Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 263 (2025) 418–425

Procedia Computer Science

www.elsevier.com/locate/procedia

International Conference on Industry Sciences and Computer Science Innovation

Fostering a Framework for Secure Data Transfer From IoT Devices To Cloud

Beatriz Lopes de Oliveira[a], Rui Alves[a,*], Tiago Pedrosa[b]

[a]*Instituto Politécnico de Bragança, Campus de Santa Apolónia, 5300-253 Bragança, Portugal*
[b]*CeDRI, SusTEC, Instituto Politécnico de Bragança, 5300-253 Bragança, Portugal*

Abstract

The Internet of Things (IoT) has indeed introduced a new era of connectivity and data-driven insights, profoundly altering everyday activities execution. The data collected by IoT devices, ranging from simple sensors to sophisticated systems integrated within urban infrastructure, gather and transmit data on a large scale, supporting a wide spectrum of applications including individual health metrics, home energy consumption, traffic flow, and industrial operations. The massive volume and diversity of data captured offer valuable insights and optimization opportunities but also create new crucial challenges, particularly within the information security domain. The integrity and reliability of this data are key for maintaining the accuracy, consistency, and trustworthiness of the information produced. These aspects are especially critical considering that, the generated data often involves sensitive and personal information utilized across various applications and systems. Ensuring the data's integrity and reliability is not just about maintaining its quality but also about securing the privacy and security of individuals. This paper presents a programming framework, using asymmetric cryptography, that allows the transfer of generated data from IoT devices to the Cloud, mitigating some of the most common attack surfaces.

Keywords: IoT; Cybersecurity; Cryptography; Authenticity; Integrity; Messages

1. Introduction

Currently, the Internet of Things (IoT) [5] represents a revolutionary paradigm in the domain of information and communication technologies, characterized by the widespread presence of a variety of devices, such as sensors, actuators, and smartphones, all interconnected through the Internet. This concept offers a set of advantages in daily life, increasing comfort, efficiency, and safety across multiple aspects of human activity. In sectors such as healthcare, wearable IoT devices can monitor vital signs in real-time, providing critical data either individuals and healthcare professionals and facilitating early detection of potential health issues. Furthermore, IoT applications in smart cities

* Corresponding author.
E-mail addresses: a47333@alunos.ipb.pt (Beatriz Lopes de Oliveira)., rui.alves@ipb.pt (Rui Alves)., pedrosa@ipb.pt (Tiago Pedrosa).

1877-0509
10.1016/j.procs.2025.07.051

improve urban living by optimizing traffic management, reducing energy consumption, and enhancing public safety through intelligent surveillance systems.

Moreover, to support this sort of environment, concepts such as ”Edge intelligence” appeared in recent years, which has allowed the optimization of IoT systems, processing and analyzing information from the data generated by them. In short, Edge intelligence [4] is the integration of artificial intelligence (AI) capabilities directly into edge computing environments, where data processing occurs near the source of data generation, rather than in centralized data centers. The main points are minimizing latency, reducing bandwidth use, and enhancing real-time decision-making by allowing devices and local systems to analyze and act on data instantly.

However, the effectiveness of edge intelligence [6], like all data-driven technologies, is inherently dependent on the quality, reliability, and authenticity of the input data. When edge devices receive inaccurate, manipulated, or ”fake” data, the integrity of the decision-making process is compromised, potentially leading to inaccurate outcomes or actions, which is especially critical, in crucial applications like traffic management or patient care.

To ensure the authenticity and accuracy of data becomes paramount, this paper proposes a programming framework to protect the data from the point of collection through to its final destination, thereby preserving the trustworthiness and reliability of the information generated. Using concepts such as asymmetric cryptography to encrypt and sign the data, blockchain, and sockets BSD, the main goal of this paper is to prevent unauthorized access, interception, or tampering to avoid compromising the confidentiality, integrity, and availability of the data generated on IoT Devices.

The rest of the paper is organized as follows: section 2 presents the threat model of the current work; section 3 presents the proposed solution; the related work is present in section 4; section 5 concludes the paper and defines some directions for future work.

2. Threat Model

The Internet of Things (IoT) plays a pivotal role in the modern world by interconnecting devices and enabling them to communicate, share data, and perform tasks autonomously. Its significance lies in its ability to revolutionize various sectors, from healthcare and agriculture to manufacturing and transportation. However, despite the advantages, the security of these systems is still a crucial topic to solve, especially, in contexts, where there are more attack vectors beyond the network.

A significant part of the current IoT systems are installed in physical buildings such as hospitals, government units, or even in homes. Despite the extra security mechanisms, there are physical barriers (e.g. fences, gates, and security cameras), which is an obstacle to malicious activity. Therefore, an attacker to gain access to a resource within these systems may first try other attack vectors [11] rather than start by physical access to resources. As it is possible to see in figure 1, physical cyberattacks such as firmware tampering, manipulation of generated data, and unauthorized access are much easier to perform if the IoT system is installed in a remote area (e.g agricultural remote lands, nature monitoring systems, etc.) than if it is installed inside a building (e.g hospital, home, etc.) because, in the second one, there is greater flexibility to use stricter physical barriers and security controls.

In this way, the design of IoT systems that are used, for example, in farmland, environmental monitoring, wildlife conservation, or in others where there are no physical barriers and/or restricted security controls, should be carefully analyzed to minimize as much as possible the consequences of a possible attack.

Another important aspect of the motivation of this work is reinforcing security when the data is stored. It is crucial to ensure the integrity, and reliability of the data generated by IoT devices, from creation to storing in the cloud (in transit state). However, in some execution scenarios, it is necessary to ensure that data are secure at rest state and only can be decrypted and accessed by authorized persons when specific events happen in the flow chart of the application model concerned.

To sum up, the impossibility of putting physical barriers in some activity sectors is an urgent topic to solve because the reliability and integrity of the data collected may be compromised easily without the hardening mechanisms appropriate. Thereby, the main motivation for this work is to provide tools that enable the reinforcement of security levels in IoT systems more specifically in the execution environments specified previously. The section 4 presents the similar solutions proposed in this paper found in the literature of the area.

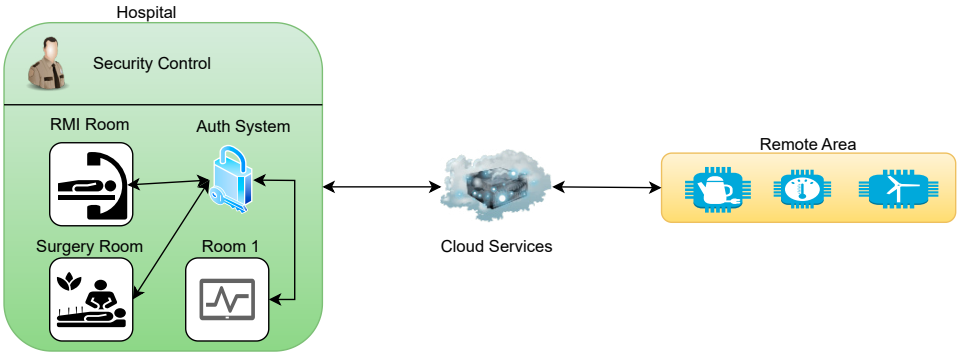

Fig. 1. Threat Model

3. Proposed Solution

The figure 2 presents an overview of the framework proposed in this work. Over the framework design process, two main parts have been considered to reinforce the security level and avoid single points of failure. These parts are described by: the secure transfer of messages between IoT devices and the cloud provider; the secure storage of temporary private keys.

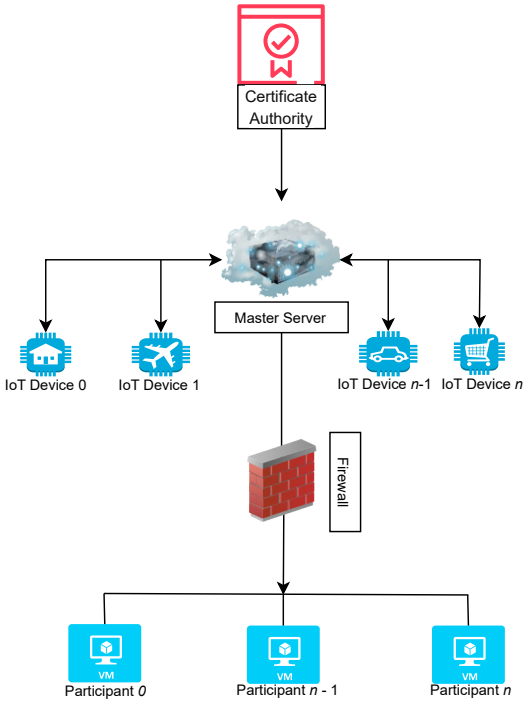

Fig. 2. Proposed Solution General Overview

The main concept of this framework is *message cycle*, which is the time interval, in seconds, when an authenticated IoT device can send data to the cloud through a secure and reliable channel. To allow this, the solution presented has been built on top of the Certificate Authority which is responsible for signing and validating all certificates; the Master Server where is accommodated the blockchain records (Listing 1) of all IoT devices and management of all operations in exchange message process; IoT device(s) with application software and its private key generated at the register process; a set of virtual machines (participants) that save the shares of the generated temporary private keys.

Regarding its operation, three important processes have been defined: the authentication of the IoT device (section 3.1); the request of a temporary key pair (section 3.2); secure distribution of the temporary private keys (section 3.3).

3.1. Authentication of the IoT Device

The first assumption to use this framework lies that all IoT devices must be previously authenticated before being put in the execution environment. The figure 3 represents the sequence diagram of the authentication process

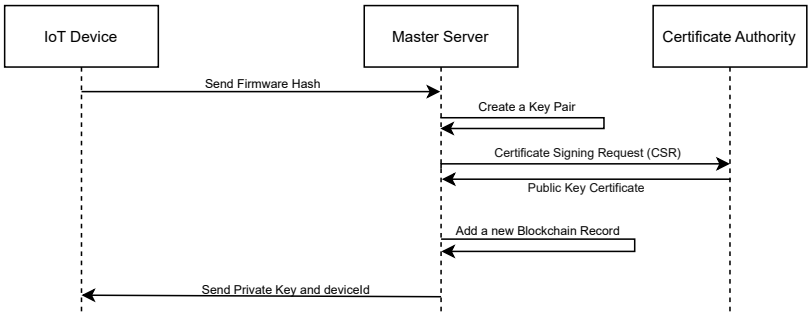

Fig. 3. Sequence diagram of the authentication process.

The process starts on the IoT Device which sends its installed firmware hash to the Master Server. This hash is generated by the SHA-3 function, a versatile, secure, and efficient cryptographic hash function, making it an excellent choice for IoT devices' diverse and resource-constrained environments [10].

After receiving the hash, the Master Server creates a pair of keys and sends them to the certificate authority to be signed. The keys are created using the ECC (Elliptic Curve Cryptography). Elliptic Curve Cryptography (ECC) [8] is highly favored for Internet of Things (IoT) applications because of its capability to offer robust security through comparatively smaller key sizes than conventional algorithms like RSA.

Listing 1. IoTDeviceRegistry Contract

```
contract IoTDevice {

    struct CrossChainReference {
        bytes32 referenceId;
        uint256 nonce;
        uint256 expiresAt;
    }

    struct Device {
        bytes32 deviceId;
        bytes32 firmwareHash;
        bytes publicKey;
        uint256 registeredAt;
        uint256 deviceExpiresAt;
        CrossChainReference[] crossChainRef;
    }
    (...) // more code lines
}
```

After signing the keys, a new entry is added to the *IoTDevice* smart contract represented by Listing 1, which is built on top of the Ethereum blockchain [2]. The role of this contract is the *Device struct*, which stores all necessary information about an IoT device such as the unique identifier (*deviceId*); the current firmware's hash (*firmwareHash*); the device's public key (*publicKey*); timestamps for registration and expiration to manage the device's lifecycle. The field *crossChainRef* has been added to support the message cycles of each IoT device. This enhances the traceability and verification processes for IoT device data, ensuring higher security (see 3.2). The authentication process ends when is sent the generated private key and the randomly generated device identifier to the IoT device (*deviceId*).

To sum up, it's relevant to highlight some design decisions, namely, the generation of the IoT device firmware hash and the hardening around the private key storage deployed in the IoT device. Hash generation mitigates some common attacks such as firmware tampering, replay, and spoofing. This measure is especially significant in IoT execution contexts, where it isn't possible to implement simple and basic security measures such as physical barriers. Regarding the private key safety, even if it's compromised, access to the system is denied, because the verification of the signature of the messages on the Server Master will fail since the private key doesn't match with the corresponding public key.

3.2. Request Temporary Key Pair

The diagram in figure 4 represents the procedure to generate the temporary keys for each message cycle. When a new message cycle starts, the client creates a message with: the device identifier and the current firmware hash and then generates the hash of this message using the SHA-3 function and signs it with the permanent private key. Next, the Master Server receives the message and validates the signature of the message, using the corresponding public key of the client. After receiving a valid signature, the *temporary key pair* is generated and sent to the certificate authority to be signed.

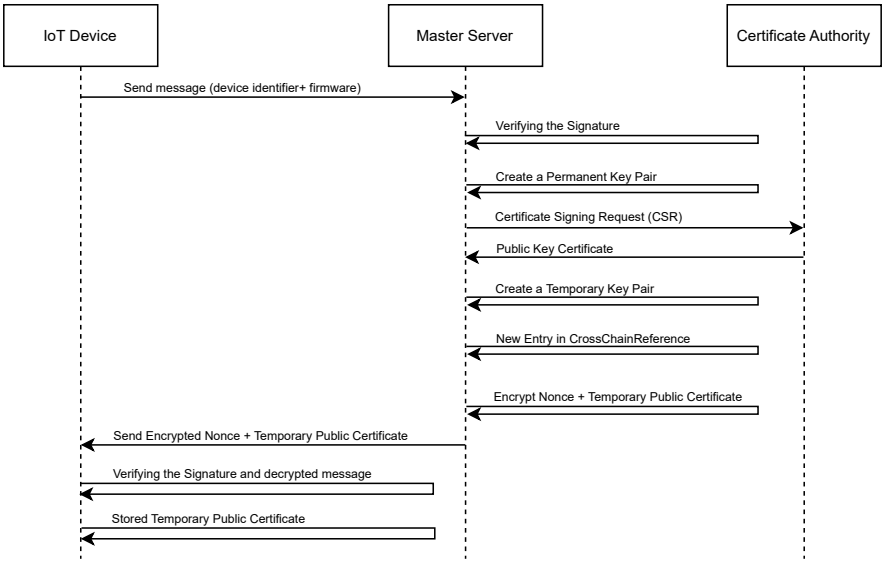

Fig. 4. Sequence Diagram of the Request Temporary Key Pair

When the public key certificate is received, a *nonce* parameter is generated and a new entry, in an array represented by *crossChainRef*, is added in *struct Device* of the corresponding client on *IoTDevice* smart contract. Finally, the master server creates a message with the generated nonce and the temporary public key to send to the client. Before sending, this message is signed with the client's permanent public key. The *CrossChainReference* structure acts as a bridge, linking the IoT devices and message cycles. Each *CrossChainReference* contains information pertinent to a specific blockchain of message cycles ensuring that cross-chain interactions are verifiable and secure.

After these steps are complete, all messages sent by the IoT client are encrypted by the temporary public key and stored on the Master Key during the validity of the keys. Each message contains a payload and a body. The payload

is composed of the device ID, the hash of the current firmware, and the nonce value of the message cycle. The body content is the application data, encrypted with the temporary public key. The message is signed with the permanent private key before it is sent. Finally, the master server only accepts the message and stores the data when the signature is valid and the concatenation of the nonce with firmware hash corresponds to the registered message cycle since each IoT Device only has a single active message cycle at the same time.

As a bottom line, the nonce parameter was added to mitigate attacks such as replay attacks and man-in-the-middle attacks. The generation of temporary keys for each messaging cycle and IoT device is another of the mitigation measures included. Therefore, if temporary keys are compromised, only information encrypted with these keys is damaged. The message decryption process can only be done by the restricted group of individuals with authorized access to private keys. The temporary private key storage process will be explained in section 3.3.

3.3. Distribution of the Temporary Private Keys

The sharing of temporary private keys is performed through the method Shamir's Secret Sharing as shown in the figure 2. Shamir's Secret Sharing [7] is a cryptographic method that divides a secret into parts, giving each participant its unique part, yet requiring a minimum number of parts to reconstruct the secret. The algorithm is based on polynomial interpolation in finite fields, leveraging the mathematical property that a polynomial of degree n-1 is uniquely determined by n points, ensuring that the secret remains secure as long as the threshold condition is met and protected against partial compromise.

This algorithm is supported by Public Key Infrastructure (PKI), where each participant, including the Master Server which distributes the secret shares, generates a unique public-private key pair, facilitating secure communication and authentication. The public keys are exchanged among the group for encrypting messages or shares, ensuring that only the intended recipient with the corresponding private key can decrypt them. Each participant's share is encrypted with their public key for secure distribution. The master uses his private key to sign the shares, providing a method for participants to verify the master's authenticity. Similarly, during the reconstruction of the secret, participants can authenticate each other through digital signatures, ensuring that only legitimate shares are used in the process.

In the design of the current solution was been defined a fixed number of participants (pa) to receive shares. The key pairs, necessary for to exchange of secure and authenticated messages between the participants and the master server, were previously created, and distributed then signed by the Certificate Authority (CA) that existed in the architecture.

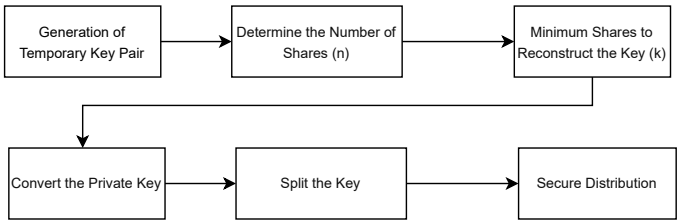

Fig. 5. Fluxogram of the split and distribution shares process using the SSS Algorithm.

The figure 5 describes the actions performed to split each private key generated in many shares. This process occurs when an IoT client requests the generation of a temporary key pair (start a new message cycle). The next step is determining the number of shares (n) and the minimum number of shares to reconstruct the original key (k). These values must be obtained by $n = \mathrm{pa} \cdot 0.7$ and $k = \mathrm{n} \cdot 0.8$. Before splitting the key, it is necessary to convert it to a suitable numeric format to be used by the scheme. Then, the solution is able to divide the key into n shares with a threshold of k, where each share should be a part of the key such that any k shares can reconstruct the private key but fewer than k shares reveal no information about the private key.

Thereby, the transfer of shares is executed with SFTP. In each participant set up, a user with write-only permissions, and the SSH service is configured only to allow the use of SFTP. The reliability and authenticity of the messages and shares exchanged are guaranteed by the key pairs generated for each participant. Due to the fact, that the user used by

the master only has write-only permissions, this component is no longer a single point of failure. Thus, in the case of master server compromise, the shares exchanged previously with participants are secure.

To sum up, using Shamir's Secret Sharing (SSS) mitigates several security risks. As each participant holds a share, useless on its own, but collectively a certain subset can reconstruct the secret. With this algorithm, it is possible to prevent single points of failure, reduce the risk of insider attacks since no single participant can uncover the secret alone, and guard against data loss and tampering because the secret remains intact unless a threshold number of shares is compromised or lost.

4. Related Work

This work [9], under the scope of the Internet of Medical Things (IoMT) concept, introduces a novel framework aimed at securely transmitting patient data to remote locations, regarding essential criteria for the management of electronic health records (EHR). Some differences were identified between the approach described above and the solution proposed in this paper: problem domain, the approach described is applied to health records and image processing.

The authors [14] introduce a novel approach integrating two-factor security authentication with key negotiation, leveraging chaotic mapping technology. This scheme intends to enhance communication security across devices, cloud servers, and edge devices, and at the same time, optimize computational and communication resources. In this approach, there are many similarities with the solution proposed in the current paper, however, it is not clear as firmware tampering attacks on remote IoT devices are mitigated. However, the usage of chaotic mapping technology is an important difference related to the solution presented in this paper.

This approach [13] suggests an authentication and identity tracking framework tailored for drone-assisted Internet of Vehicles (IoV), bolstered by Roadside Units (RSUs) with edge computing capabilities. Among other components, a Trusted Authority is used with a set of public and private keys similar to the current solution. Moreover, the usage of an asymmetric approach, in a cryptographic context is another similar point.

An efficient secure self-authenticating transfer protocol (SSATP) designed for communication between Edge and Fog nodes is suggested in this work [12]. This protocol is intended as a secure alternative for transporting CoAP (Constrained Application Protocol), replacing the traditionally used UDP and DTLS. Compared to the solution presented, use symmetric cryptography and the problem domain is more generic, applied to authentication between edge nodes and fog nodes and not remote IoT Devices.

The security framework proposed here [1] starts with the authentication of the user and IoT devices, followed by the activation of the linked IoT devices which then transmit data to the cloud server. To safeguard the transmission of data from IoT devices is being used employs a combination of Elliptic Curve Cryptography (ECC) and Genetic Algorithms (GA) for key generation. This is a very similar solution to that presented in this paper. However, there is no message cycle concept, the same key is used to encrypt data coming from authorized IoT devices, which means if that key is compromised all information may be threatened. Furthermore, there are no measures that mitigate IoT device firmware tampering attacks.

IoTAttest [3] facilitates the real-time verification of the authenticity and reliability of gathered data. Furthermore, this framework simplifies the expansion of the IoT system, allowing for the easy integration or detachment of components without compromising privacy or data integrity. This is another solution, with concepts similar to the one presented, but is not clear if it uses the asymmetric approach for data encryption and if there is a rotation of keys periodically.

5. Conclusion

Ensuring the integrity and privacy of data generated by IoT devices is today more than an urgent need. On one hand for the amount of data generated by these devices and on the other hand for the human dependency on cloud systems. The fact is that the concept of IoT is increasingly present in different sectors of society. Due to the different characteristics of many of them, it is not possible to ensure acceptable levels of information security. Therefore, creating solutions that reinforce the security of the data generated in IoT systems from its creation to its storage is

crucial, but more than that, is fundamental that these solutions maintain the same level of security regardless of their execution context.

This work, although still an ongoing project, presents a conceptual security architecture for data transfers to the cloud, for edge computing scenarios where data is sent in large quantities and often with sensitive information. Among many other points, the main goal of this solution is to ensure the same level of security regardless of its execution context, maintaining the integrity and reliability of the data generated by IoT devices immutable. It is also important to reinforce the possibility of integrating this solution with concepts such as homomorphic encryption which can be described as an approach that allows to perform operations and produce results without the need to decrypt the data. This integration is possible because the proposed framework requires that the data remain encrypted while stored, and can only be decrypted when certain events in the application flowchart occur.

In the current state of development, the solution is in the alpha prototype. Some tests have already been performed to validate the exchange of messages and the respective process of creating permanent and temporary keys. As a future work, the following points were left: analyze latency in data exchange and management of generated keys; perform penetration tests to verify the level of security offered by the solution; improve the security of private key distribution; analyze the feasibility of using this solution in execution contexts with limited network access.

Acknowledgements

This work was partial supported by national funds through FCT/MCTES (PIDDAC): CeDRI, UIDB/05757/2020 (DOI: 10.54499/UIDB/05757/2020) and UIDP/05757/2020 (DOI: 10.54499/UIDP/05757/2020); and SusTEC, LA/P/0007/2020 (DOI: 10.54499/LA/P/0007/2020).

References

[1] Ali, S., Anwer, F., 2024. Secure iot framework for authentication and confidentiality using hybrid cryptographic schemes. International Journal of Information Technology URL: https://doi.org/10.1007/s41870-024-01753-w, doi:10.1007/s41870-024-01753-w.

[2] Buterin, V., 2014. Ethereum: A next-generation smart contract and decentralized application platform. URL: https://ethereum.org/en/whitepaper/. accessed: [3 Mar 2024].

[3] Dirin, A., Oliver, I., Laine, T.H., 2023. A security framework for increasing data and device integrity in internet of things systems. Sensors 23. URL: https://www.mdpi.com/1424-8220/23/17/7532, doi:10.3390/s23177532.

[4] Hu, H., Jiang, C., 2020. Edge intelligence: Challenges and opportunities, in: 2020 International Conference on Computer, Information and Telecommunication Systems (CITS), pp. 1–5. doi:10.1109/CITS49457.2020.9232575.

[5] Kumar, S., Tiwari, P., Zymbler, M., 2019. Internet of things is a revolutionary approach for future technology enhancement: a review. Journal of Big Data 6, 111. URL: https://doi.org/10.1186/s40537-019-0268-2, doi:10.1186/s40537-019-0268-2.

[6] Molokomme, D.N., Onumanyi, A.J., Abu-Mahfouz, A.M., 2022. Edge intelligence in smart grids: A survey on architectures, offloading models, cyber security measures, and challenges. Journal of Sensor and Actuator Networks 11. URL: https://www.mdpi.com/2224-2708/11/3/47, doi:10.3390/jsan11030047.

[7] Nenov, L., Kassev, K., 2022. Security analysis of shamir's secret sharing, in: 2022 Seventh Junior Conference on Lighting (Lighting), pp. 1–4.

[8] Nita, S.L., Mihailescu, M.I., 2023. Elliptic curve-based query authentication protocol for iot devices aided by blockchain. Sensors 23. URL: https://www.mdpi.com/1424-8220/23/3/1371, doi:10.3390/s23031371.

[9] Parah, S.A., Kaw, J.A., Bellavista, P., Loan, N.A., Bhat, G.M., Muhammad, K., de Albuquerque, V.H.C., 2021. Efficient security and authentication for edge-based internet of medical things. IEEE Internet of Things Journal 8, 15652–15662. doi:10.1109/JIOT.2020.3038009.

[10] Sharma, N., Parveen Sultana, H., Singh, R., Patil, S., 2019. Secure hash authentication in iot based applications. Procedia Computer Science 165, 328–335. URL: https://www.sciencedirect.com/science/article/pii/S1877050920300508, doi:https://doi.org/10.1016/j.procs.2020.01.042. 2nd International Conference on Recent Trends in Advanced Computing ICRTAC -DISRUP - TIV INNOVATION , 2019 November 11-12, 2019.

[11] Tiwari, V., Dwivedi, 2016. Analysis of cyber attack vectors. doi:10.1109/CCAA.2016.7813791.

[12] Venčkauskas, A., Morkevicius, N., Jukavičius, V., Damaševičius, R., Toldinas, J., Grigaliūnas, Š., 2019. An edge-fog secure self-authenticable data transfer protocol. Sensors 19. URL: https://www.mdpi.com/1424-8220/19/16/3612, doi:10.3390/s19163612.

[13] Wu, F., Li, X., Luo, X., Gu, K., 2022. A novel authentication scheme for edge computing-enabled internet of vehicles providing anonymity and identity tracing with drone-assistance. Journal of Systems Architecture 132, 102737. URL: https://www.sciencedirect.com/science/article/pii/S1383762122002223, doi:https://doi.org/10.1016/j.sysarc.2022.102737.

[14] Zhu, W., Zhou, C., Jiang, L., 2024. A trusted internet of things access scheme for cloud edge collaboration. Electronics 13. URL: https://www.mdpi.com/2079-9292/13/6/1026, doi:10.3390/electronics13061026.