IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 7 October 2022, accepted 25 October 2022, date of publication 28 October 2022, date of current version 4 November 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3217815

RESEARCH ARTICLE

Integration of Hardware Security Modules and Permissioned Blockchain in Industrial IoT Networks

ANTONIO J. CABRERA-GUTIÉRREZ[1,2], ENCARNACIÓN CASTILLO[2], ANTONIO ESCOBAR-MOLERO[1], JOSÉ A. ÁLVAREZ-BERMEJO[3], DIEGO P. MORALES[2], AND LUIS PARRILLA[2]

[1]Infineon Technologies AG, Neubiberg, 85579 Bavaria, Germany
[2]Department of Electronics and Computer Technology, Faculty of Sciences, University of Granada, Granada, 18071 Andalucia, Spain
[3]Department of Informatics, University of Almeria, Almeria, 04120 Andalucia, Spain

Corresponding author: Antonio J. Cabrera-Gutiérrez (antoniojavier.cabreragutierrez@infineon.com)

This work was supported in part by Infineon Technologies AG; in part by the European Union's Horizon 2020 Research and Innovation Program through the Cyber Security 4.0: Protecting the Industrial Internet of Things (C4IIoT) Project under Agreement 833828; and in part by FEDER/Junta de Andalucía-Consejería de Transformación Económica, Industria, Conocimiento y Universidades, under Project B-TIC-588-UGR20.

ABSTRACT Hardware Security Modules (HSM) serve as a hardware based root of trust that offers physical protection while adding a new security layer in the system architecture. When combined with decentralized access technologies as Blockchain, HSM offers robustness and complete reliability enabling secured end-to-end mechanisms for authenticity, authorization and integrity. This work proposes an efficient integration of HSM and Blockchain technologies focusing on, mainly, public-key cryptography algorithms and standards, that result crucial in order to achieve a successful combination of the mentioned technologies to improve the overall security in Industrial IoT systems. To prove the suitability of the proposal and the interaction of an IoT node and a Blockchain network using HSM a proof of concept is developed. Results of time performance analysis of the prototype reveal how promising the combination of HSMs in Blockchain environments is.

INDEX TERMS Blockchain, cryptographic standards, hardware security module, hyperledger fabric, trusted platform module.

I. INTRODUCTION

Industrial Internet of Things (IIoT) collects and analyses data to deliver insights that help industrial organizations become more agile and making better-informed business decisions more quickly than ever before [1]. This leads to better quality control, and more efficient, streamlined supply chain management. It also benefits predictive maintenance, field service, energy and facilities management, and asset tracking.

In the Digitization of Everything era, security breaches are no longer even newsworthy. The spread of cloud services and the advent of the Internet of Things (IoT) have urged enterprises to enhance security and rethink their company policies. The overall complexity of a smart factory IoT system is extensive, and the number of security loopholes subsequently increases to a dramatic extent [2]. Clearly, traditional firewalls and antivirus systems will not be sufficient to protect complex IIoT infrastructures. An IIoT network requires an advanced security system, not only to ensure a non-disruptive smart factory workflow or to protect employees and assets, but also to secure business-critical information from competitors.

The information produced by the IIoT devices needs to be gathered and stored securely in specialised systems or hardware. Usually, for managing and processing this information, a client-server model is set up where dedicate machines, provide functionality to other programs or devices. These functionalities may include sharing data or resources between

The associate editor coordinating the review of this manuscript and approving it for publication was Sedat Akleylek.

multiple clients, performing computation for a client using specialised software, or simply to gather and store information securely produced by the computational clients (in what respects to Industry 5.0, such actors might be the IIoT devices and robots).

Protocols used in IIoT typically are implemented using client-server (Zigbee [3] or LoRa [4]) or publish-subscribe (MQTT [5]) paradigms. In order to ensure enrollment and communications in such networks, these protocols often include additional mechanisms to introduce security such as symmetric encryption, mainly based on Advanced Encryption System (AES) [6], or they can also include public-key cryptography through Transport Layer Security (TLS) [7]. Nevertheless, these types of architectures are prone to cyber-attacks [8].

The cost of cybercrime includes damage and destruction of data, stolen money, productivity losses, theft of intellectual property, theft of personal and financial data, embezzlement, fraud, post-attack disruption of the normal course of business, forensic investigation, restoration of harnessed data and systems, and reputational harm. In this context, industrial organisations pursuing to implant IIoT, the concern for a cyber-attack is not only focused on loss of data, but also on safety, integrity and availability of data and services. Consequently, the top four IoT security issues that need the greatest attention are authentication/authorization, access control, data encryption and the use of IoT devices as potential gateways to sensible systems [9].

Decentralized paradigms provide solutions to these needs by allowing data access control by different entities in order to enable auditability of events and policies, and to verify the integrity of all data items. Blockchain solutions are based on this concept [10], making use of cryptography to sign transactions or to add/remove nodes to/from the network. Distributed ledger technologies [11] (DLT) such as Blockchain, are based on maintaining distributed copies of a database which contains records of the transactions performed across the network. This scheme, along with a consensus algorithm previously agreed by all participants in the network for validating the transactions, allows reaching authenticity and immutability of those records [12]. However, these networks present serious scalability problems when the ledger is required to be updated and validated by a large amount of participants [13]. In order to avoid this issue, the number of participants in the network should be limited, or the traditional consensus mechanisms should be modified. The solution adopted by Blockchain networks designed as a support to currencies, as Bitcoin or Ethereum, where the transaction must be validated by 51% of the entities that makes up the network, does not provide a feasible solution to the scalability problem and requires high computing and energy resources. A more efficient proposal is the known as Permissioned Blockchain (PB) [14].

PB is a distributed ledger which is not publicly accessible. In this scheme, the participation of a member in the network requires certain permissions granted at registration time by Blockchain administrators through certificates. Hence, PB offers an additional security layer over typical Blockchain networks such as Bitcoin. Furthermore, PBs are compound by entities who require an identity and a role definition within the Blockchain.

Typically, the keys and certificates involved in a Blockchain are stored in a ''software wallet'' [15]. In the case of public-key cryptosystems, public-private key pairs have to be generated through random number generators (RNG) in order to follow cryptographic standards. If these RNGs are implemented in software, the generated keys are also stored in software, thus becoming a security vulnerability [16]. Software-based security is not enough to protect systems as the stored data can be read, modified and distributed effortlessly. In order to avoid it, a hardware-based root of trust that renders embedded software trustworthy becomes necessary. In this sense, Hardware Security Modules (HSMs) offer a solution which relies on [17]:

- (I) High entropy random number generation.
- (II) Tamper-proof protection, by enabling secure storage of private cryptokeys and sensitive information. In this sense, HSMs are designed to guarantee inaccessibility of store information from external means, thus hindering physical attacks.
- (III) Keys backup and restoration.

In short, the existing problems in traditional architectures such as the centralization of resources can be mitigated by a decentralization of them using Blockchain technologies. Nevertheless, this introduces a new concern regarding how to protect sensitive data, since typically, these data are stored in software repositories. Protecting cryptographic material that is used intensively in Blockchain networks has become essential, thus making the use of HSMs sense.

The use of HSMs allows the storage and generation of the keys in a secure way. Thus, the combination of both components, HSMs and DLT technologies, offer a high robustness to the system in two levels:

- (I) HSM adds a new security layer –hardware-enabled security level– which impacts in the higher-level system security protocols.
- (II) DLT enables horizontal security –security between entities connected to the network in the same layer– in device-to-device communications. This level relies on the decentralized access control.

With the union of these two technologies, the problems of centralization and the protection of keys in software repositories are solved. This paper proposes the integration of HSM (focusing on Trusted Platform Modules (TPM)), and Blockchain, emphasising the key elements that make this integration possible, as well as the development of a proof of concept that demonstrates the suitability and performance of the communication between these two technologies in an IoT node.

The rest of the article is organized as follows: Section II presents an overview of the actual communication paradigms

and the way they manage cryptography used in the context of IIoT networks. Section III describes HSMs, focusing on TPMs, and PB technologies, focusing on Hyperledger Fabric. In Section IV, there will be a discussion about the different key components which are essential to integrate both technologies, mainly public-key cryptographic algorithms and standards (PKCS). Furthermore, a proof of concept is presented, showing the different interactions between IoT nodes and the Blockchain network. This section also presents a performance test which shows the capabilities of TPM when operating on an IoT node in a Blockchain network. Furthermore, a security analysis is carried out showing the attacks that this architecture prevents. Finally, Section V summarizes the conclusions, emphasizing the benefits of the union of these two technologies.

II. STATE OF THE ART

Current security paradigms on computer networks are based on Public Key Infrastructures (PKI) [18]. In this type of paradigms, the security of the overall system relies on a Certification Authority (CA), thus presenting some issues inherent to centralization: on the one hand there is a Single Point Of Failure (SPOF), and in the other hand, every attack will be directed to CAs, which will require an extremely high level of security. Also, Denial of Service (DoS) attacks can be performed more easily [19], and the requirement of user identification for registering in CAs implies lack of anonymity and privacy.

Since the emergence of IoT technologies, especially in industrial networks, the development of new security paradigms has become a need. In this scenario Blockchain is considered the most relevant technology for introducing a decentralized security system in IIoT networks. The combination of IIoT and Blockchain offers a trusted system where the information is reliable and can be traceable. Data stored in a Blockchain ledger remains immutable over the time as well as the sources remain identified at any time.

In a typical IoT system, registration and authentication data are stored in a central entity. These data are required for the registration of new IoT nodes, but the need of this central entity introduces some vulnerabilities, as has been previously carried out. One solution to this issue consists on decentralising this architecture as shown in Fig. 1, where the database corresponding to the central entity is replicated in different client nodes. After decentralization, an attacker has more difficulties to perform unauthorized modifications in the database, because the different clients have to approve these changes. As will be described in next subsection, Blockchain enables the implementation of that decentralized infrastructure.

A. BLOCKCHAIN NETWORKS

Blockchain technologies are being applied to multiple fields [20], [21], [22], although its application to the IIoT scenario is relatively recent. In this field, Blockchain has a lot of potential use cases as automotive and mobility [23], consumer

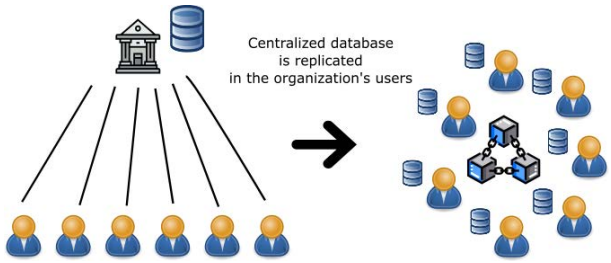

FIGURE 1. Typical PKI infrastructure vs. decentralized ledger infrastructure.

applications [24], tracking logistics [25], supply chains [26], energy [27], and health [28]. In all these cases, Blockchain acquires greater relevance due to the participation of different organizations in order to complete a given process. These organizations must get to an agreement that will be later translated into policies implemented in the Blockchain in the form of smart contracts, which reflect in source code the existing relationships among the organizations involved. The execution of these contracts is ensured intrinsically.

In recent years, several Blockchain platforms have been developed which are subject to continuous changes. One of the most popular is Ethereum [29], which is the first open-source Blockchain platform introducing the concept of smart contracts. As it is well known, smart contracts enable a lot of new applications of Blockchain beyond cryptocurrencies. Smart contracts are an useful feature for IIoT, but in the case of Ethereum, some characteristics prevent from being used in the IIoT use cases. Basically, its consensus algorithm is based on Proof of Work (PoW) [30]. This process requires a high amount of computing resources to avoid attackers to modify the Blockchain, while users generating a new block are rewarded with cryptocurrencies. This consensus algorithm is not applicable to an IIoT environment because the nodes that make up the network are usually low-power consumption devices, and they are not designed to perform such large computations, but for data collection and even run some control algorithm. There are other types of consensus mechanisms such as the Proof of Stake (PoS) [30] which has lower computing requirements, but in contrast, this consensus mechanism requires that the entity who wants to participate in the network must make use of cryptocurrencies, i.e, all the entities in the network must have at least a small amount of them.

The use of smart contracts in Blockchain networks can generate vulnerabilities at the application level that an attacker can exploit. While this can lead to a problem, the use of HSM at the hardware level helps to mitigate these vulnerabilities by restricting the attacker's attackable surface to higher layers of the application.

There are other Blockchain platforms such as Multichain [31] or Quorum [32], which are different variations of Bitcoin and Ethereum, respectively. Multichain is a private Blockchain as opposed to its counterparts, which are public. Quorum is the permissioned version of Ethereum, what

means that the participants of the network have some restrictions and they play different roles in the interaction among them.

In order to avoid the issues associated with the use of cryptocurrencies, there are other Blockchains that do not use cryptocurrencies, thus making them more adaptable to IIoT use case. One example is Hyperledger Fabric [33] which will be discussed further.

B. HARDWARE SECURITY LAYER

Blockchain introduces useful features for building a decentralized infrastructure, but it also introduces a significant security risk regarding the storage of keys in the different nodes. Usually these keys are kept in a repository, but not only that, they are also generated by software. This can lead to a major security breach, since algorithms used to generate required keys could be vulnerable to different attacks. Concretely, Pseudo-Random Number Generators (PRNG) commonly used for generating keys are vulnerable to key replication if seeds are predictable or not properly randomized [34], [35].

In the particular case of IoT devices these issues arise intensively because it is not easy to have good sources for generating true random values required for the seed, being firmware-generated random values not enough for guaranteeing secure keys generation. Therefore, for having a reliable entropy source in these devices it is necessary to generate random values directly from physical sources. Then, a feasible solution is to use the well known True Random Number Generators (TRNG), which generate true random numbers from high entropy microscopic physical events in the hardware as statically noise of signals, photoelectric effect or quantum phenomena [36]. In this way, HSMs offer TRNGs which make it ideally to be used in devices that interact with the Blockchain. HSMs hinder side-channel attacks in the sense that when the key is generated inside the chip, the attacker cannot know the time the chip has taken to create the key internally, and there are specific countermeasures against the analysis of noise, electronic leaks and power consumption [37], [38].

Regarding the storing of the generated keys, they are usually stored in repositories, which it is a big risk, as commented before. Indeed, the keys can be easily extracted, manipulated and replicated by an attacker.

A little bit more robust method to protect the keys by software is a Trusted Execution Environment (TEE) [39]. TEE is a standard that creates an isolated environment which runs over or in parallel with the operating system. A TEE guarantees the authenticity of the executed code, the integrity of the runtime states and the confidentiality of its code, data and runtime states stored. Thus, TEE provides secure enclaves in order to execute and store sensitive assets and critical data such as private keys.

Although a TEE enabled system resists software attacks, it is still vulnerable to kernel faults, side-channel and physical attacks, which can be performed in order to undermine the isolated environment [40]. Furthermore, in the case of IoT devices TEE can not be implemented, because they usually run firmware without any operating system support.

This issue can be overcome using an HSM, because once the key has been generated, it can never be extracted, thus providing a secure storage for generated keys. In this way, HSMs present tamper resistance which avoids physical attacks such as probing attacks where an attacker sets a probe on a wire and reads the signals being transmitted over the wire during chip computations [41].

As discussed above, the Blockchain platforms and hardware security layer technologies have drawbacks and security issues that can be exploited in certain scenarios, such as extracting the cryptographic keys from a TEE using hardware attacks or failing to properly protect cryptographic material used in a Blockchain network. This is why this article brings together the combination of HSMs, focusing in TPMs, and PB networks offering together the benefits of each technology, thus providing an IIoT architecture that incorporates different layers of security, proposing a more robust system on the technologies analysed in this section.

The next section will discuss in more detail the benefits of HSMs, in particular TPMs as well as PB technologies focusing in Hyperledger Fabric.

III. BACKGROUND

Before starting to discuss about the integration of the two components and the proof of concept presented in this article, the components that make up the proposal of this paper should be introduced, focusing especially on TPMs and Hyperledger Fabric since the proposed integration cannot be understood without an in-depth knowledge of the characteristics offered by these technologies.

A. HARDWARE SECURITY MODULES

HSMs are being extensively used for device protection, providing a secure framework for authentication and identification. An HSM consists of a cryptographic processor which implements in hardware different cryptographic algorithms required for these tasks. In this sense, it offers tamper protection against harmful manipulation and strong authentication mechanisms [42].

HSMs usually are delivered in different form factors. Typical ones are security cards, widely used in people identification, and chips installed in a PCB which are connected to the CPU of the system under protection. The HSM will be required to perform different security tasks: signing, signature validation, encryption, decryption or hashing, as well as secure storage and trusted random number generation. In short, an HSM provides a root platform of trust [43].

TPMs are a subgroup of HSM devices, whose features are defined by the Trusted Computing Group (TCG) [44]. In the next section it will be explained in detail.

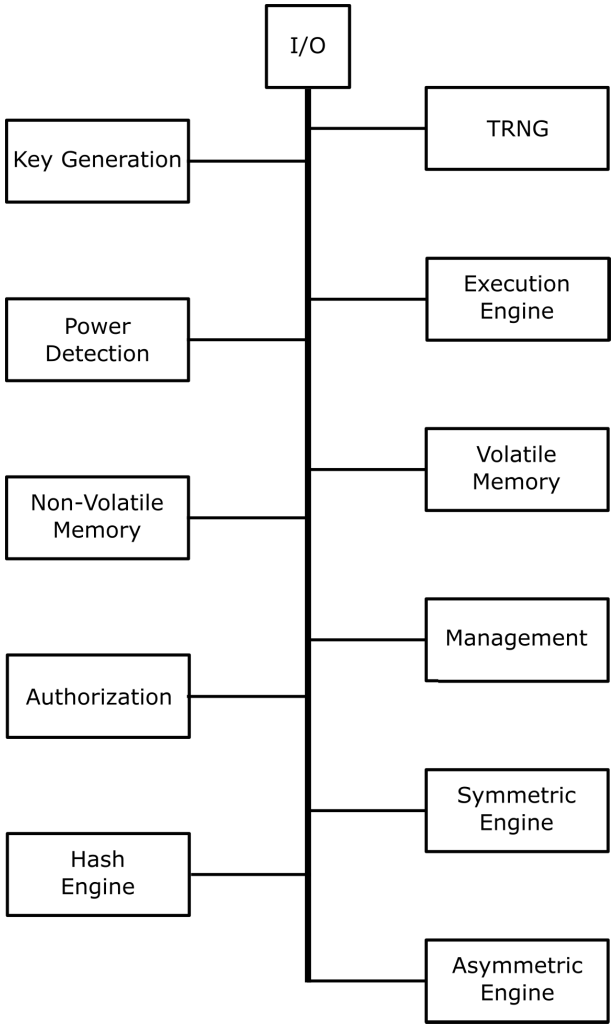

FIGURE 2. TPM2.0 architecture.

1) TRUSTED PLATFORM MODULES

TPMs are standardized by the TCG. Being TPM2.0 [45], [46] the latest specification. In all of them, a TPM is defined as a hardware device including volatile and non-volatile storage, and a set of cryptographic algorithms implemented in hardware. In addition, the different TPM standards include an API specification to interact with the TPM [47]. In short, a TPM is a hardware component that provides secure storage of cryptographically protected data (keys, certificates, passwords and other data related to the internals of the TPM as Platform Configuration Registers) and enables the generation of keys inside the TPM using TRNG, private/public key encryption and signature operations. Fig. 2 shows the architecture of a TPM2.0 device [45], where the different components are interconnected by a bus, which also connects to the I/O interface. In addition to the aforementioned TRNG and the non-volatile memory, where the Platform Configuration Registers (PCR) [48] are located, there are other important modules such as the symmetric and asymmetric key engines and the key generation engine.

Typically, TPMs are used in computing systems supporting a BIOS or a similar firmware in charge to boot the device for adding a security layer below the software. Indeed, keys generated in the device cannot be extracted outside of the TPM, hence, data secured by these keys will be not exposed. In this sense, the TPM provides a root of trust. In this scheme, PCRs are records containing a concatenation of hashes [48] which are the base of the different protection mechanisms performed by the TPM. As an example, during the secure boot process provided by modern BIOSes, the startup firmware checks different parameters of the system as the peripherals attached, the status of the memory, and others. These parameters are hashed and compared to the ones stored in PCRs. If the result of these comparisons is correct, the system will boot successfully. The hashed values are usually critical parameters of the system, thus preventing the system from booting if any modification is detected [49]. Another important feature provided by PCRs is attestation, which is the ability of proving authenticity in the system using a public-key cryptosystem with an Endorsement Key (EK) [50]. In a typical attestation scenario, the PCRs values along some piece of code or executable are signed with the EK in the TPM. In order to verify the authenticity of the code, the signature has to be verified jointly to the PCR values [51] thus guaranteeing that the code which is running in the system is totally secure.

PCRs are also involved in another operation known as key sealing. When the measured system parameters correspond with the values stored in the PCRs, it is possible to unseal a key used for encrypting data in the system. Without this key, data cannot be decrypted, preventing the access to protected data if the system is not in a safe state [52]. Furthermore, the process of encrypt/decrypt used with the TPM is called binding/unbinding since the data encrypted using a key in the TPM can be only decrypted using the same key of the TPM. If this key is used in other TPM the data cannot be decrypted/unbounded.

In conclusion, the TPM can perform different cryptographic operations and results in a root of trust from which different operations can be carried out with security guaranteed by the TPM. In fact these operations can also be performed by a normal HSM, but the main advantage of a TPM is that these operations are supported by the TCG, which ensures that all these operations are in the current state of the art and are standardized.

B. PERMISSIONED BLOCKCHAIN TECHNOLOGIES

As it had been commented before, Blockchain technologies offer a new paradigm that comes to replace the typical PKI schemes. However decentralized solutions built on Blockchain technologies have difficulties to scale to the amount of devices and data aggregated by IIoT. This is one of the reasons why PB technologies appear [53].

These technologies offer more versatile consensus protocols and allow the enrollment of a limited amount of participants in comparison with public and no-permissioned Blockchains. This relies on the fact that it eliminates the

unnecessary computation required to reach the consensus protocol. In this scheme, participants in the Blockchain network have different permissions to read, access and write information, while the configuration of the Blockchain is defined by the policies of the network, typically agreed by the members participating in it. As a consequence, the policies defined within a PB affect the behavior of all the members. In any case, this dependency on the policies does not affect the classification of the defined Blockchain in terms of being public, private, public permissioned or private permissioned. What really makes the difference is the maintenance of the identity of each participant in the Blockchain. This means that an entity can participate in the Blockchain network if, and only if, it has been previously certified as acting as itself by a CA. This is why the approach of the PB networks are considered as hybrid, since there is a central authority (the CA), which is the one that gives the authorization to be able to participate in the network [54], thus not being totally decentralized as in the permissionless networks.

This is the main feature that makes permission-based Blockchain networks so popular in the industry since security, identity and a defined role for each network participant are required. There are several cases of use in the industrial field [55], [56] that utilize this type of technology. In order to provide some examples, the most typical one is the traceability chain, which must ensure that the characteristics of some consumer product are kept intact through the logistic chain. In this case, the agencies involved in this business model would be the intermediaries/logistics service, producers and consumers. Another example would be the supply chain, which would involve as many logistics services as banks and sellers/buyers. These are just some scenarios, but there are more cases emerging, as the energy exchange which is another scenario that is taking more importance with the time [57]. To carry out the implementation of these use cases there are whole suites and frameworks dedicated to the implementation of PB technologies, being Hyperledger Fabric one of them.

1) HYPERLEDGER FABRIC

Hyperledger Fabric (HLF) [33], [58] is a PB with support for executing smart contracts. HLF allows organizations to collaborate in a Blockchain establishing different roles and entities. Each node has its own function depending of its role that carries it out. HLF defines four different nodes, which are:

- HLF peer nodes, used to store a copy of the ledger, to endorse new transactions by invoking smart contracts, to commit new blocks into the ledger and to allow querying the ledger.
- HLF ordering nodes, used to create and distribute new blocks of transactions to the HLF peers. The organization that owns this node will be the one that creates the network and establishes the policies that govern the network.
- HLF clients, used to communicate with peer and ordering nodes in order to query the ledger and to propose new transactions.
- HLF Certificate Authorities, which issue certificates to administrators and network nodes.

In addition to this, a consortium must be defined specifying which organizations will participate in the network. These organizations communicate through a channel. This offers great versatility since an organization can participate in several channels with different organizations at the same time. For example, in the case of the supply chain, the logistics service can participate in one channel with the buyer and in another with the seller, but the buyer and seller do not have a common channel.

2) HYPERLEDGER FABRIC OPERATIONS

In order to start the network, a client needs to execute some function contained in the smart contracts. When a function is triggered, an operation is done in the ledger: *write* or *read*. When an operation is executed the endorsement policy comes into play. It describes which organizations must approve transactions before they will be accepted by other organizations onto their copy of the ledger.

When a transaction proposal takes place, that is, the client initiates it, the process to be carried out is the following:

- The peers verify that the transaction is well formed and that it has not been done before, avoiding replay attacks. It is also verified that the future transaction satisfies the policies of the channel.
- The transaction proposal executes some function of the smart contract.
- A response is produced which will be reflected on the state of the ledger if it is a write operation (or not if it is a read operation).
- The application disseminates both the transaction proposal and the response within a message to the peers for them to verify them.
- Once the transaction is verified following the channel policies, each peer updates its ledger and the status of the database.

Another operation that typically takes place in HLF is the enrollment process of users to the network. This operation is performed in the client, which has to contact the CA when it wants to enroll. Then, the CA issues a certificate as a result of the enrollment operation.

HLF defines two types of enrollment depending on whether the client to register as an administrator or a normal user. In the case of to be registered as administrator it is required to provide first some credentials (username and password) which must be already configured in advance in the CA. Then, the client makes a Certification Signing Request (CSR) where he attaches those credentials and, as a consequence of that, the CA returns a valid certificate if registration has result successful.

In the case of a registration as a user, the process is similar except that it requires the previous enrollment of an administrator which serves the user to register in the CA. Once this is done, the process of enrollment is the same as for an administrator. So in conclusion, in the administrator there is a process of enrollment and in the user a process of registration and enrollment.

As it can be seen, within the operations to be carried out in the Blockchain there are different cryptographic operations which should be performed in the TPM. In the next section a detailed integration between these two components is provided, Hyperledger Fabric and TPM, describing the operations carried out by a Blockchain network.

IV. INTEGRATION BETWEEN HYPERLEDGER FABRIC AND TRUSTED PLATFORM MODULE

Every operation related to the DLT requires cryptography [59], the need of secure hardware support by means of an HSM is a promising solution to the potential vulnerabilities exposed by relying just on software.

A. ELLIPTIC CURVE CRYPTOGRAPHY

Key generation algorithms are involved in the generation of cryptographic keys. The size of these keys is directly related to the memory resources required for storing them, and the corresponding certificates and digital signatures. In the past, RSA [60] was the preferred Public Key Cryptosystem (PKC), but the updated computing capabilities of attackers requires a continuously increasing size of the RSA keys, which represents a problem for processors with limited computational and energetical resources [61]. In this context, Elliptic Curves Cryptography (ECC) has emerged as an alternative to RSA for PKC, as it provides a similar level of security to RSA, but requiring smaller key sizes.

One of the main issues with ECC is the selection of a secure curve for cryptographic applications [62]. The choice of the curve determines the parameters that lead to its efficiency and security strength. The main curves that are used in ECC algorithms are Weierstraß, Montgomery and Edward Curves. The National Institute of Standards and Technology (NIST) recommends the Weierstraß curves, whose general equation is:

$$y^2 = x^3 + ax + b \tag{1}$$

NIST establishes different recommended curves over binary and prime fields. Some examples of NIST curves defined over prime finite fields are:

- P-192, also known as secp192r1 and prime192v1.
- P-256, also known as secp256r1 and prime256v1.
- P-224, also known as secp224r1
- P-384, also known as secp384r1.
- P-521, also known as secp521r1.

In the case of Blockchain, the first curve used was the secp256k1 [63], also known as the ''Bitcoin curve''. This curve is used also in Ethereum. The prime256v1 curve has been recommended by the NSA for use in government affairs. However, due to the boom in quantum computing, this recommendation has been updated by proposing that the curves to be used for greater safety should be the secp384r1 when quantum computing reaches a more advanced stage. For example, the secp384r1 curve offers 192 bits of security instead the more commonly used curves like prime256v1 which provides 128 bits [64]. In the case of Hyperledger Fabric, the supported curves are prime256v1, also known as NIST-P256 curve, secp384r1 and secp521r1. For the HLF network to be compatible with an HSM, it must support the same or at least one of these elliptic curves. In this paper we have tested the compatibility with a TPM, concretely, with the Infineon OPTIGA^{TM} TPM2.0. Both Infineon TPM and HLF shares the NIST-P 256 curve.

B. PUBLIC-KEY CRYPTOGRAPHIC STANDARDS

Another key point to make possible the integration of these two technologies is the set of cryptographic standards implemented by them.Indeed, standards in cryptography establish the mechanisms and protocols to implement cryptographic algorithms in a secure way while facilitating encrypted data interoperability. Also, if a security breach is discovered, the corresponding standard is discarded and replaced by another. This results in the fact that the standards used are always being updated.

Cryptographic algorithms and protocols are standardized by different organizations dedicated to this purpose, some of them being public, and other private. Examples of public entities are NIST [65] IEEE [66], while RSA security LLC is a private company that has issued a set of standards called Public Key Cryptographic Standards (PKCS). Some of these PKCS standards have been abandoned or withdrawn, but others are in use today. In the case of PKCSs being used both by TPM and HLF, these are the PKCSs concerning communication between them. On the one hand we have PKCS10 [67], which it is also known as CSR. This PKCS specifies the format that messages sent to a CA must follow in order to authenticate the device on the network. This PKCS comes into play when a user or administrator needs to be registered on the HLF network. Once the CSR has been successfully requested, the CA issues a certificate in X.509 format. This certificate is stored in the device, either in the TPM or directly in the client's memory. A CSR contains the applicant's public key and data that acts as a proof of device identifier. Both the public key and the data are signed by the CA's private key, which generates an X.509 certificate, which is sent back to the applicant. On the other hand, the PKCS11 [68] is the standard which defines a standard method to access cryptographic services from tokens/devices such as HSMs, smart cards, and others.

In this sense, PKCS11 isolates an application from the details of the cryptographic device. That means, the application does not have to change to interface to a different type of device or to run in a different environment; thus, it enables the application to be portable. In our HLF-TPM integration

proposal, PKCS11 standard will work as an interface between the TPM and HLF. Indeed, supports PKCS11 standard in order to facilitate integration with HSMs. By default, HLF uses a software wallet in order to store the cryptographic material. Adding the correspondent configuration in HLF, we can change this storage method by a hardware storage. The PKCS11 standard is implemented in the TPM by the TCG group, therefore all the functions and methods that it contains inside are tested and proven using different applications, and we will use this standard as a hinge between these two technologies. The corresponding API is implemented at the highest level within the TPM software stack (TSS), so it abstracts the application on top of it, making it independent of the type of Blockchain network used, of all the functionalities implemented at a lower level in the TSS and of the TPM itself. Then, any TPM that includes the TSS in its implementation will be compatible with applications including the PKCS11 standard.

Having discussed the two main keys to the integration of these two technologies, we will now proceed to explain the different operations carried out between Hyperledger Fabric and TPM2.0. The idea of choosing TPM2.0 as the hardware support element lies in its ability to work with systems that implement operating systems as it enables operations such as trusted boot and remote attestation that other types of HSMs do not have.

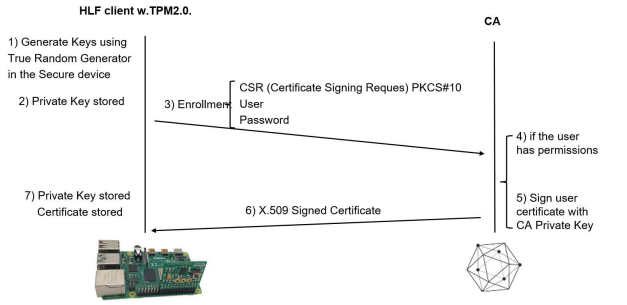

FIGURE 3. Enrollment mechanism.

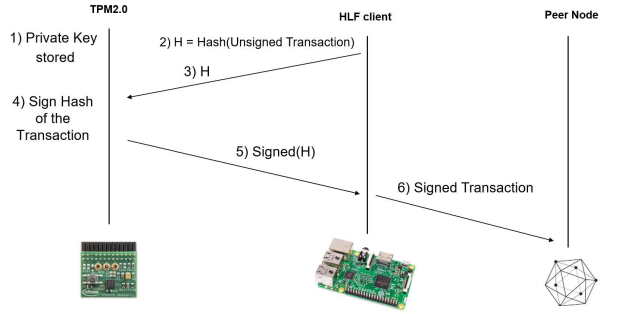

FIGURE 4. Signature procedure mechanism between the TPM2.0 and the peer node.

C. OPERATIONS PERFORMED IN HYPERLEDGER FABRIC USING TPM2.0

In this section we will discuss how some critical HLF operations may be performed jointly with a TPM. These operations, which are traditionally carried out using software tools, which, as explained above, generate security breaches since they can be easily attacked, are executed internally in the TPM. These operations will be the enrollment of new users and administrators, and the signing of transactions. In the following, it is assumed that an HLF client is implemented on an edge node that collects data from an industrial environment. This data is periodically sent to the Blockchain network which is running distributed across different entities, CAs, peers, etc. Fig. 3 shows the proposal of an enrollment mechanism for the HLF client equipped with a TMP2.0 HSM interacting with a CA of the Blockchain network.

When a client wants to enroll in an HLF network, it has to initiate an enrollment handshaking with the CA. This process is the same whether it enrolls as an administrator or as an user, with the exception that the user has to be previously registered by an administrator. As shown in Fig. 3, in this operation the message sent to the CA contains the CSR and a user and password. CSR fields contain information concerning the identity of the client in accordance with the standard. In addition to the CSR, two fields are also included, the user name and password, parameters that have been previously stored in the CA. To successfully complete the enrollment process, the username and password sent have to match with the stored values. In the case of user enrollment, the administrator must have previously registered the user. Therefore, the registration process basically involves sending the user's username and password to the CA. The CA returns a secret and it is used by the user in the enrollment process instead of the username and password.

Note that the main difference between the two processes is that the registration process of the administrator has been previously completed. This may be because the system is deployed with predefined administrators or because it has been initialized through other mechanisms, such as sending the administrator's parameters through another medium, such as through credentials stored on a physical device or through other protocols. Once the CA has received the CSR with the corresponding parameters, the CA signs the CSR with its private key and generates a X.509 certificate as a consequence. This certificate will be sent back to the client, who can store it in the TPM or simply in the device's memory. This certificate will be in charge of validating the communications later, when the client interacts with the rest of the Blockchain network.

Fig. 4 shows the signature procedure when an HLF client interacts with other peer node, where the TPM performs the required cryptographic operations. Note that this direct interaction between peer nodes is only possible if the clients have been previously enrolled into the Blockchain network.

In this situation, transactions are triggered by the HLF clients. These clients, typically, include sensors producing data that is uploaded to the ledger, thus generating a distributed copy of this data. In order to upload and generate this

distributed copy, the clients have to start a transaction. In the case of HLF, the fields in a transaction are:

- Header: Including the metadata of the transaction.
- Signature: A cryptographic signature of the transaction hash.
- Proposal: The input of the smart contract which results in the data to update in the ledger.
- Response: The output of the smart contract; if the transaction is validated successfully, the output will be added to the ledger.
- Endorsements: A list of different endorsements nodes which have to validate the transaction fulfilling the endorsement policies of the network.

These fields form a data structure that is hashed and then signed by the client. Then, this signature is included in the corresponding field, and sent to the HLF network, thus forming a chain. It should be noted that this operation, which is typically executed in software, in our scheme is carried out internally in the hardware included in the TPM. Next, the request is received by the peer nodes which are in charge of validating the transaction and applying the necessary changes on the ledger, in the case of being required. Regarding the transaction verification process, it is carried out by other nodes as the Orderer and peers. In this process, the different nodes verify the transaction signature by means of the client's public key. Note that this process is not internal to the TPM, and it is not carried out within the HLF client where the TPM is located.

Basically these are the main processes involving the TPM in an HLF client. Of course the TPM can act on the other nodes, both CA and peers. The next subsection presents a proof of concept of the proposed application of TPM to HLF networks.

D. PROOF OF CONCEPT

In order to demonstrate the viability of combining TPM2.0 and HLF in an IoT node, a proof of concept has been developed. The scheme of the demonstration is shown in Fig. 5, where the interaction of an IoT node with a simulated HLF Blockchain network is presented. The IoT node is a Raspberry Pi 4 Model B (RPI) with an Infineon OPTIGA^{TM} TPM2.0 attached. The RPI runs a HLF client which interacts with the other nodes being part of the Blockchain network. In this proof of concept, a default HLF network configuration is used, as the objective is to show the feasibility of inter-operation between HLF and TPM and not a deployment of entities for a specific use case. The HLF network includes two organizations and two peer nodes per organization. Each organization has its own CA. There is also an HLF Orderer node, which performs transactions ordering [58] and maintains the list of organizations in the network. Each of these entities runs in Docker containers [58] in a virtual network hosted on a separate computer. This environment shows a scheme of two organizations which are part of a consortium in which a client, that belongs to one of these organizations,

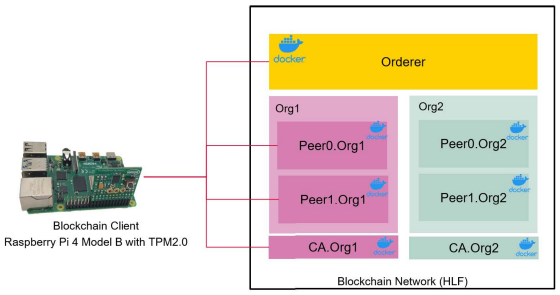

FIGURE 5. Proof of concept, raspberry Pi 4 model B with hyperledger fabric.

wants to register in order to be able to send data to the Blockchain as transactions.

With these elements, this proof of concept allows to show the feasibility of the integration between HLF and TPM2.0, as well as that the mechanisms performed in a Blockchain network from the point of view of a client, can be perfectly integrated in an IIoT network. These mechanisms, both signature and enrollment process, are the ones in which the TPM actively participates in. The implementation of these mechanisms into the TPM creates a root of trust along the Blockchain network. Nevertheless, the possibility of integrating these mechanisms using different firmware or variants in the TPM can lead to different time performances. This aspect will be discussed in the next section.

This proof of concept includes at least two peer nodes per organization to make it more realistic. For validating a transaction made by the client at least two nodes, one from each organization, are required. The interaction between the HLF client and the Blockchain network starts with the enrollment process. As commented previously, the enrollment process in HLF is different depending on the role of the client, which can have the role of administrator or user. In order to enroll a user, the client must enroll an administrator before. The administrator plays the role of registering new users who will be in charge of querying and updating the ledger.

This mechanism is shown in Fig. 6, and starts with the enrollment of the administrator. Note that the CA must have previously initialized some credentials (name and password) in order to allow the operation. When the enrollment process is finished, the administrator receives the certificate issued by the CA. For user enrollment, the user must first register with the CA through the administrator, after that, it returns a token called ''secret'' that can only be used once for the user to enroll. After this process the user receives a certificate which will be stored.

Once this process is completed, the TPM of the RPI will store both the user and administrator keys as well as the certificates issued by the CA. The user is in charge of interacting with the ledger by executing functions defined in the smart contracts running on the peer nodes. These operations can be either query or update operations. Operations involving a write, and therefore, an update to the ledger must be

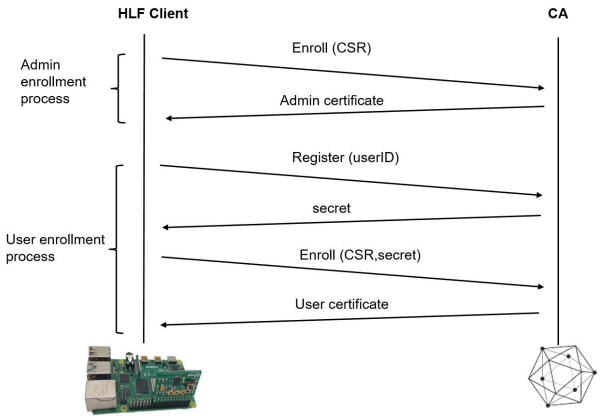

FIGURE 6. Admin and User enrollment process.

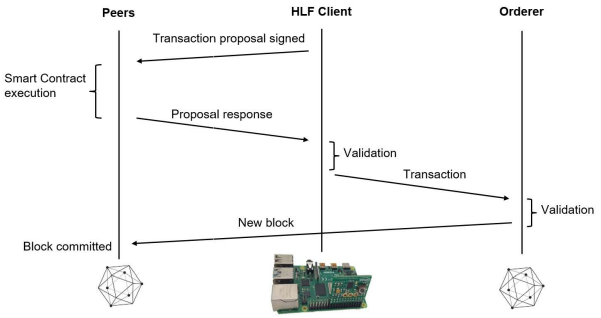

FIGURE 7. Transaction mechanism between client node and the peer nodes.

performed through transactions signed by the client and verified by the peer nodes.

In Fig. 7 it is shown how the client initiates a transaction sending a transaction proposal to the peers. The peers perform the endorsement service which is in charge to execute the smart contract and obtain a proposal response with the output of the smart contract and the endorser's signatures. Then, the client receives the proposal response and validates it. The client builds a transaction and sends it to the Orderer. This transaction includes the transaction proposal, transaction response and the endorsements. The Orderer validates the transaction and creates a block which contains the validated transactions and broadcast this block to all the peer nodes. The peers execute the transaction and update the state of the database decentralized in the peer nodes. The block is finally committed in all peer nodes.

Both the enrollment process and the transaction process are the two main mechanisms that occur in HLF. These processes are initiated by the client, which makes use of the TPM to generate the keys and sign them with the private key. This proof of concept is intended to illustrate what the complete process would look like using an IoT node. On this basis, it is possible to build much more complex use cases that encompass real use cases. The next section shows the results obtained from this proof of concept in order to discuss its application in IIoT environments.

E. PERFORMANCE ANALYSIS

The use of TPM in IoT nodes has certain limitations compared to an HSM, Trusted Execution Environments (TEE) or cryptographic software. Normally an HSM is dedicated to be a hardware accelerator for cryptographic operations. This implies that the time to perform an operation in the TPM is longer than if it were done in an HSM. Otherwise, the gain in security that a TPM offers makes it advantageous depending on the scenario. In this work a benchmark has been performed comparing different approaches. To evaluate this time comparison, an average has been made over 100 samples. The operations to be measured are:

- Generation of the keys: Generation of a NIST-P265 private-public key pair using the TPM.
- Signature: Execution of the signature algorithm using prime256v1 elliptic curve and SHA-256 hash function.
- Verification of the signature: It is carried out using the public key and the signature previously computed.
- Commissioning of the client: The commissioning encompasses both key generation and the CSR, i.e. the signing of the client's public key by the CA in order to issue the certificate to the user. The resulting time is the sum of the creation of the keys plus the signing of the CSR by the CA's private key.

These operations are internal to the TPM because the objective is to measure the time difference between the different approaches. Indeed, the configuration of HLF for those measurements does not affect the performance since this configuration is an independent process to these cryptographic operations. The platform used for the measurements was the same as the one used in the proof of concept in the previous section: Raspberry Pi 4 Model B 8Gb, chipset Broadcom BCM2711, Quad core Cortex-A72 (ARM v8) 64-bit SoC at 1.5GHz. The elements for executing the different algorithms involved in the proof of concept are the following:

- A TSS implementation, for TPM2.0, including a TPM command line interface, all integrated into TPM Tools Release 5.2 [69].
- TPM2 Access Broker and Resource Manager (TPM ABRM), Release 2.4.0 [70], for communication in the cryptographic operations between the TSS and physical or software TPM.
- OpenSSL V3 [71] for executing the cryptographic operations without using TPM.

In order to compare different scenarios, we have considered four approaches:

- Hardware TPM: Using TSS connecting via serial peripheral interface (SPI) to the RPI.
- Software TPM: Using the emulator of TPM developed by IBM [72].
- OpenSSL with Hardware TPM engine: Using cryptographic software like OpenSSL indicating specific engine (TPM2 TSS) in order to perform the operations.
- OpenSSL without engine: This approach only uses the OpenSSL library software.

Table 1 presents the results carried out for these four approaches. Note that from this table, the use of hardware TPM results in an increase in execution time in our proof of concept. Indeed, the results regarding the key generation, show that HW TPM is approximately 5 times slower than SW TPM, 30 times slower than OSSL and only 16 times slower than OSSL ENG approach. With regard to Sign operation, this is 5 times slower than SW TPM, 28 times slower than SW OSSL and 8 times slower than OSSL ENG version. In Verify operation, the HW TPM is 5 times slower than SW TPM, 17 times slower than OSSL and 9 times slower than OSSL ENG. Finally, in the commissioning mechanism, the HW TPM is 5 times slower than SW TPM, 50 times slower than OSSL approach and 8 times slower than OSSL ENG version.

These results are expected as the TPM to act as a hardware accelerator but as an element that offers a higher level of hardware and software security. The approach using Software TPM (SW TPM) is slower than software approaches using OpenSSL (5 and 3 times slower than OSSL and OSSL ENG for key generation, respectively). This is because the software TPM uses a socket for the communication between the TPM simulator driver and the TPM engine. In the case of operations using OpenSSL with TPM engine (OSSL ENG), the required time is much higher than using OpenSSL without TPM engine (OSSL) (almost 2 times slower in key generation and 6 times in commissioning), although it is less than Hardware TPM (HW TPM). This is due to the additional overhead generated by the communication between the OpenSSL stack and the TPM. It is clear that the fastest approach is to use software only (3.5 times faster than SW TPM in sign operation) as operations are not using the slow SPI link to retrieve data from the TPM and all the computing is made by the cores and the memory controllers which is much faster than the connection to the HSM. This software approach leads to it being the most widely used option in systems that do not implement any kind of security. The significant time overhead carried out when using a hardware TPM is a disadvantage in scenarios where high performance of real-time is required but the efficiency in the implementation of the smart contracts can help in hiding latencies if concurrent operations can be performed. So the selection of the HLF and the ability to implement the smart contracts is also key. Nevertheless, in systems where safety is a key factor, this time overhead is compensated by security features provided by a hardware TPM. In fact, TPMs were conceived to provide system robustness and a secure storage system. Its main features do not include the increasing of the processing speedup, which is reasonable since they are not designed for that purpose. HSMs in general provide co-processing when performing cryptographic operations, many servers use HSMs to speed up cryptographic operations of libraries such as SSL. What makes the use of TPM specially suitable compared to other HSMs is that, in addition to offering the features of HSMs, it offers higher level mechanisms such as secure booting and remote attestation. The implementation of these mechanisms leads to TPMs following a standardization process and, hence, the software stack they implement is more robust. This standardizing element in TPMs is what makes it so promising in IoT systems as it is much easier to include in different environments, such as Blockchain networks.

TABLE 1. Execution time of the different approaches in milliseconds.

	HW TPM	SW TPM	OSSL	OSSL ENG
Create	0.5746	0.112	0.0197	0.0347
Sign	0.6030	0.1070	0.0212	0.0727
Verify	0.3379	0.0678	0.0195	0.0377
Commissioning	1.9934	0.3565	0.0384	0.251

In IIoT speedups are important but not losing messages and ensuring that the data feeding the smart contracts is legit a priority. Because this data will impact the big data pipelines of the industry. With the inclusion of a TPM module on the board of the IIoT node dedicated to data collection, it has to be considered that given the particular conditions of the design the gains of this approach are greater than the losses.

Regarding the timing differences of the hardware proposal versus the software emulated versions, it has to be considered that an ARM node running a conventional operating system has been used (it is not a native IIoT procedure) where it has to prioritize the tasks of attention to GPIO and the operating system tasks, in this scenario, communications via SPI are not particularly prioritized, so for example any block maintenance operation in the storage FFS modules would have higher priority than access to the SPI. This, in a specific IIoT node would probably not be so unbalanced. Still, the time differences are not an argument to consider this a non-viable solution. An IIoT node collects data over a period of time (it doesn't just collect a piece of data and immediately forward it to the Blockchain). This makes the time differences between the emulated and hardware versions negligible as data collection times overlap with TPM access times. According to Fig. 8, the actual situation of these nodes is such that:

Even so, as far as scalability is concerned, in this type of architectures we can find an important bottleneck in the loss of messages to be received by the transaction Orderer node and in how the smart contracts to which the data incorporated in the transactions are directed have been implemented. Thus, for example, transactions with different timestamps can be computed concurrently [73]. The addition of a TPM can be efficiently hidden, as shown in Fig. 8 and that the existence of an increasing number of IIoT nodes equipped with this protection does not impact on the overall performance. However, it should be considered that in order to optimize the scalability of the whole architecture and its security, the nature of the transactions should be analyzed to exploit their parallelism to the maximum and be aware that the data processed by the smart contracts will be incorporated into a big data pipeline that will affect future business processes, so guaranteeing their security is crucial. Attacking the software repositories (and wallets) where keys and certificates are stored through smart contracts is nowadays a very fashionable attack vector. This is another reason why TPM should be included.

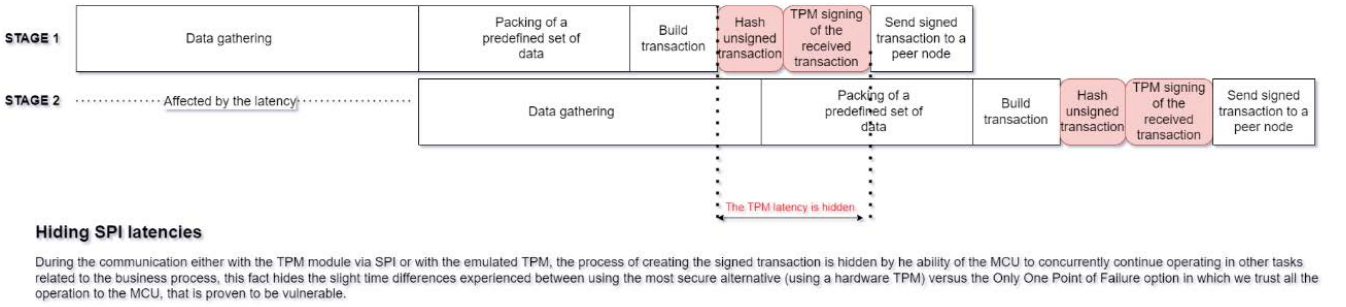

FIGURE 8. Comparison between TPM access time and IIoT data collection procedure.

TABLE 2. Security features of each approach.

	Physical Attacks	Micro-architectural Attacks	Software Attacks
Hardware TPM	✓	✓	✓
Software TPM	✗	✓	✓
OpenSSL	✗	✗	✗
OpenSSL with TPM engine	✓	✓	✓

In this sense, Table 2 summarizes the four approaches analyzed in relation with the protection offered against three sets of attacks: Physical attacks [9], micro-architectural attacks [74] and software attacks [75].

F. SECURITY ANALYSIS

Table 2 summarizes the different attacks against the different implementations studied for our proposal in the performance analysis. As can be seen, the use of a Hardware TPM is the approach that avoids the most kind of attacks. From a practical point of view, the proof of concept presented in this article is the basic resilience unit on which a IIoT network, consisting of different nodes implementing both a TPM and a Blockchain client, will be built.

The integration of the TPM with Blockchain technologies in networks of IoT nodes presents important security advantages. The main advantage of using a TPM in this type of architectures when compared to other technologies such as TEEs, is the ability to prevent physical attacks. Even so, within an IIoT environment, a network implementing this concept is exposed to other types of attacks:

- Denial of Service attacks [19]: As in our proposal the network has no central entity, attacks over specific nodes will not cause the network outage, thanks to decentralization provided by the Blockchain [53].
- Side channel attacks: HSMs and in particular TPMs avoid these kind of attacks offering tamper proof protection. Examples of these types of attacks are timing attacks [76] or fault induction techniques [77].
- Authentication attacks: Using multi-factor authenticated schemes implemented in the TPM and storing the credentials as private keys in hardware, make this type of attack useless in case an attacker wants to fraudulently access the Blockchain [78].
- Reverse engineering attacks: TPMs by offering tamper proof protection against invasive attacks, in which an attacker attempts to modify or alter the intrinsic functioning of the hardware, learning how it works or making it work as he wants, prevent such attacks [79].
- Replay attacks: This type of attack is compromised in Blockchain systems because for a transaction to be valid, it must be approved by the participants of the network. In the case of our proposal, the sending of erroneous data is detected in the verification process, even timestamp can be added to the data to mitigate this attack [80].
- Remote code execution attacks: This attack occurs when an attacker inserts code into the system to execute it at will. Thanks to remote attestation or trusted boot mechanisms, the TPM can check at runtime if malicious software is running [81].
- Sniffing: Also known as Man-in-the-Middle attacks. These attacks are solved thanks to encrypted communications over secure channels in which the keys are stored in the TPM. If an attacker is sniffing the channel, all he will see is the encrypted data and will not be able to decrypt it because the keys are securely stored in the TPM [82].
- Brute force attacks: Through trying combinations in the seed of a key generation, the attacker can find out a cryptographic key. Using TRNG implemented in the TPM, thanks to the high entropy of key generation, a brute force attack becomes impossible and very expensive [50].
- Impersonation attacks: An attacker can obtain the cryptographic keys of a user from the Blockchain and impersonate him, by storing the keys inside the TPM. These keys cannot be extracted, so an attacker will never be able to replace the user identity [52].
- Malware attack: Through a security breach or a peripheral, the attacker can introduce malware into the device. Mechanisms such as the trusted boot and the remote attestation prevent this kind of attacks [83].

Among all these types of attacks, software attacks are more common. Since the TPM is a standardized device by the TCG,

it includes different mechanisms that a normal HSM does not support: it adds remote attestation and trusted boot to the device. In this case, as the RPI is an operating system supported platform both operations are supported. Trusted boot in the startup of the system helps to check the integrity of the device from the beginning. This integrity is checked also at run time using the remote attestation mechanism. This is a big step forward compared to HSMs, as it also provides mechanisms that check the state of the system during runtime. In addition, it is possible to establish TLS connections between devices, client and server. As a Blockchain is a decentralized network, this connection is performed between the nodes. The use of TPM reinforces the security of IoT devices and offers a high level of robustness to prevent these attacks through the mechanisms it implements.

V. CONCLUSION

This paper proposes combining TPM and PB technologies such as HLF for building a trusted IoT node. Indeed, procedures for the interaction of TPM and HLF nodes have been presented. Furthermore, the proof of concept presented in this paper shows how, at a higher level, an IoT node interacts with the HLF Blockchain. The performance benchmark conducted in this paper sheds light on the possible uses of TPMs in the IoT world. As it has been shown, integration of TPM and Blockchain provides many advantages when interacting with each other: from the interaction at the operations level, such as signature and enrollment, to the applicability in industrial environments, adding the main characteristics that they bring as a technology.

Using this proof of concept as a main element, reliable and robust Blockchain networks can be built in which different attacks are mitigated. Future promising steps lie in the application of these two technologies together in a real IIoT environment, e.g. logistics or smart grids.

Indeed, Blockchain brings a wealth of benefits to industrial environments. Allows interoperability among enterprises, saving cost, eliminating bureaucracy, etc. These advantages added to the use of hardware secure elements makes the system fully robust, and enables secure end to end communication between different components through the Blockchain. The joint applicability of these two components in an industrial environment generates great added value and means that all monitored processes have a root of trust in the extraction of data and a root of immutability. The use of HSMs in Blockchain networks results in a very fruitful combination as on the one hand the keys that are stored and generated within the HSM cannot be extracted and, on the other hand, they add a layer of security when obtaining data from the devices located at the edge. In addition, using Hyperledger Fabric as a Blockchain network, improves latency and scalability in transaction approval, creating a consortium where privacy and authentication are the main features which makes it suitable for industrial environments, creating an added value compared to traditional Blockchain networks, such as Ethereum or Bitcoin. All these considerations allow to augur a great future for these two technologies to go hand in hand thanks to the great advantages they bring together and their promising future in industrial environments. Finally, it should be noted that the proof of concept as well as the performance analysis testify to the seamless integration and future applicability in IIoT as the minimum security unit of a built network.

REFERENCES

[1] H. Boyes, B. Hallaq, J. Cunningham, and T. Watson, "The industrial Internet of Things (IIoT): An analysis framework," *Comput. Ind.*, vol. 101, pp. 1–12, Oct. 2018.

[2] A.-R. Sadeghi, C. Wachsmann, and3 M. Waidner, "Security and privacy challenges in industrial Internet of Things," in *Proc. 52nd Annu. Design Autom. Conf.*, Jun. 2015, pp. 1–6.

[3] S. C. Ergen, "ZigBee/IEEE 802.15. 4 summary," Univ. California, Berkeley, Berkeley, CA, USA, Tech. Rep., Sep. 2004, p. 11, vol. 10, no. 17.

[4] A. Semtech, "An1200. 22 LoRa modulation basics," *Semtech Appl. Note*, to be published.

[5] O. Standard. (2014). *Mqtt Version 3.1. 1*. [Online]. Available: http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/os/mqtt-v3.1.1-os.html

[6] P. S. Munoz, N. Tran, B. Craig, B. Dezfouli, and Y. Liu, "Analyzing the resource utilization of AES encryption on IoT devices," in *Proc. Asia–Pacific Signal Inf. Process. Assoc. Annu. Summit Conf. (APSIPA ASC)*, Nov. 2018, pp. 1200–1207.

[7] J. Mades, G. Ebelt, B. Janjic, F. Lauer, C. C. Rheinländer, and N. Wehn, "TLS-level security for low power industrial IoT network infrastructures," in *Proc. Design, Autom. Test Eur. Conf. Exhib. (DATE)*, 2020, pp. 1720–1721.

[8] A. K. Goel, A. Rose, J. Gaur, and B. Bhushan, "Attacks, countermeasures and security paradigms in IoT," in *Proc. 2nd Int. Conf. Intell. Comput., Instrum. Control Technol. (ICICICT)*, vol. 1, 2019, pp. 875–880.

[9] G. Loukas, *Cyber-Physical Attacks: A Growing Invisible Threat*. Oxford, U.K.: Butterworth-Heinemann, 2015.

[10] I. Bashir, *Mastering Blockchain*. Birmingham, U.K.: Packt, 2017.

[11] A. Sunyaev, "Distributed ledger technology," in *Internet Computing*. Cham, Switzerland: Springer, 2020, pp. 265–299.

[12] D. C. Mills, "Distributed ledger technology in payments, clearing, and settlement," Tech. Rep., 2016.

[13] H. Natarajan, S. Krause, and H. Gradstein, "Distributed ledger technology and blockchain," Tech. Rep., 2017.

[14] B. Cao, Y. Li, L. Zhang, L. Zhang, S. Mumtaz, Z. Zhou, and M. Peng, "When Internet of Things meets blockchain: Challenges in distributed consensus," *IEEE Netw.*, vol. 33, no. 6, pp. 133–139, Nov. 2019.

[15] A. Gorkhali, L. Li, and A. Shrestha, "Blockchain: A literature review," *J. Manag. Anal.*, vol. 7, no. 3, pp. 321–343, 2020.

[16] Y. E. H. Shehadeh and D. Hogrefe, "A survey on secret key generation mechanisms on the physical layer in wireless networks," *Secur. Commun. Netw.*, vol. 8, no. 2, pp. 332–341, 2015.

[17] K. Mayes and K. Markantonakis, "Secure smart embedded devices, platforms and applications," Tech. Rep., 2014.

[18] P. Tasatanattakool and C. Techapanupreeda, "Blockchain: Challenges and applications," in *Proc. Int. Conf. Inf. Netw. (ICOIN)*, 2018, pp. 473–475.

[19] D. G. W. Qin, U. N. Sujit, L. Jie, and T. Singh, "Vulnerabilities and attacks on PKI," in *CS2107-Semester IV 2014–2015*. 2015, p. 45.

[20] T. Geng and Y. Du, "Applying the blockchain-based deep reinforcement consensus algorithm to the intelligent manufacturing model under Internet of Things," *J. Supercomput.*, pp. 1–23, 2022.

[21] R. Huo, S. Zeng, Z. Wang, J. Shang, W. Chen, T. Huang, S. Wang, F. R. Yu, and Y. Liu, "A comprehensive survey on blockchain in industrial Internet of Things: Motivations, research progresses, and future challenges," *IEEE Commun. Surveys Tuts.*, vol. 24, no. 1, pp. 88–122, 1st Quart., 2022.

[22] G. Srivastava, J. Crichigno, and S. Dhar, "A light and secure healthcare blockchain for IoT medical devices," in *Proc. IEEE Can. Conf. Electr. Comput. Eng. (CCECE)*, May 2019, pp. 1–5.

[23] P. K. Sharma, N. Kumar, and J. H. Park, "Blockchain-based distributed framework for automotive industry in a smart city," *IEEE Trans. Ind. Informat.*, vol. 15, no. 7, pp. 4197–4205, Jul. 2018.

[24] S. Bulbul and G. Ince, "Blockchain-based framework for customer loyalty program," in *Proc. 3rd Int. Conf. Comput. Sci. Eng. (UBMK)*, Sep. 2018, pp. 342–346.

[25] C.-L. Chen, Y.-Y. Deng, W. Weng, M. Zhou, and H. Sun, "A blockchain-based intelligent anti-switch package in tracing logistics system," *J. Supercomput.*, vol. 77, no. 7, pp. 7791–7832, Jul. 2021.
[26] G. Perboli, S. Musso, and M. Rosano, "Blockchain in logistics and supply chain: A lean approach for designing real-world use cases," *IEEE Access*, vol. 6, pp. 62018–62028, 2018.
[27] S.-C. Oh, M.-S. Kim, Y. Park, G.-T. Roh, and C.-W. Lee, "Implementation of blockchain-based energy trading system," *Asia–Pacific J. Innov. Entrepreneurship*, vol. 11, no. 3, pp. 322–334, Dec. 2017.
[28] P. Rani, P. Kaur, V. Jain, J. Shokeen, and S. Nain, "Blockchain-based IoT enabled health monitoring system," *J. Supercomput.*, pp. 1–25, May 2022.
[29] G. Wood, "Ethereum: A secure decentralised generalised transaction ledger," *Ethereum Project Yellow Paper*, vol. 151, pp. 1–32, Apr. 2017.
[30] L. S. Sankar, M. Sindhu, and M. Sethumadhavan, "Survey of consensus protocols on blockchain applications," in *Proc. 4th Int. Conf. Adv. Comput. Commun. Syst. (ICACCS)*, Jan. 2017, pp. 1–5.
[31] G. Greenspan. (2015). *Multichain Private Blockchain-White Paper*. [Online]. Available: http://www.multichain.com/download/MultiChain-White-Paper.pdf
[32] J. Morgan, *Quorum Whitepaper*. New York, NY, USA: JP Morgan Chase, 2016.
[33] E. Androulaki and A. Barger, "Hyperledger fabric: A distributed operating system for permissioned blockchains," in *Proc. 13th EuroSys Conf.*, 2018, pp. 1–15.
[34] A. Everspaugh, Y. Zhai, R. Jellinek, T. Ristenpart, and M. Swift, "Not-so-random numbers in virtualized Linux and the whirlwind RNG," in *Proc. IEEE Symp. Secur. Privacy*, May 2014, pp. 559–574.
[35] E. Stark, M. Hamburg, and D. Boneh, "Symmetric cryptography in Javascript," in *Proc. Annu. Comput. Secur. Appl. Conf.*, Dec. 2009, pp. 373–381.
[36] M. Herrero-Collantes and J. C. Garcia-Escartin, "Quantum random number generators," *Rev. Mod. Phys.*, vol. 89, no. 1, Feb. 2017, Art. no. 015004.
[37] N. Kamoun, L. Bossuet, and A. Ghazel, "A masked correlated power noise generator use as a second order DPA countermeasure to secure hardware AES cipher," in *Proc. ICM*, 2011, pp. 1–5.
[38] J.-W. Lee, J.-H. Hsiao, H.-C. Chang, and C.-Y. Lee, "An efficient DPA countermeasure with randomized Montgomery operations for DF-ECC processor," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 59, no. 5, pp. 287–291, May 2012.
[39] M. Sabt, M. Achemlal, and A. Bouabdallah, "Trusted execution environment: What it is, and what it is not," in *Proc. IEEE Trustcom/BigDataSE/ISPA*, vol. 1, Aug. 2015, pp. 57–64.
[40] P. Jauernig, A.-R. Sadeghi, and E. Stapf, "Trusted execution environments: Properties, applications, and challenges," *IEEE Secur. Privacy*, vol. 18, no. 2, pp. 56–60, Mar. 2020.
[41] R. Anderson and M. Kuhn, "Tamper resistance-a cautionary note," in *Proc. 2nd Usenix Workshop Electron. Commerce*, vol. 2, 1996, pp. 1–11.
[42] S. Mavrovouniotis and M. Ganley, "Hardware security modules," in *Secure Smart Embedded Devices, Platforms and Applications*. Springer, 2014, pp. 383–405.
[43] S. Bajikar, "Trusted platform module (TPM) based security on notebook PCS-white paper," *Mobile Platforms Group Intel Corp.*, vol. 1, p. 20, Jan. 2002.
[44] *Trusted Computing Group*. Accessed: Jul. 21, 2022. [Online]. Available: https://trustedcomputinggroup.org
[45] *Trusted Platform Module Library Part 1: Architecture*. Accessed: Jul. 21, 2021. [Online]. Available: https://trustedcomputinggroup.org/wp-content/uploads/TCG_TPM2_r1p59_Part1_Architecture_pub.pdf
[46] *Trusted Platform Module Library Part 1: Structures*. Accessed: Jul. 21, 2022. [Online]. Available: https://trustedcomputinggroup.org/wp-content/uploads/TPM-Rev-2.0-Part-2-Structures-01.38.pdf
[47] C. Mitchell, Ed., *Trusted Computing*. Piscataway, NJ, USA: IEE Press, 2005.
[48] W. Arthur, D. Challener, and K. Goldman, "Platform configuration registers," in *A Practical Guide to TPM 2.0*. Springer, 2015, pp. 151–161.
[49] A. Tomlinson, "Introduction to the TPM," in *Smart Cards, Tokens, Security and Applications*. Cham, Switzerland: Springer, 2017, pp. 173–191.
[50] W. Arthur, D. Challener, and K. Goldman, *A Practical Guide to Tpm 2.0: Using the New Trusted Platform Module in the New Age of Security*. Cham, Switzerland: Springer, 2015.
[51] L. Gu, X. Ding, R. H. Deng, B. Xie, and H. Mei, "Remote attestation on program execution," in *Proc. 3rd ACM workshop Scalable Trusted Comput. (STC)*, 2008, pp. 11–20.
[52] B. Parno, "The trusted platform module (TPM) and sealed storage," *TPM Document*, Jun. 2007.
[53] C. Brunner, F. Knirsch, A. Unterweger, and D. Engel, "A comparison of blockchain-based PKI implementations," in *Proc. 6th Int. Conf. Inf. Syst. Secur. Privacy*, 2020, pp. 333–340.
[54] A. Miller, "Permissioned and permissionless blockchains," in *Blockchain for Distributed Systems Security*. 2019, pp. 193–204.
[55] I. Haq and O. Muselemu, "Blockchain technology in pharmaceutical industry to prevent counterfeit drugs," *Int. J. Comput. Appl.*, vol. 180, no. 25, pp. 8–12, Mar. 2018.
[56] C. Lin, D. He, X. Huang, K. K. R. Choo, and A. V. Vasilakos, "BSEIN: A blockchain-based secure mutual authentication with fine-grained access control system for Industry 4.0," *J. Netw. Comput. Appl.*, vol. 116, pp. 42–52, Aug. 2018.
[57] Z. Li, J. Kang, R. Yu, D. Ye, Q. Deng, and Y. Zhang, "Consortium blockchain for secure energy trading in industrial Internet of Things," *IEEE Trans. Ind. Informat.*, vol. 14, no. 8, pp. 3690–3700, Aug. 2017.
[58] C. Cachin, "Architecture of the hyperledger blockchain fabric," in *Proc. Workshop Distrib. Cryptocurrencies Consensus Ledgers*, Chicago, IL, USA, Jul. 2016, vol. 310, no. 4, pp. 1–4.
[59] C. Cachin, M. V. Sorniotti, and T. Weigold, "Blockchain, cryptography, and consensus," IBM, Res., Zürich, Switzerland, Tech. Rep. 2016, 2016.
[60] D. Boneh, "Twenty years of attacks on the RSA cryptosystem," *Notices AMS*, vol. 46, no. 2, pp. 203–213, 1999.
[61] D. Mahto and D. K. Yadav, "RSA and ECC: A comparative analysis," *Int. J. Appl. Eng. Res.*, vol. 12, no. 19, pp. 9053–9061, 2017.
[62] J. Lopez and R. Dahab, "An overview of elliptic curve cryptography," Tech. Rep., 2000.
[63] J. Bos, J. Halderman, N. Heninger, J. Moore, M. Naehrig, and E. Wustrow, "Elliptic curve cryptography in practice in international conference on financial cryptography and data security," Tech. Rep., 2014.
[64] D. John Cook. *Curve NIST P-384*. Accessed: Aug. 23, 2022. [Online]. Available: https://www.johndcook.com/blog/2019/05/11/elliptic-curve-p-384/
[65] C. F. Kerry and C. R. Director, "FIPS PUB 186–4 federal information processing standards publication digital signature standard (DSS)," Tech. Rep., 2013.
[66] D. Jablon, "IEEE P1363 standard specifications for public-key cryptography," in *Proc. IEEE NIST Key Manag. Workshop CTO Phoenix Technol. Treasurer*, Nov. 2001, pp. 1–26.
[67] M. Nystrom and B. Kaliski, "PKCS# 10: Certification request syntax specification version 1.7," Tech. Rep., 2000.
[68] R. Griffin and V. Fenwick, "PKCS# 11 cryptographic token interface base specification version 2.40," *OASIS Open*, 2015.
[69] *Release 5.2*. Accessed: Sep. 23, 2022. [Online]. Available: https://github.com/tpm2-software/tpm2-tools/releases/tag/5.2
[70] *Release 2.4.0*. Accessed: Sep. 23, 2022. [Online]. Available: https://github.com/tpm2-software/tpm2-abrmd/releases/tag/2.4.0
[71] *Openssl 3.0*. Accessed: Sep. 23, 2022. [Online]. Available: https://wiki.openssl.org/index.phpOpenSSL_3.0
[72] *IBM's Software TPM 2.0*. Accessed: Aug. 8, 2023. [Online]. Available: https://sourceforge.net/projects/ibmswtpm2/
[73] P. S. Anjana, S. Kumari, S. Peri, S. Rathor, and A. Somani, "An efficient framework for optimistic concurrent execution of smart contracts," in *Proc. 27th Euromicro Int. Conf. Parallel, Distrib. Netw.-Based Process. (PDP)*, Feb. 2019, pp. 83–92.
[74] Q. Ge, Y. Yarom, D. Cock, and G. Heiser, "A survey of microarchitectural timing attacks and countermeasures on contemporary hardware," *J. Cryptograph. Eng.*, vol. 8, no. 1, pp. 1–27, Apr. 2018.
[75] J. Deogirikar and A. Vidhate, "Security attacks in IoT: A survey," in *Proc. Int. Conf. I-SMAC (IoT Social, Mobile, Anal. Cloud) (I-SMAC)*, 2017, pp. 32–37.
[76] S. Skorobogatov, "Physical attacks and tamper resistance," in *Introduction to Hardware Security and Trust*. Cham, Switzerland: Springer, 2012, pp. 143–173.
[77] R. Anderson and M. Kuhn, "Low cost attacks on tamper resistant devices," in *Proc. Int. Workshop Security Protocols*. Cham, Switzerland: Springer, 1997, pp. 125–136.
[78] Z. Du, X. Li, and K. Shen, "Trusted firmware services based on TPM," in *Proc. Int. Conf. Trusted Syst.* Cham, Switzerland: Springer, 2009, pp. 227–235.

[79] P. Choi and D. K. Kim, "Design of security enhanced TPM chip against invasive physical attacks," in *Proc. IEEE Int. Symp. Circuits Syst.*, May 2012, pp. 1787–1790.
[80] P. Ramanan, D. Li, and N. Gebraeel, "Blockchain-based decentralized replay attack detection for large-scale power systems," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 52, no. 8, pp. 4727–4739, Aug. 2021.
[81] H. Tan, G. Tsudik, and S. Jha, "MTRA: Multiple-tier remote attestation in IoT networks," in *Proc. IEEE Conf. Commun. Netw. Secur. (CNS)*, Oct. 2017, pp. 1–9.
[82] J. Choi, B. Ahn, G. Bere, S. Ahmad, H. A. Mantooth, and T. Kim, "Blockchain-based man-in-the-middle (MITM) attack detection for photovoltaic systems," in *Proc. IEEE Design Methodologies Conf. (DMC)*, Jul. 2021, pp. 1–6.
[83] D. Li, Y. Zhang, J. Cui, D. Liu, Y. Sun, Z. Guan, and X. Wang, "Remote audit scheme of embedded device software based on TPM," in *Proc. IEEE 8th Int. Conf. Big Data Secur. Cloud (BigDataSecurity), IEEE Int. Conf. High Perform. Smart Comput. (HPSC), IEEE Int. Conf. Intell. Data Secur. (IDS)*, May 2022, pp. 61–66.

ANTONIO J. CABRERA-GUTIÉRREZ was born in Noalejo, Jaén, Spain. He received the B.Eng. and M.Eng. degrees in computer engineering from the University of Granada, Granada, Spain, in 2018 and 2019, respectively, where he is currently pursuing the Ph.D. degree in secure and reliable communication protocols in the Industrial IoT networks. In 2019, he joined Infineon Technologies AG, Neubiberg, Germany, as the Ph.D. Candidate with the University of Granada. He is involved in different research projects covering topics related to the Industrial Internet of Things, security, cryptography, and virtualization environments. His current research interests include hardware security, blockchain technologies, and the IoT embedded systems.

ENCARNACIÓN CASTILLO received the M.Sc. and Ph.D. degrees in electronic engineering from the University of Granada, Granada, Spain, in 2002 and 2008, respectively. From 2003 to 2005, she was a Research Fellow with the Department of Electronics and Computer Technology, University of Granada, where she is currently a Tenured Professor. During a Research Fellowship, she carried out part of her research with the Department of Electrical and Computer Engineering, Florida State University, Tallahassee, FL, USA. Her work has led to the publication of 42 papers in indexed journals, more than 60 contributions to international conferences, the contribution to 12 projects in the several national and regional programs, 11 technology-transfer contracts, and two patents in Spain. Her current research interests include VLSI and FPL signal processing systems, smart instrumentation for biosignal processing, new sensors and methods for sensing, and the design of cryptoprocessors based on elliptic-curve cryptography for its application to the IoT devices. She serves as a reviewer and a guest editor for several journals.

ANTONIO ESCOBAR-MOLERO received the Ph.D. degree from the RWTH Aachen University, in 2020, with a work focused on the dependable IoT wireless networks. He is a Research and Development Engineer at Infineon Technologies AG, working in European funding projects related to the IoT, cybesercurity, and AI. He believes decentralization technologies, such as distributed ledger technologies are today's best route toward a fair and robust internet.

JOSÉ A. ÁLVAREZ-BERMEJO is a Tenured Professor at the Dpto. Informática, Universidad de Almería, Spain. His experience in the private industrial sector led him to get a position at the Department of Computer Architecture and Electronics, Universidad de Almería, in 2001, where he serves as a Tenured Professor. His teaching has been mainly at the College of Engineering, University of Almeria. His research career has been carried out at the Supercomputing: Algorithms Group, from 2001 to 2018 and in the FQM-211 Categories, computation and ring theory, researching in cybersecurity until today. He strongly collaborates with the multidisciplinary research group ECSens. His research is mainly devoted to cybersecurity and cryptographic protocols. He was previously focused in the supercomputing scenario and human–computer interaction (HCI), where he was awarded twice with national awards mentions. All his work led to the publication of 26 papers in indexed journals (Q1and Q2), more than 70 contributions to international conferences, the supervision of one Ph.D. dissertation, several technology-transfer contracts, and three patents. His teaching activity led to have more than 30 papers and six books. He is currently a member of the European Cybersecurity Training Education Group, where he develops training for law enforcement agencies across European state members, collaborating with CEPOL, EUROPOL, OSCE, and other international institutions focused in securing the digital world. He is also a member of the ECTEG Project Decrypt. For more details: https://wpd.ugr.es/ecsens/

DIEGO P. MORALES received the B.Sc., M.Eng., and Ph.D. degrees in electronics engineering from the University of Granada, in 2001 and 2011, respectively. Since 2001, he has been an Associate Professor with the Department of Computer Architecture and Electronics, University of Almeria, before joining the Department of Electronics and Computer Technology, University of Granada, in 2006, where he is currently works as a Professor (Tenured). He is the Co-Founder of the Biochemistry and Electronics as Sensing Technologies (BEST) Research Group, University of Granada. He has coauthored more than 80 scientific contributions. His current research interests include low-power energy conversion, energy harvesting for wearable sensing systems, and new materials for electronics and sensors.

LUIS PARRILLA received the M.Sc. degree in physics (majoring in electronics), the M.A.Sc. degree in electronic engineering, and the Ph.D. degree in physics from the University of Granada, Granada, Spain, in 1993, 1995, and 1997, respectively. In 1995, he joined the Department of Electronics and Computer Technology, University of Granada, where he has been serving as a Professor, since 2000. He has authored more than 70 technical papers in international journals and conferences. His current research interests include the protection of IP cores on VLSI and FPGA-based systems, development of high-performance arithmetic and algebraic circuits for the IoT, cryptographic applications, the design of specific architectures for cryptographic processors, and biosignal processing. He serves as a reviewer and a guest editor for several journals.