a record of the trace to outputs of P and certifies using IEE

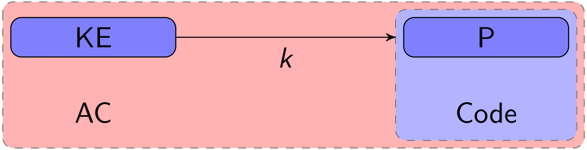
Verifying: check certificate and trace consistency

Composition?

We want

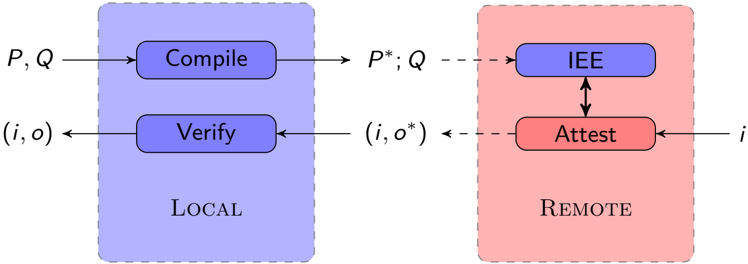


We can only do



Solution: AC definition with built-in composition

Composable AC



Properties

- Q is executed as is in IEE
- Attestation for P

Minimal leakage

Problem

The semantics of P does not guarantee anything on the semantics of P^*.

Goal

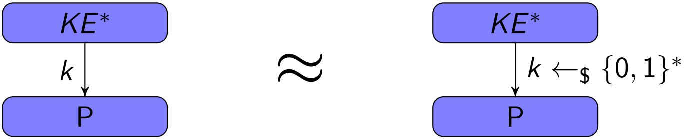
Ensure that internal values are not leaked; simulate execution without accessing internal values

$$\exists \mathcal{S}. \quad \mathcal{S}[T(P)] \approx P^*$$

(and trace is consistent)

Key exchange utility

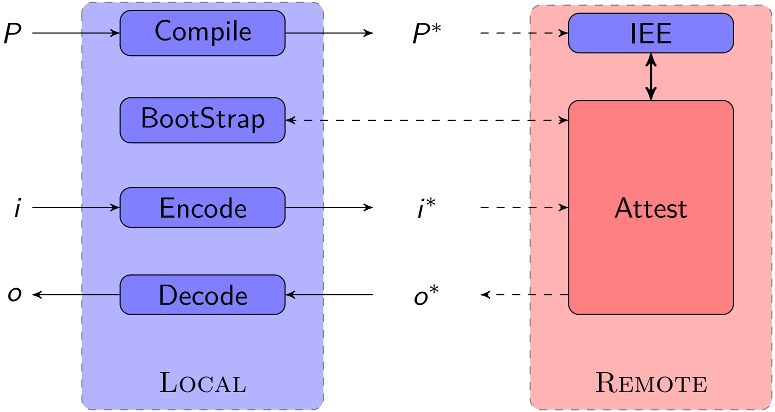
If KE is *passively secure*, AC secure, minimal leakage:



Intuition:

- Use AC to ensure that trace is valid
- Use minimal leakage to remove compilation
- Use passive security to replace key

Secure Outsourced Computation



Security

- Secrecy of I/O
- Authenticty of inputs

Conclusion

- A reusable notion of AC security
- A simple notion of AttKE and utility
- A way to achieve SOC

Strong points: modularity, relatively simple proofs, besides AC not tied to a particular platform
Interesting points: built-in composability, leakage

Next steps

- Put the toolbox to the test.
- Broaden the scope (multi-party computation).

A Cryptographic Treatment of Software Guard Extensions

M. Barbosa **B. Portela** G. Scerri B. Warinschi

InfoBlender – 1st of July 2015

- Andrew Baumann, Marcus Peinado, and Galen Hunt.
  Shielding applications from an untrusted cloud with haven.
  In *USENIX Symposium on Operating Systems Design and Implementation (OSDI)*, 2014.

- Matthew Hoekstra, Reshma Lal, Pradeep Pappachan, Vinay Phegade, and Juan Del Cuvillo.
  Using innovative instructions to create trustworthy software solutions.
  In *Proceedings of the 2nd International Workshop on Hardware and Architectural Support for Security and Privacy, HASP*, volume 13, 2013.

- Felix Schuster, Manuel Costa, Cedric Fournet, Christos Gkantsidis, Marcus Peinado, Gloria Mainar Ruiz, and Mark Russinovich.
  Vc 3: Trustworthy data analytics in the cloud.
  In *Proceedings of the 36th IEEE Symposium on Security and Privacy, SP*, volume 15, 2014.