JOHANNES KEPLER
UNIVERSITY LINZ

Submitted by
Michael Preisach, BSc
1155264

Submitted at
Institute for Networks and Security

Supervisor and First Examiner
Univ.-Prof. DI Dr. **René Mayrhofer**

Second Examiner
DI **Tobias Höller**

May 20, 2020

Project Digidow: Biometric Sensor

Master Thesis

to obtain the academic degree of

Master of Science

in the Master's Program

Computer Science

JOHANNES KEPLER UNIVERSITY LINZ
Altenbergerstraße 69
4040 Linz, Österreich
www.jku.at
DVR 0093696

JOHANNES KEPLER
UNIVERSITY LINZ

Abstract

What is it all about? Why is that interesting? What is new in this thesis? Where is the solution directing to?

Zusammenfassung

Das am Institut für Netzwerke und Sicherheit entwickelte Projekt *Digital Shadow* benötigt in vielen Bereichen ein prüfbares Vertrauen um eine Erkennung von Nutzern anhand ihrer biometrischen Daten zu erkennen und Berechtigungen zuzuteilen. Das Vertrauen soll dem Nutzer die Möglichkeit geben, die Korrektheit des Systems schnell und einfach zu prüfen, bevor er/sie disesm System biometrische Daten zur Verfügung stellt Diese Masterarbeit beschäftigt sich nun mit den existierenden Werkzeugen, die ein solches Vertrauen schaffen können. Das implementierte System kombiniert diese Werkzeuge, um damit sensible Daten von Nutzern aufzunehmen und im Netzwerk von Digital Shadow zu identifizieren. Es soll dabei sicher gestellt sein, dass eine fälschliche Verwendung der sensiblen Nutzerdaten ausgeschlossen wird. Anhand dieses Systems werden die Eigenschaften einer vertrauenswürdigen Umgebung für Software diskutiert und notwendige Rahmenbedingungen erläutert.

Contents

1 Introduction

We all live in a world full of digital systems. They appear as PCs, notebooks, cellular phones or embedded devices. Especially the footprint of embedded computers became so small that they can be used in almost all elctrical devices. This product category form the so called *smart* devices.

With all these new devices a lot of societal problems could be solved in the past few decades. Many of them automate services to the public like managing the bank account, public transportation or health services. There is an endless list of services that can be done by a computer.

The downside of all these digital services is that using these services generate a lot of data. Besides of the intended exchange of information, many of the services try to extract metadata as well. Which IP is connected? What kind of device is that? Is the software up to date? Was this device here in the past? Which other sites did the user browse? This is an endless list of questions which can be answered with a set of metadata. And all this data is collected when users browse the Internet. At the end the user may not be charged financially but one pay with this metadata. The customer becomes the product.

However when a project is financed by the public, it should be possible to show users that there is a difference in the usage. It should be possible to prove that an application or a computer system is honest to the user. People should be convinced of this honesty and build trust in using this kind of software.

1.1 introduction in project digidow

The Project *Digital Shadow* is under ongoing developüment at the Institute of Networks and Security and creates a scalable system for authentication. Key feature is privacy by design and a provable system to create trust to the end user.

At this early stage the interfaces and interaction points are not fully defined.

This is a brief description of the process of authentication:

1.2 Biometric Sensor use case in DigiDow

derive the use case of the Biometric sensor out of the above model.

1.3 Goals and Definitions

You should be able to attach a variety of sensors to the system. The system should then fulfill the followin requirements

- *Sensor Monitoring.* The System should be able to monitor the sensor itself.
- *System Monitoring.* It should be possible to track the state of the system. Especially every modification of the system should be detected.
- *Freshness of Sensor Data.* To prevent replay attacks, the system should proof that the provided biometrc data is captured live.
- *Integrity of Sensor Data.* As it is possible for an attacker to modify the provided data during the capturing process, integrity should guarantee that the data comes from the sensor in an unmodified manner.
- *Confidentiality of Sensor Data.* It should not be possible to eavesdrop any sensitive data out of the system. Furthermore almost all kinds of metadata (e. g. information about the system or network information) should not be published
- Usage Model of Biometric Sensor

This thesis will describe a system, which is part of the Digital Shadow network. Therefore it has to meet the common principles in information security, namely:

- *Availability*:
- *Integrity*: ISO 27000 (Data Integrity)
- *Confidentiality*: ISO 27000

Upon AIC it should be possible for users to prove honesty of the system. This is what *trust* defines in information security

1.3.1 Requirements

- given a set of software, this system should provide information that exaclty this version of software is running on the system. (Integrity)
- The system must furthermore show that it is a member of valid biometric sensors (Attestation)
- All the given information should be anonymized. It should not be possible to gain any other information about the system (Anonymity)
- It should be ensured that no sensitive data is stored at the biometric sensor

Scope of this thesis is on implementing the system from from hardware to application layer. Is is not supposed to think about the network communication.

1.4 Description of structure

1. What exists out there?
2. What is the theoretical solution
3. What about the implementations used - what is the limitation of the used tools?
4. How far are we? what has to be considered next?

2 Related Work

- What exists in the field?
- Keylime
- Xaptum ECDAA
- FIDO 2 ECDAA
- Strongswan Attestation
- Linux IMA
- Secure Boot
- Intel TXT
- Trusted Execution Environment (TEE)
- nanovm (nanovms.com)

3 Concept

The theoretical tool that should be formed to one whole system implementation in this thesis.

3.1 Definition of the Biometric Sensor

What part fulfills the BS and what needs to be done. Record Sensor data, Network Discovery, send sensor data via trusted channel to PIA

3.1.1 What has the BS to do?

1. Listen for a Trigger to start the Authentication Process
2. Collect Sensor Data (Picture, Fingerprint) and calculate a biometric representation
3. Start Network Discovery and find the PIA of this person
4. Create a trusted and secure channel and transmit the attributes for verification
5. Restore the state of the system as it was before this transaction

3.2 Attack Vectors and Threat Model

3.2.1 The Threat Model

- Definition of sensitive data / privacy / metadata
- This version of BS is not owned by the user, there is no personal data in the System

- Rogue Personal Identity Agent (PIA)
- Metadata Extraction
- Attribute extraction
- Sensor Data Modification/manipulation
- Wiretap between Sensor and System (USB or network)
- Physical Manipulation of the BS-System
- Network - Retransmission of sensor data of a rogue BS
- Network - Blocking Data transmission of a rogue BS
- Rogue BS Sensor Data aggregation
- Rogue BS Sensor data modification before transmission

3.3 Trust and Security

Trust is an essential term in this thesis. In the world of IT security, the term *trusted computing* defines a secured environment where special or confidential computing jobs are dispatched. This environment or product usually meets the following requirements

- *Minimalization.* The number of features and hence the complexity must be as low as possible.
- *Sound definitions.* Every function should be well defined. There should be no margin for interpretation left. Security Engineers should be involved in the development.
- *Complete testing.* Testing for trusted computing includes a threat analysis and exhaustive testing if possible.

Since software and hardware testing is never complete, it is hard to find a good balance between feature set and testing completeness.

However trust in IT is not equal to security. It defines a subset of IT security where this small well defined environment is embedded in a larger system which is usually untrusted. Claiming a system *secure* spans the constraints of trust over the complete system, which is not affordable for commodity computers these days. However it is possible to use the trusted environment to get some guarantees on the untrusted parts of a system as well In Chapter 3 we will show how trust will be extended in a commodity PC.

Differentiation between trust and security — and the problem that not everyone is using that right.

3.4 Systems of Trust

All trust systems are built on the standards of Trusted Computing Group.

3.4.1 Secure Boot, TXT, ...

Trusted Boot is not the same as Secure Boot. Explain the difference

3.4.2 TPM1.2

Initial Version of the crypto-coprocessor, successfully spread into many systems, but hardly any integration in Trust/security Software

3.4.3 TPM2.0

The *Trusted Platform Module* (TPM) is a small cryptocoprocessor that introduces a variety of new features to the platform. This module is part of a standard developed by the Trusted Computing Group (TCG), which current revision is 2.0[4].

The hardware itself is strongly defined by the standard and comes in the following flavors:

- *Dedicated device.* The TPM chip is mounted on a small board with a connector. The user can plug it into a compatible compute platform. This gives most control to the end user since it is easy to disable trusted computing or switch to another TPM.
- *Mounted device.* The dedicated chip is directly mounted on the target mainboard. Therefore any hardware modification is impossible. However most PC platforms provide BIOS features to control the TPM.

- *Firmware TPM (fTPM).* This variant was introduced with the TPM2.0 Revision. Firmware means in this context an extension of the CPU instruction set which provides the features of a TPM. Both Intel and AMD provide this extension for their platforms for several years now. When activating this feature on BIOS level, all features of Trusted Computing are available to the user.
- *TPM Simulator.* For testing reasons, it is possible to install a TPM simulator. It provides basically every feature of a TPM but cannot be used outside the operating system. Features like Trusted Boot or in hardware persisted keys are not available.

Even the dedicated devices are small microcontrollers that run the TPM features in software which gives the manufacturer the possibility to update their TPMs in the field. FTPMs will be updated with the Microcode updates of the CPU manufacturers.

The combination of well constrained hardware and features, an interface for updates and well defined software interfaces make TPMs trustworthy and reliable. Since TCG published the new standard in 2014 only six CVE-Entries handled vulnerabilities with TPMs[1]. Only two of them had impact on the implementation of a dedicated chip:

- *CVE-2017-15361*

Using the TPM

On top of the cryptographic hardware, the TCG provides several software interfaces for application developers:

- *System API (SAPI).* The SAPI is a basic API where the developer has to handle the resources within the application. However this API provides the full set of features.
- *Enhanced System API (ESAPI).* While still providing a complete feature set, the ESAPI makes some resources transparent to the application like session handling. Consequently, this API layer is built on top of the SAPI.
- *Feature API (FAPI).* This API layer is again built on top of the ESAPI. It provides a simple to use API but the feature set is also reduced to common use cases. Although the Interface was formally published from the beginning, an implementation is available since end of 2019.

[1]https://cve.mitre.org/cgi-bin/cvekey.cgi?keyword=%22Trusted+Platform+Module%22

The reference implementation of these APIs is published at Github[2] and is still under development. At the point of writing stable interfaces are available for C and C++, but other languages like Rust, Java, C# and others will be served in the future. The repository additionally provides the tpm2-tools toolset which provides the FAPI features to the command line. Unfortunately, the command line parameters changed several times during the major releases of tpm2-tools[3].

The Hardware

The TCG achieved with the previous mentioned software layers independence of the underlying hardware. Hence, TCG provided different flavors of of the TPM

TCG defined with the TPM2.0 standard a highly constrained hardware with a small feature set. It is a passive device with some volatile and non-volatile memory, which provides hardware acceleration for a small number of crypto algorithms. The standard allows to add some extra functionality to the device. However the TPMs used in this project provided just the minimal set of algorithms and also the minimal amount of memory.

Since TCG published its documents, several IT security teams investigated concept and implementations of TPMs.

- security problems with some implementations
- Hierarchies
- Endorsement Key
- Attestation Identity Key
- Key management

3.5 Integrity Measurements

Extend the Chain of Trust beyond the boot process. The Kernel can measure many different types of Resources. What is a useful set of measurements

3.6 Verify Trust (DA and DAA)

Use the TPM to proof trustworthiness to other instances like the PIA

4 Implementation

4.1 Trusted Boot

- Trusted Boot with GRUB 2.04: TPM support available; PCR mapping
- Secure Boot with Unified Kernel; another PCR mapping
- Benefits and Drawbacks of both variants

Limitations due to bad implementation on BIOS-Level, no Certificate Verification Infrastructure available for TPMs? Needs to be proven for correctness.

4.2 Integrity Measurement Architecture

Available on Ubuntu, RedHat and optionally Gentoo. The Kernel has the correct compile options set.

4.2.1 Handling external hardware

How can camera and fingerprint sensor be trusted? What is the limitation of this solution?

4.3 Interaction with TPM2

tpm2-tools 4.x are usable to interact with the TPM from the command line. Available on all major releases after summer 2019. Fallback is using the TPM2 ESAPI or SAPI, which is available on almost all Linux distributions.

4.4 Direct Anonymous Attestation

DAA Project from Xaptum: Working DAA handshakt and possible TPM integration. Requires an Attestation Key which is secured with a password policy.

5 Conclusion and Outlook

5.1 Testing

These are the test results

5.2 Limitations

Still hard to set up a system like that. Documentation is available, but hardly any implementations for DAA and IMA.

5.3 Outlook

Hardening of the system beyond IMA useful. Minimization also useful, because the logging gets shorter.

Table 5.1 is an example of a table, in which the numbers are aligned at the comma, every second line is colored and the commands `\toprule`, `\midrule` and `\bottomrule` are used [1].

Table 5.1: Example

Länge l in m	Breite b in m	Höhe h in m
12.454	1.24	335.3
543.22	32.123	33.21
353.0	33.0	33.0
23.3	333.2	32.4

List of Figures

List of Tables

Bibliography

[1] Will Arthur, David Challener, and Kenneth Goldman. *A Practical Guide to TPM 2.0.* Jan. 2015. doi: 10.1007/978-1-4302-6584-9.

[2] TPM2 Software Community. *TPM2 Tools.* 2020. url: https://github.com/tpm2-software/tpm2-tools (visited on 05/15/2020).

[3] Pawit Pornkitprasan. *Its certainly annoying that TPM2-Tools like to change their command line parameters.* Oct. 2019. url: https://medium.com/@pawitp/its-certainly-annoying-that-tpm2-tools-like-to-change-their-command-line-parameters-d5d0f4351206 (visited on 02/27/2020).

[4] *The TPM Library Specification.* 2019. url: https://trustedcomputinggroup.org/resource/tpm-library-specification/ (visited on 05/16/2020).

Installation instructions

1 Installing IMA on Arch

https://wiki.archlinux.org/index.php/Kernel/Arch_Build_System in combination with https://wiki.gentoo.org/wiki/Integrity_Measurement_Architecture:

```
sudo pacman -S asp base-devel
cd ~
mkdir build && cd build
asp update linux
asp export linux #Linux repo exported to this directory
```

Change *pkgbase* in `linux/PKGBUILD` to custom name, e.g. linux-ima. Check `linux/config` for the following settings:

```
CONFIG_INTEGRITY=y
CONFIG_IMA=y
CONFIG_IMA_MEASURE_PCR_IDX=10
CONFIG_IMA_LSM_RULES=y
CONFIG_INTEGRITY_SIGNATURE=y
CONFIG_IMA_APPRAISE=y
IMA_APPRAISE_BOOTPARAM=y
```

For optimizing file access, add to every fstab-entry *iversion*. It prevents creating a hash of the file at every access. Instead the hash will only be created when writing the file.

`updpkgsums` generates new checksums for the modified files.

`makepkg -s` then makes the new kernel

`SSllMMxxii` Hallowelt