The Attestable Model: properties and API

Carlos Molina-Jimenez and Jon Crowcroft
Department of Computer Science and Technology, University of Cambridge
carlos.molina, jon.crowcroft @cl.cam.ac.uk

5th October 2023

Abstract

An attestable is an abstract model of a computational environment. This document, defines its API and the security properties that an attestable needs to satisfy and its interface to interact with the applications hat use it. The focus of the discussion is on attestables that the CAMB project is currently implementing as a proof of concept and to be used in the implementation of a fair exchange protocol.

1 Introduction

The implementation of some applications can be significantly simplified by the inclusion of components that are capable of executing programs under the notion of **independent computation**. An independent computation is an execution environment, that is, a component where some operations are executed. It salient feature is that once the program is launched inside this component to compute some data, the development of the computation cannot be influences by anybody, including the owner of the component. The program runs independently until it reaches its final state, crashes or forever if it reaches an infinite loop.

Components that can provide independent computation can be implemented using different approaches and technologies. The aim of the CAMB project is to implement them on a Morello Board and use it for supporting application that are sensible to exfiltration of information, such as the FEWD (Fair Exchange Without Disputes) protocol. Comprehensive information about FEWD and advantages of using attestables in its implementation can be found in [2], in particular, in chapters 6 and 8.

To this end, we have devise an abstract model of the element with independent computation that we need in the CAMB project. The model specifies its physical and deployment properties and well its API. We call it an **attestable** to

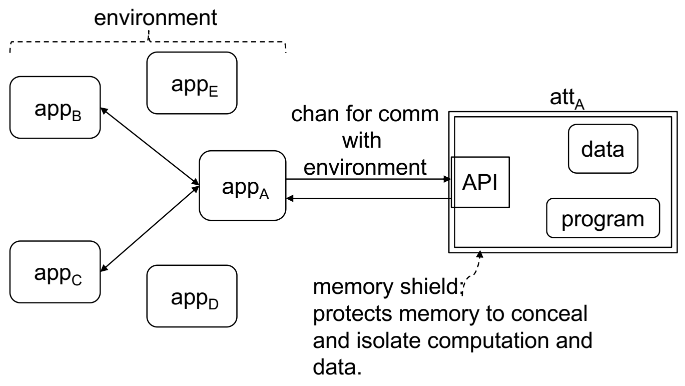
environment
app$_E$
app$_B$
chan for comm with environment
att$_A$
data
API
app$_A$
program
app$_C$
app$_D$
memory shield; protects memory to conceal and isolate computation and data.

Figure 1: Abstract view of an attestable and its environment.

convey the idea that the properties that this execution environment boast can be attested the parties interested in interacting with it.

Fig. 1 shows an attestable and its surrounding environment.

In our model, an attestable is owned by a user, for example, Alice. In the figure, att_A is an attestable owned by Alice. The owner has physical control over her attestable. Alice can switch it on and off or even worse, physically destroy it. Since has physical control over the component, she can control and manipulate the communication channel that the attestable uses to communicate with the environment. She can delay or delete the messages that the attestable sends and receives and even, worst, disconnect it.

We envision two potential interaction models between the attestable and its environment. In one shows in figure the attestable is logically located behind a single application controlled by the attestable's owner. In the example of the figure, Alice's attestable att_A is deployed behind Alice's application app_A. Alice's application is the only component that can interact with the attestable API and operates as the only bridge that connects the attestable to the environment. Applications interested in interacting with att_A, such as app_B and app_C, rely on app_A forward services. app_A receives and forwards messages, but it can also receive and drop them accidentally or maliciously.

Another potential interaction model is to release communication constraints to allow any application (app_B, app_C, app_D, app_E, etc.) to interact directly with the API of Alice's attestable. Under this model, the attestable can work more independently but it becomes more complex. For example, it needs to deploy standard Internet protocols such as RESTful and and SSL/TLS. However, its exposure to the whole external world might be detrimental to the properties that the component is expected to satisfy.

We believe that its is better to keep the desirable to keep API simple, therefore we focus on the interaction model shown in Fig. 1. We will describe the operations that the API of the attestable shown in the figure need to support.

2 The attestable security guarantees

1. An attestable executes the program and data loaded by its owner. It has the security mechanisms that makes it trustworthy.
2. An attestable is a black box that conceals its internal information (current state and data). It does not leak information. It releases information only as programmed and only through its API. In the figure, the protection wall is represented the double line.
3. The owner controls the attestable's interaction: API input and output messages. However the owner's application does not necessarily has access to the actual content of messages because they are normally encrypted under keys unknown to the owner's application, to app_A, in the figure.
4. Applications interested in interacting with the attestable can attest the potential behaviour of the attestable, that its, its properties.

   Attestation involves the verification of the software (program and data) loaded to the attestable before launching. Some applications might need verification of the attestable's current state.

3 Attestable's API

Attestables need to support three basic operations:

- **load**
  - **input:** A software, including a program, data and its configuration parameters.
  - **output:** The hash of the loaded software. Internally, this operation wipes the memory allocated to the attestable before loading the software.
- **attest**
  - **input:** Nonce
  - **output:** A message signed by the attestable under a private key, containing the hash of the loaded software and cryptographic information: chain certificate to verify the authenticity of the attestable and ephimeral public key to be used for further interactions with the attestable.
- **process**

– **input:** Encrypted message containing input data, program hash, public key of the attestable and public key of the remote party.

– **output:** Error if decryption fails or the program hash or attestable's public key do not match. Otherwise the attestable executes the program on the input data and returns the result: a message encrypted under the remote publickey containing the program's output.

4 Attestables implementation

The attestable described aboveis a model that can be implemented using different technologies.

For example, it can be implemented commercial computers with special hardware support like Intel SGX enclaves, a tamper–proof chip on a mobile phone like current JavaScrypt cards and crucial to the CAMB project, on Morello Boards equipped with cheri–capabilities.

Incidentally the attestable shown in Fig. 1 can be compartment created with cheri–capabilities running on a Morello Board.

5 Work in progress and challenges

The implementation of attestables with the properties discussed in Section 2, on the current version of the MB presents several challenges. To implement the memory shield shown in Fig. 1 one needs to devise mechanisms to grant the attestable exclusive access to its memory. That is, mechanisms that to implement mutual distrust between the attestable and the privilege software such as the kernel. In essence we need to devise software and hardware mechanisms similar to those provided by Intel SGX to implement enclaves[3].

Another challenge is to device mechanism for remote attestation without root of trust built it the MB. The MB because was released to test their capabilities not for end-to-end encryption or secure remote access [1]. The MB includes mechanisms for secure booting, however, it does not include a Trusted Platform Module (TPM) or EPID (Enhanced Privacy ID) key that can assist in remote attestation. The approach that we are taking is to firstly figure out the remote attestation API that the cheri–based attestable on MB needs to support and next figure out how the API can be implemented with existing mechanisms, for example, with support external hardware like ARM trustzone.

References

[1] Dbrooke. *Is there a Trusted Platform Module (TPM) on the Morello board?* `https://community.arm.com/support-forums/f/morello-forum/`

49176/is-there-a-trusted-platform-module-tpm-on-the-morello-board/. visited on 17 Sep 2023. 2021.

[2] Carlos Molina–Jimenez et al. *Fair Exchange: Theory and Practice of Digital Belongings*. In press, to appear in 2023. World Scientific, 2023.

[3] Victor Costan and Srinivas Devadas. *Intel SGX Explained*. https://eprint.iacr.org/2016/086.pdf. 2016.