HAL open science

Implementing Secure Applications thanks to an Integrated Secure Element

Sylvain Guilley, Michel Le Rolland, Damien Quenson

▶ **To cite this version:**

Sylvain Guilley, Michel Le Rolland, Damien Quenson. Implementing Secure Applications thanks to an Integrated Secure Element. 7th International Conference on Information Systems Security and Privacy, INSTICC, Feb 2021, Vienne (en ligne), Austria. pp.566-571, 10.5220/0010298205660571 . hal-03084250v3

HAL Id: hal-03084250

https://telecom-paris.hal.science/hal-03084250v3

Submitted on 14 Feb 2021

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Distributed under a Creative Commons Attribution 4.0 International License

Implementing Secure Applications thanks to an Integrated Secure Element

Sylvain Guilley[1,2,3][a], Michel Le Rolland[4] and Damien Quenson[4]

[1] *Secure-IC S.A.S., Tour Montparnasse, 27th floor, 75015 Paris,* FRANCE.

[2] *TELECOM-Paris, Institut Polytechnique de Paris, 19 Place Marguerite Perey, 91120 Palaiseau,* FRANCE.

[3] *École Normale Supérieure, 45 rue d'Ulm, Département d'informatique, CNRS, PSL University, 75005 Paris,* FRANCE.

[4] *Secure-IC S.A.S. Headquarters, 15 rue Claude Chappe, Bât. B, 35510 Cesson-Sévigné,* FRANCE.

{sylvain.guilley, michel.lerolland, damien.quenson}@secure-ic.com

Keywords: Integrated Secure Element · Host Protection · Certification Schemes · Security Flexibility · Internet of Things.

Abstract: More and more networked applications require security, with keys managed at the end-point. However, traditional Secure Elements have not been designed to be connected. There is thus a need to bridge the gap, and novel kinds of Secure Elements have been introduced in this respect.

Connectivity has made it possible for a single chip to implement multiple usages. For instance, in a smartphone, security is about preventing the device from being rooted, but also about enabling user's online privacy. Therefore, Secure Elements shall be compatible with multiple requirements for various vertical markets (e.g., payment, contents protection, automotive, etc.). The solution to this versatility is the integration of the Secure Element within the device main chip. Such approach, termed iSE (integrated Secure Element), consists in the implementation of a subsystem, endowed to manage the chip security, within a host System-on-Chip.

The iSE offers flexibility in the security deployment. However, natural questions that arise are: how to program security applications using an iSE? How to certify those applications, most likely according to several different schemes?

This position paper addresses those questions, and comes up with some key concepts of on-chip security, in terms of iSE secure usage. In particular, we will show in this paper that iSE nowadays shall be designed so that the product it embeds is certifiable in a multiplicity of schemes, and so even before the product is launched on the market.

1 Introduction

We are witnessing the rapid development of our digital society, which is characterized by the explosion of the number of handheld and IoT devices. In this context, the security of the end-points is essential, as those objects represent an obvious entry point for attackers into the whole platform. Actually, the end-points are nowadays entrusted to secure the link starting from the device up to the cloud. Such end-to-end security is achieved by designing a solid root-of-trust which stems from the device. It is the role of the "Secure Element" to implement such fundamental security foundation.

From a historical perspective, the Secure Element has known several successive development stages:

- It was initially discrete, and designed mainly to defend themselves (i.e., it features anti-tampering capabilities). It was also removable, and mostly dedicated to one user (namely the subscriber);
- Then, it has become more aware of the security of the application, and has thus been termed "embedded". It may consist in legacy discrete Secure Element, but adapted to smaller form factors and implemented closer to the circuit it protects. Being attached to the device, it is not removable and shall therefore accept with flexibility several use-cases, and is therefore suitable for machine-to-machine (M2M) contexts;
- Eventually, some Secure Elements have become integrated into the main chip of the end-point device. Consequently, it got melted as a subsystem within the host chip it protects.

The Secure Elements, under those three avatars, enforce end-point security principles. Namely, they implement:

- *Minimum surface exposition*: in that the interface of a Secure Element is reduced to as few pins and functions as possible. For instance, one can remember ISO/IEC 7816-2 standard (ISO/IEC 7816, 2014), published back in the late 80's,

[a] https://orcid.org/0000-0002-5044-3534

which required compliant smartcards to have only one single in/out (IO) pin for the communication with the external world, in half-duplex.

- *Defense in depth*: Secure Elements implement several layers of protection. For instance, the Secure Element is typically entrenched behind a defensive layer, be it physical (e.g., SPI or Serial line) or logical (e.g., protocol break through mailboxes).
- *Minimal complexity rationale*: the control logic in Secure Elements is crafted to be focused on its security mission, and is thus not polluted by interference with other businesses, which are left to the "rich world".

The proper implementation of those security principles is verified through a process known as "certification", whereby a third party laboratory assesses the product documentation and performs a counter-expertise in terms of offensive security testing.

In this position paper, we account for the raise of the integrated Secure Element (referred to as "iSE" in the sequel), and explain the challenges linked with its adoption. The argumentation is oriented towards explaining the stakes regarding security requirements, the technological evolution (connected devices, lifecycle management, etc.) and the moving normalization landscape. The targeted audience is the security solution architects aiming at gaining a pedagogical overview about end-point security trends.

Contributions. We list the advantages of the iSE over the traditional approach using a Secure Element located outside of the main end-point chip. Then, we discuss integration examples, for different representative applications. Most importantly, we provide a rationale for multiple simultaneous vertical market requirements, and account for a strategy for the chip hosting the iSE to get multiple certifications according to various different schemes. The key concept of the iSE "intellectual property" block pre-certification is introduced and we demonstrate its advantages over the state-of-the-art.

Outline. The rest of this article is structured as follows. The current trends in user's requirements are described and analyzed in Sec. 2. This section motivates for the relevance of iSE paradigm. The instantiation of iSE in different contexts is the topic of Sec. 3. This section shows that iSE architecture can be protean within the host chip, depending whether the host can be trusted or not. Finally, our conclusions and perspectives are in Sec. 4.

2 Current needs

2.1 Embedded versus integrated Secure Elements

Comparison. The figure 1 illustrates the layout difference between *embedded* and *integrated* Secure Elements. The left side of this figure shows the secure chip and its companion Secure Element, connected by a SPI (Serial Peripheral Interface) link. In the right side, one can see the iSE, as a security subsystem buried into the applicative chip (termed the "host").

The symbioticity of the iSE with its host provides more natural and better performing security application.

Since Secure Elements are off-the-shelf components, an attacker can easily analyze a discrete Secure Element in a "training mode" on some samples before launching his sharpened attack on the targeted product. At the opposite, the iSEs are more protected since the attacker shall first do some reverse-engineering to identify the iSE resources to target.

Embedded Secure Elements are stereotyped circuits, such as Trusted Platform Modules (TPM). Many such chips can be produced in large series, leading to economies of scale. But this gain comes with the fact that the electronic board is more complex, since there are two chips to integrate and solder. In addition, embedded Secure Elements have finite resources, and can be somehow restricted in their usage, e.g., lack memory. Moreover, although they can be certified, they typically are so only for one application.

integrated Secure Elements (iSEs) are customized for each host chip. This does not preclude some level of reuse, on the contrary, but the hardware of the iSE shall be recompiled. One advantage is that the iSE can be taylored to meet exactly the requirements of each application, in terms both of security level and performances. Regarding board-level integration, the iSE solution is cheaper since it does not add a dedicated chip (it is integrated into a System-on-Chip).

In addition, the iSE has at its disposal large amount of resources, as most of those can be shared with the host (such as various kinds of memories[1], connectivity, debug capabilities, etc.). Moreover, the iSE is more efficient since it can naturally interface

[1]For instance non-volatile FLASH, One-Time Programmable (OTP) or electric fuses, Random Access Memories (RAM), etc.

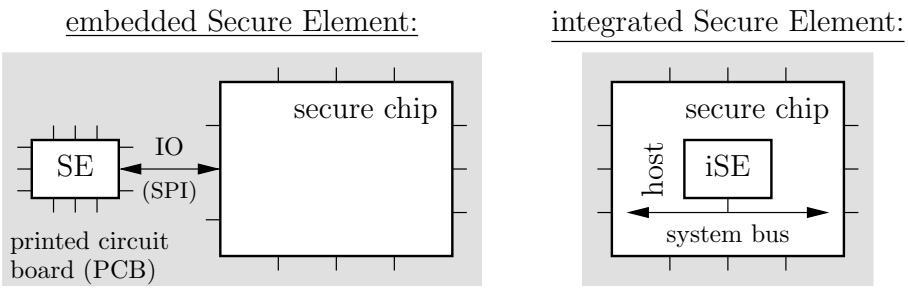

Figure 1: Comparison between embedded versus integrated Secure Element

with various applications (e.g., it is on the same die as the host applicative code it is responsible to check upon "secure-boot", the crypto services offered by the iSE can access the data to encrypt/sign/authenticate within one clock cycle, etc.). Reciprocally, the iSE can also help with the verification of further external resources, such as the configuration's authenticity verification of host clock/power module, memory management unit (MMU), debug JTAG port, etc. In a nutshell, the iSE is extraverted, in that it can directly protect its host and other resources. This "mutual win-win exchange" is illustrated in Fig. 2.

Regarding application finetuning, the iSE can have its code debugged directly within the host, which is an advantage for the ease of integrated chip prototyping (including host along with iSE, simultaneously).

2.2 Modern requirements

Beyond the form factor considerations, we now underline that there are more fundamental driving factors for iSE to be considered as the best option. One major compelling reason is that current chips are so complex that they are not specialized for only one application, but instead provide multiple services. For instance, one single chip can enable payment, access to digital contents, (private) social networks, etc. The security management of heterogeneous applications (so-called "apps"), each requiring a different security scheme, can rapidly become a nightmare with discrete Secure Elements, where it can be elegantly fulfilled with the iSE paradigm.

Even more, iSE is future-proof, in that new apps (with specific security requirements) can be installed after deployment. This stays true even if applications are not known nor foreseen upon chip design. This on-field flexibility is appreciable in the context of highly competitive market segments where investments on the chip are carried out ahead of full knowledge of market potentialities.

3 Use-cases with iSE

This section presents some typical integration patterns of a Secure Element in a security chip.

3.1 Programmer's model

The cornerstone of security management in a Secure Element is the concept initially forged by RSA Labs company in the standard PKCS #11 (OASIS, 2020). It consists in segregating security *objects* from *actions* which can be carried out on them. Namely, in PKCS #11, keys are kept within the Secure Element, and the user can only request actions on them through an Application Programming Interface (API). When implemented correctly (Delaune et al., 2008), it is proven that whatever the actions from the user side, keys cannot cross the border between the SE and the host. Correct implementation requires nonetheless some attention, such as carefully attributing roles to keys, otherwise API-level attacks can apply (Clulow, 2003).

Now in practice, there can be several ways to integrate the iSE within the host. Those variants depend on the perceived likelihood of host corruptibility. Namely, an illustration is provided in Fig. 3. The security application is represented as (for the code) and (for the secret data). Three typical application partitioning schemes are presented in this figure.

(a) When it is assumed that the host is likely to be compromised (but not the iSE), then actions are carried out on the Application Processor (AP), and keys are kept concealed within the iSE;

(b) When the protocol cannot easily be split between actions and objects, and that the host is assumed untrusted, then the whole secure applications resides within the iSE: actions are executed by the iSE processor;

(c) When the host is assumed to be incorruptible, then the iSE boundary can be dissolved: the keys are managed by the iSE and the actions (even

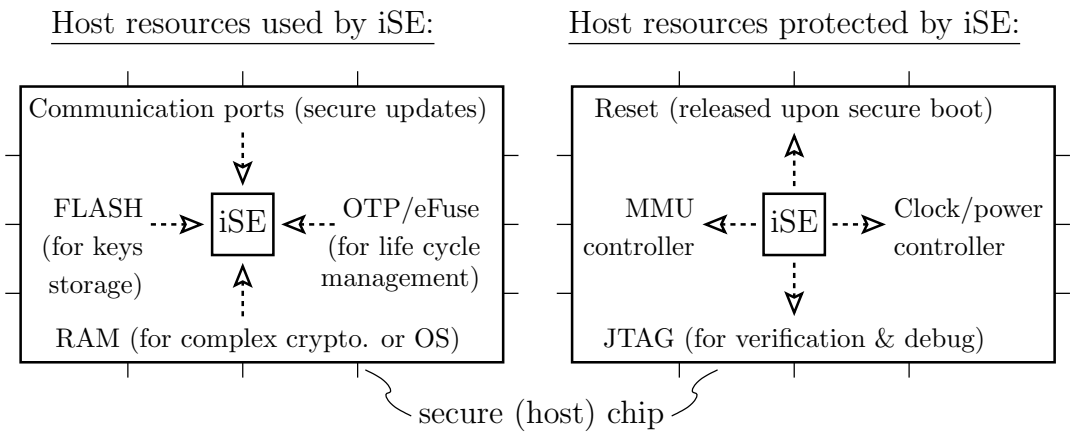

Figure 2: (left) Resources the iSE requires from the host to extend its functional capabilities; (right) Host resources the iSE can verify

complex, i.e., unproven in terms of harmlessness should the code be manipulated, e.g. by a bug exploitation or a malware) are performed by the host AP.

Notice that technologies such as Global Platform Trusted Execution Environment (TEE (Global Platform, 2020b)) can be seen as a virtualization of iSE. Indeed, TEE consists in an API which allows for a given host to behave two-fold:

- in a regular manner, termed "Rich Execution Environment", where untrusted apps can be run, and
- in a secure manner, termed "Trusted Execution Environment", with logical and/or physical segregation of resources, including memory, monotonic counters, timers, random number generators, etc.

The security of TEE technology is mature, as there is even a Protection Profile which formalizes it (GlobalPlatform Device Committee, 2016). Now, physical security is explicitly out of the scope of TEE, hence a TEE is not sufficient *per se* to play the role of a Secure Element. However, many practical attacks have been reported (Cerdeira et al., 2020; Novella, 2020). Side-channel attacks, and in particular cache-timing attacks, represent a large number of them.

3.2 Some examples of iSE usage

Some standards actually impose an iSE, such as Global Platform (GP) Virtual Primary Platform (VPP (Global Platform, 2018)) specification, as used for instance in European telecommunications standards institute (ETSI) Smart Secure Platform (SSP (ETSI, 2019)) standard. Also, in the following use-cases, we highlight in particular that different usages require different certification schemes.

Application: platform security (case (a)).

The host runs the application, and asks the iSE for key management and cryptographic services. Examples of applications are PKCS #11 (OASIS, 2020) (as in Hardware Security Modules) and TCG TPM (Trusted Computing Group, 2019). This use-case is adapted to an architecture corresponding to case (a) in Fig. 3.

Application: secure communication (case (a)).

In this application, the iSE is offloading the host for cryptographic computations; therefore, the iSE can be dissolved amidst the host, as in case (c) of Fig. 3. Such architecture is suitable for protocols such as EVITA (FP7 European Project, 2020), IEC 62443 (International Society of Automation (ISA, https://www.isa.org/), 2020), or Car2Car V2X PP (CAR 2 CAR Communication Consortium, 2019). Specifically, this latter reference clearly shows the paradigm shift from embedded (Figure 1) to integrated (Figure 2) Secure Element, called "HSM" for "Hardware Security Module" in automotive market parlance.

Application: iSIM (integrated SIM, i.e., integrated Subscriber Identity Module) (case (b)).

The iSE is the enclave. It ensures the secure execution of the application, such as GP VPP (Global Platform, 2018) or ETSI SSP (ETSI, 2019). The case (b) of Fig. 3 is suitable for this use-case. Typical security requirements are enunciated in Common Criteria (Consortium, 2013) Protection Profile PP0084 (Bundesamt für Sicherheit in der Informationstechnik, 2014).

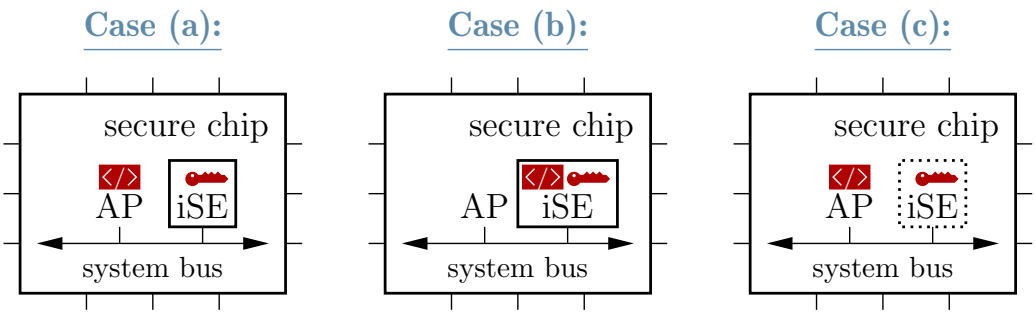

Figure 3: Different instantiations of an iSE within an host chip. Cases (a) and (b) are suitable when the host can be subverted (e.g., it can be "rooted"); Case (a) works for simple applications where the specification is designed to gracefully split control and data, whereas (b) is used for complex applications, e.g., which require an Operating System (OS). Case (c) is the natural solution for those chips which are shielded from the outside world, hence protected by design from remote infection.

Since IoT devices cannot manage discrete SIM cards, the market is shifting to this iSE paradigm.

Application: Root-of-Trust (e.g., Open Compute Project (OCP Security, 2020), Open Titan (Open Titan, 2020)) (case (c)).

Here, the iSE is responsible for booting an external host, which is typically a server in a Cloud infrastructure. As the root-of-trust is standalone in its chip, architectural case (c) of Fig. 3 is also sufficient. There is no security standard in place for now for this segment.

3.3 How to ensure that the device is pre-certifiable?

As one can see, there are multiple certification schemes. It might happen that one of the certification is required late in the product life cycle, e.g., after that the product is manufactured. We define the term "pre-certification" as the action to anticipate a future certification. This concept is disruptive as, traditionally, certification was a necessary milestone to enter a market. However, nowadays, the context is different for several reasons:

- Some chips are so versatile that they can be deployed in various market verticals, e.g., IoT, industry, telecommunication stations, etc. which are each governed by different security standards.
- Obtaining consensus on security regulation can take a long time. For instance, Global Platform SESIP (Global Platform, 2020a) certification has been proposed by the industry to remedy for the lack of relevant international normative requirements in the field of IoT.
- Users sometimes adopt technologies before the security model is completely understood; this is for instance the case with IoT, massively deployed as smart home assistants, smart assistants, wearable health trackers, etc. (see (ETSI, TC CYBER, 2019, §1)).

Such proactive attitude in being agnostic to security specific requirements demands that some precautions be taken, amongst which:

- Preparation of a clear documentation, including user manuals and test plans;
- Sound design decisions, e.g., implementation of roles, of error management, etc.
- Functional features like keys zeroization (cf. NIST FIPS 140 (NIST FIPS, 2009)) shall be supported, etc.

In addition, pre-silicon actions are necessary. For instance, using a Physically Unclonable Function (Bruneau et al., 2018) to implement the master key is a bullet-proof security-by-design presilicon choice. Also, pre-silicon security evaluation (Souissi et al., 2019) allows to attest of resistance against attacks even before product production stage. Some Common Criteria schemes (typically the NSCIB in the Netherlands) have already started to evaluate virtual products, i.e., their source code, before it is implemented.

Clearly, adjustments in the design can be fine-tuned at software level, and documentation might need some elaboration, but those are minor changes if the underlying precautions have been respected.

4 Conclusions and perspectives

The integrated Secure Element (iSE) is emerging as a novel paradigm for ensuring the end-point security. The iSE brings flexibility in the security management, in that iSE can support various security applications. Moreover, by being instantiated directly within

the host chip, it can protect several resources (e.g., clock, reset, MMU, debug, etc.), and can be reconfigured, hence it is future-proof. This highlights the need for pre-certified iSE, as an asset for the chip to comply with different certifications, even before being launched on different markets.

Leveraging the iSE concept, further security applications can be envisioned. For instance, the iSE can behave actively as a "cybersecurity probe" within the device, responsible for reporting security events. This opens the door for end-point devices used as sentinels for gathering security threats at system-level. With the capability to garner intelligence this way, security analytics can be carried out. This allows for the security operator to understand very precisely the attacks perpetrated on the chip. Such information enables him to proactively anticipate them, thereby reversing the advantage in favor of the defense side.

ACKNOWLEDGEMENTS

This work has been partly funded by the ARCHISEC project (number ANR-19-CE39-0008).

REFERENCES

Bruneau, N., Danger, J., Facon, A., Guilley, S., Hamaguchi, S., Hori, Y., Kang, Y., and Schaub, A. (2018). Development of the Unified Security Requirements of PUFs During the Standardization Process. In *SecITC, Bucharest, Romania, November 8-9, 2018*, volume 11359, pages 314–330. Springer.

Bundesamt für Sicherheit in der Informationstechnik (2014). SI-CC-PP-0084-2014: Security IC Platform Protection Profile with Augmentation Packages. Version 1.0. https://www.commoncriteriaportal.org/files/ppfiles/pp0084a_pdf.pdf.

CAR 2 CAR Communication Consortium (2019). Protection Profile V2X Hardware Security Module. Version 1.4.0.

Cerdeira, D., Santos, N., Fonseca, P., and Pinto, S. (2020). SoK: Understanding the Prevailing Security Vulnerabilities in TrustZone-assisted TEE Systems. In *2020 IEEE Symposium on Security and Privacy, SP 2020, San Francisco, CA, USA, May 18-21, 2020*, pages 1416–1432. IEEE.

Clulow, J. (2003). On the Security of PKCS#11. In Walter, C. D., Çetin Kaya Koç, and Paar, C., editors, *CHES*, volume 2779 of *Lecture Notes in Computer Science*, pages 411–425. Springer.

Consortium, C. C. (2013). Common Criteria (*aka* CC) for Information Technology Security Evaluation (ISO/IEC 15408). Website: http://www.commoncriteriaportal.org/.

Delaune, S., Kremer, S., and Steel, G. (2008). Formal Analysis of PKCS#11. In *Proceedings of the 21st IEEE Computer Security Foundations Symposium, CSF 2008, Pittsburgh, Pennsylvania, USA, 23-25 June 2008*, pages 331–344. IEEE Computer Society.

ETSI (2019). Smart Secure Platform (SSP) — ETSI TS 103 666-1 V15.0.0. https://www.etsi.org/deliver/etsi_ts/103600_103699/10366601/15.00.00_60/ts_10366601v150000p.pdf.

ETSI, TC CYBER (2019). EN 303 645 V2.0.0, Cyber Security for Consumer Internet of Things.

FP7 European Project (2020). E-safety Vehicle Intrusion protecTed Applications (EVITA). https://www.evita-project.org/, accessed on Oct 27, 2020.

Global Platform (2018). Virtual Primary Platform (VPP) v1.0.1. https://tinyurl.com/yybvbge4.

Global Platform (2020a). Security Evaluation Standard for IoT Platforms (SESIP). Version 1.0. Document Reference: GP_FST_070.

Global Platform (2020b). Trusted Execution Environment (TEE) Committee. https://globalplatform.org/technical-committees/trusted-execution-environment-tee-committee/.

GlobalPlatform Device Committee (2016). TEE Protection Profile Version 1.2.1, GPD_SPE_021. https://www.globalplatform.org/specificationform.asp?fid=7831.

International Society of Automation (ISA, https://www.isa.org/) (2020). ISA-62443-1-5. Security for industrial automation and control systems Industrial automation and control system protection levels.

ISO/IEC 7816 (2014). (Joint technical committee (JTC) 1 / Sub-Committee (SC) 17), Identification cards – Integrated circuit cards. (details).

NIST FIPS (2009). (Federal Information Processing Standards) publication 140-3, Security Requirements for Cryptographic Modules (Draft, Revised). page 63. http://csrc.nist.gov/groups/ST/FIPS140_3/.

Novella, E. (2020). TEE-reversing: A curated list of public TEE resources for learning how to reverse-engineer and achieve trusted code execution on ARM devices. https://github.com/enovella/TEE-reversing (accessed December 7, 2020).

OASIS (2020). PKCS #11 TC. https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=pkcs11.

OCP Security (2020). Specification of Root of Trust, version 1.0. https://www.opencompute.org/projects/security.

Open Titan (2020). Open source silicon Root of Trust (RoT) project. https://opentitan.org/, accessed on Oct 28, 2020.

Souissi, Y., Facon, A., and Guilley, S. (2019). Virtual Security Evaluation - An Operational Methodology for Side-Channel Leakage Detection at Source-Code Level. In *Codes, Cryptology and Information Security - Third International Conference, C2SI, Rabat, Morocco, April 22-24, 2019*, pages 3–12.

Trusted Computing Group (2019). Trusted Platform Module (TPM), Library Specification, Family "2.0", Level 00, Revision 01.59. https://trustedcomputinggroup.org/work-groups/trusted-platform-module/.