engineering proceedings

MDPI

Proceeding Paper

A Simulation Tool for Security in ZigBee-Based IoT Networks †

Marieta Yordanova *, Aydan Haka, Veneta Aleksieva and Hristo Valchanov

Faculty of Computer Sciences and Automation, Technical University of Varna, 9010 Varna, Bulgaria; aydin.mehmed@tu-varna.bg (A.H.); valeksieva@tu-varna.bg (V.A.); hristo@tu-varna.bg (H.V.)

* Correspondence: marieta.yordanova@tu-varna.bg

† Presented at the International Conference on Electronics, Engineering Physics and Earth Science (EEPES'24), Kavala, Greece, 19–21 June 2024.

Abstract: The rapid development of IoT technologies leads to their wide use in various spheres of life such as healthcare, agriculture, automotive, etc. IoT devices generate a large amount of data, which, if accessed without authorization, can lead to problems for both organizations and individuals. Therefore, security in IoT networks is a critical aspect. Due to the limited computing resources and memory of IoT devices, cryptographic algorithms are often used for their efficiency. One of the widely used cryptographic algorithms in IoT is the symmetric AES algorithm. It is the main encryption algorithm in ZigBee IoT networks. This paper presents the realization of a simulation environment that enables the investigation of the secure connection process of end nodes in a ZigBee wireless IoT network. The developed environment makes it possible to simulate the process of the secure connection of an end device in a ZigBee network. The simulation makes it possible to trace the process of obtaining an encrypted network key from the trust center based on ZigBee specifications. It is also possible to simulate the process of denying access of an end device to the network. The simulation environment implements network key encryption using the install code or a well-known key defined by the ZigBee Alliance. The encryption functionality in the simulator is realized through the implementation of the AES cryptographic algorithm.

Keywords: Internet of Things; network security; cryptography; ZigBee; AES; simulation

check for updates

Citation: Yordanova, M.; Haka, A.; Aleksieva, V.; Valchanov, H. A Simulation Tool for Security in ZigBee-Based IoT Networks. *Eng. Proc.* **2024**, *70*, 21. https://doi.org/10.3390/engproc2024070021

Academic Editor: Grigor Mihaylov

Published: 1 August 2024

Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

Technology known as the Internet of Things (IoT) has brought about remarkable advancements in connectivity, enabling the integration of various objects within a unified network. This network encompasses software applications and intelligent systems, allowing seamless interaction, extensive data collection, and efficient data processing. Consequently, a wide array of services can be provided to end users. In recent years, IoT has had a profound impact on the field of communication, empowering companies to automate processes and reduce labor costs. Undoubtedly, IoT stands as one of the most significant technologies for both business and everyday life, with the potential to further gain traction in its usage. From small sensors and actuators to cutting-edge healthcare, smart home, and smart industry devices, the applications of IoT are diverse and encompass various sectors [1].

Despite the numerous advantages of IoT, the huge amount of data generated by devices can have negative implications on data security and privacy due to limitations in memory and computing resources [2,3]. This makes devices vulnerable to attacks such as Botnets, Man-in-the-Middle, and Distributed Denial of Service (DDoS). Ensuring security during IoT communication sessions is critical [4]. Over the years, several cryptographic solutions have been developed to protect the data generated by IoT devices and networks.

Cryptography, as a technology, encompasses various methods such as symmetric and asymmetric encryption and hashing, making it a comprehensive solution. Its primary objective is to ensure the security of data within IoT networks. By employing cryptographic

techniques, it strengthens the protection of data generated by IoT devices, as well as the management of these devices, communication channels, and identities [5].

Safeguarding data privacy in IoT settings is crucial, as the exposure of sensitive data can result in challenges for both organizations and individuals. Several research studies have effectively resolved these issues by implementing cryptographic techniques [6].

The suggestion put forth in [7] involves development and deployment of an IoT cryptographic system. The system uses the AES algorithm in the Electronic Code Book, Cipher Block Chaining and Counter modes of operation as well as additional security schemes such as AES Galois/Counter Mode and AES Galois Message Authentication Code to encrypt and de-crypt text, images and electronic data applications. The research in [8] introduces adaptive eXtensible Access Control Markup Language (XACML) access policies, specifically designed for heterogeneous distributed IoT environments. The main objective of these policies is to augment the capabilities of traditional XACML by integrating access code generation and verification schemes. To ensure the security of the access code, it is encrypted using both MD5 and SHA-1 algorithms. The research paper in [9] presents an innovative algorithm that leverages artificial intelligence to improve the encryption process of autonomous IoT systems. The algorithm's primary objective is to minimize the computational complexity associated with data encryption at both the endpoints and intermediate nodes of IoT systems. To achieve this, the algorithm employs the MD5 hashing algorithm to encrypt the data collected at the IoT edge. The research paper in [10] implemented Lamport Merkle Digital Signature Verification to safeguard sensitive patient data from unauthorized access. Additionally, the authentication of IoT devices is achieved by creating trees with hash values using the Lamport Merkle Digital Signature. The aim of this strategy is to enhance the security and privacy of IoT devices utilized in the healthcare sector. The research paper in [11] addresses the challenges surrounding IoT device identification and authentication by proposing secure communication and authentication for IoT applications using the blockchain (SCAB-IoTA) method. By leveraging the power of blockchain and a hybrid cryptosystem, which encompasses AES, the Elliptic Curve Digital Signature Algorithm (ECDSA), and the SHA hash function, this method provides a comprehensive security solution. AES is utilized to ensure the confidentiality of communication between two devices, while ECDSA establishes secure communication between the device and the cluster head. The goal of this approach is to fortify IoT security measures, offering protection against potential attacks and optimizing computing and storage costs. The authors in [12] propose a low-cost fault-attack-resilient AES for IoT applications (LC-FRAES) architecture that offers a cost-effective solution for fault detection in IoT security applications. By integrating AES encryption and decryption processes, it provides a comprehensive approach to securing data in real time. The architecture's 32-bit design allows for efficient processing, while its fault-resilient nature ensures that the system can detect and handle faults effectively. The study in [13] presents a privacy-preserving scheme that offers a solution to protect the privacy of data generated by IoMT devices. The scheme proposes the use of homomorphic encryption to secure the data sent from the DU to the base station. By employing this encryption technique, the scheme aims to prevent unauthorized access and ensure that untrusted cloud servers cannot gain knowledge about the specific tasks performed on the patient's encrypted data. The primary goal is to enhance the privacy and security of sensitive medical information, thereby instilling confidence in the patients and healthcare providers regarding the protection of their data. In [14], through the implementation of this authenticated key agreement scheme, the researchers aim to address the growing concerns surrounding the security of IoT devices and networks. By leveraging ECC, a robust cryptographic technique, the proposed solution offers a high level of protection against unauthorized access and data breaches. This approach is crucial in maintaining the trust and reliability of IoT systems in the face of evolving cyber threats.

The objective of the study in [15] is to ensure that users can search for specific information within the blockchain without compromising the security and integrity of the data. Through the integration of deep learning algorithms and homomorphic encryption,

the system offers a robust solution for protecting sensitive information while enabling efficient search capabilities. In [16], by incorporating a blend of different cryptographic algorithms, the proposed HAC framework aims to enhance the security and reliability of communication channels within IoT networks. The utilization of AES, RSA, and ECC algorithms in a hybrid manner ensures a robust and versatile approach to encryption, catering to the diverse requirements of IoT device communication. The study in [17] focuses on enhancing the security of communication between IoT and blockchain networks through the introduction of an innovative algorithm. By incorporating a secure secret code into the original message generated by IoT devices, the algorithm aims to ensure the confidentiality and integrity of the transmitted data. The use of ECC for generating the hash of the secret code and the message adds an extra layer of protection against unauthorized access. Moreover, AES is employed to secure all the information using a single secret key. These keys are generated in advance and distributed to the target devices, enabling them to decrypt the encrypted data. Additionally, the algorithm utilizes specific node characteristics to identify and authenticate the legitimate nodes involved in the communication process, further enhancing the overall security of the system. The study in [18] uses an algorithm (NTM-LEACH-RSA) to address security issues in the wireless sensor network (WSN). The main goal of this innovative approach is to optimize the energy usage while prolonging the network's operational duration. By leveraging the robust RSA cryptographic algorithm, the communication and data integrity within the WSN are safeguarded effectively. The study in [19] presents the development of Reconfigurable ASCON hash functions for IoT applications (RECO-ASCON), a reconfigurable cryptographic processor that offers a lightweight solution for handling multiple hashing modes. By incorporating the Hash and Hasha ASCON algorithms, this processor ensures the secure transmission of IoT messages. The versatility of RECO-ASCON allows it to adapt to different hashing requirements, providing a robust cryptographic solution for the ever-evolving IoT landscape. The primary objective of [20] is to introduce a robust model for securing user data and upholding patient privacy within the realm of IoMT. To achieve this, the model employs an encrypted data search model and a hybrid cryptographic algorithm that merges the principles of the Caesar cipher, Elliptic Curve Diffie–Hellman (ECDH), and Digital Signature Algorithm (DSA). This algorithmic combination enables the protection of messages during their transmission and facilitates the secure exchange of keys between clients and healthcare centers. In [21], enhancing the security and trustworthiness of software updates is the main objective of the proposed blockchain-based distributed software update framework. By utilizing smart contracts and the ECDSA digital signature algorithm, the framework ensures the authenticity of the software creator's identity. Moreover, the integrity of the software update is safeguarded through verification using the SHA-3 cryptographic hash. This comprehensive approach aims to provide end-to-end security and verification throughout the software delivery process. The research in [22] introduces a novel design for a 128-bit Blowfish architecture, which utilizes a parallel pipelined memory and a field-programmable gate array radio system. The primary objective of this study is to enhance the security of IoT-connected mobile devices. In [23], to enhance patient care and enable remote medical consultations, the MedIoTverse framework is presented. This framework encompasses vital parameters like heart rate, blood pressure, and body temperature to effectively monitor patients. The key focus of this proposed solution is to ensure the utmost security during the transmission of patient data to the health server. To achieve this, the AES-256 cryptographic algorithm is employed, providing a robust and reliable encryption method. The study in [24] aims to address the vulnerabilities associated with IoT client node authentication by incorporating timestamp and signature mask in the Tempered ECDSA algorithm. The utilization of a timestamp that is not shared and has a short lifespan adds an extra layer of security to the authentication process. Furthermore, the innovative approach to generating signatures ensures that any attempts to create fake signatures are unsuccessful.

According to the research presented in Figure 1, there is a preponderance of the use of AES algorithm in IoT networks. With key lengths of 128, 192, and 256 bits, this encryption

algorithm is widely used due to its effectiveness in making decryption a complex process. Furthermore, AES is standardized by the National Institute of Standards and Technology in the USA and is recognized as a secure and reliable cryptographic algorithm, which is essential for the security of IoT networks [25].

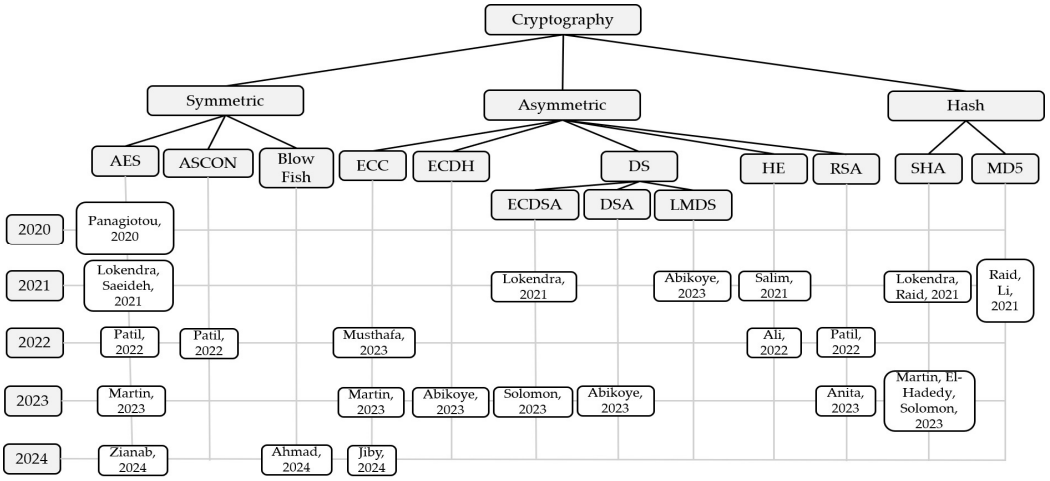


Figure 1. IoT security solutions using cryptographic methods [7–24].

The present study considers the security of the IoT technology ZigBee for wireless communications. Based on the 802.15.4 standard, the security of this technology is ensured through the utilization of the AES cryptographic algorithm. For the purpose of this paper, the process of securely connecting end devices in a ZigBee network is considered using a simulation environment.

2. ZigBee Network Standard Security

The security of ZigBee is one of its most important features. A basic security element of IEEE 802.15.4 is used in its implementation. In accordance with the ZigBee specification [26], standard security is a possible protection mode. There are various policies available in this mode for controlling the behavior or interaction of devices on the network. There were residential security and high-security modes available in earlier versions of ZigBee, but these modes have now been deprecated.

ZigBee has developed two different security models, a centralized security model and a distributed security model, to effectively protect against potential threats. In a ZigBee network with a centralized security model, the coordinator acts as the "trust center" and is responsible for setting up a centralized network by carrying out configurations and authentications once devices have joined the network. The implementation of a ZigBee network with a distributed security model involves the utilization of ZigBee routers and end devices. In this particular setup, any router possesses the ability to authorize and authenticate new devices that intend to join the network. The advantage of these networks lies in their simplified method of adding devices to the network, although this comes at the expense of a less secure network. The ZigBee router automatically generates a network key, which must be preconfigured with a link key on all ZigBee routers and end devices. This link key is utilized to encrypt the network key when transferring it from a router parent to a newly joined node. Network key distribution is the primary responsibility of this distributed trust center role, whereas trust center link key distribution is not applicable since there is no singular trust center in the network. The selection between a distributed trust center network or a trust center network is made during the network's creation. Once the network is initiated, this decision remains unchanged.

In the ZigBee network, a device takes on the important role of "trust center". This device is trusted by all other nodes and plays a vital role in distributing keys for communication purposes. It enables communication at the network layer between all nodes and at the

application layer (APL) between specific devices. Within any centralized security network, it is essential to designate a solitary device to serve as the "trust center". Moreover, all nodes within the network must exclusively recognize one active "trust center". The device serving as the "trust center" is accountable for establishing, managing, and enhancing the security measures of the ZigBee network.

In ZigBee, the level of security is dependent on the storage of symmetric keys, the security mechanisms utilized, and the correct implementation of cryptographic mechanisms, while following the related security policies. The trustworthiness of the security architecture is reflected in the secure initialization and installation of key material, as well as the secure processing and storage of such material. The correct execution of security protocols, without any steps being overlooked, is presumed. Additionally, it is assumed that the random number generators function as intended and that the secret keys remain inaccessible to unauthorized individuals, ensuring their secure storage within the device.

In a ZigBee network, security is established through the implementation of keys—"link" keys and a "network" key [27]. The communication between two APL devices is protected by a 128-bit link key that is exchanged between the devices, whereas broadcast and network layer communication is secured by a 128-bit network key that is distributed among all devices in the network. Link keys can be acquired by a device through key-transport or pre-installation, like during factory installation. Additionally, a network key can be obtained from a device through key-transport. Various application profiles have additionally created out-of-band mechanisms or key negotiation protocols for generating link keys or network keys on devices. Ultimately, the security between devices relies on the secure initialization and installation of these keys. A single variant of the network key is available, which can be employed in either distributed or centralized security models. The distribution of the network key is determined by the security model, but it does not impact the manner in which messages are secured. The utilization of either global or unique trust center link keys distinguishes the two types available. The local device's trust center link key type plays a crucial role in determining its handling of different trust center messages, such as application support sublayer (APS) commands, and whether APS encryption should be applied. The utilization of a trust center link key can also serve the purpose of securing APS data messages exchanged between the trust center and the corresponding peer device. The decision to employ APS security for these specific APS data messages lies with the higher-layer application. An application link key serves as the connection key between two devices that are not in the trust center.

Under a centralized security structure, the trust center enforces regulations regarding the inclusion of devices and the security of the network. It might be mandatory for devices to be authenticated before receiving the network key update for inclusion, or it might be mandatory for a preconfigured link key to be installed out of band. Within applications that are capable of tolerating a brief moment of vulnerability, the network key can be sent via APS-secured key transport using a well-known link key.

To facilitate trust management, a device will solely acknowledge a trust center link key or an active network key that originates from its trust center via key transport. In the context of network management within a centralized security network, the device will only accept an initial active network key and updated network keys from its trust center when it is secured with its trust center link key. In the process of configuration, a device will accept link keys meant for establishing end-to-end security between two devices only from its trust center or through application-level negotiation using a higher-level protocol between the two devices. Additional link and network keys are exclusively accepted if they are sourced from a device's trust center through secure key transport or negotiated using higher-level application protocols, apart from the initial trust center link key or network key.

Network security employs a network-wide key for the purpose of encryption and decryption. Every authorized device within the network possesses a duplicate of this key, which is utilized to encode and decode all communications within the network. An associated sequence number is assigned to the network key to differentiate a particular instance of the key. Upon updating the network key, the sequence number is incremented to

enable devices to identify the precise occurrence of the network key that has been utilized to protect the packet data. The sequence number is between 0 and 255. Once the sequence number hits 255, it loops back to 0. Each device belonging to a protected ZigBee network possesses a replica of the network key. It is essential for devices that hold the network key to safeguard it in a secure manner. Any breach of the network key could potentially compromise the confidentiality, integrity, and availability of the entire network.

In ZigBee, the standard security protocol encompasses the use of different keys to secure data in different ways. These keys are symmetric and have a length of 128 bits and can be selectively employed for encrypting or decrypting packets. The selection of whether the trust center will utilize a well-known key, an installation code, or a dynamic link key (DLK) is contingent upon striking a balance between ease of use and security. The principal application functioning on the network will influence the choice between emphasizing ease of use or improving security. The ZigBee standard supports both approaches.

- Employing a well-known key facilitates the effortless joining of devices with minimal user involvement. However, encrypting the network key with a well-known key creates a momentary security risk until the well-known key is replaced with a new one.
- The use of an installation code provides security during the initial exchange of the network key with the device, but it necessitates additional interaction between the user and the trust center. The user is required to find a means to transfer the key from the device to the trust center. This is done through a mechanism that is not part of the ZigBee network, like inputting the code into the trust center GUI from a label that presents the code on the device being added.
- The utilization of a DLK ensures the provision of the most secure form of joining, as it allows the trust center to maintain possession of the network key for an extended period before disclosing it to the joining device. Through the implementation of cryptographic methods, unique link keys are securely negotiated even when transmitted through an insecure channel.

In order to join a ZigBee standard security network, a device begins by utilizing MAC association to connect with a compatible parent device. Once the association is established, the device becomes a member of the network without authentication, as it does not possess the required network key. Before joining, the trust center could delay sending the network key and evaluate the device by sending random data requests to gather more information about its capabilities and characteristics before making a decision on authorization. Upon dispatching the success response to the MAC association request, the router communicates with the trust center by sending an update device message, indicating the intention of a new node to join the ZigBee network. Subsequently, the trust center is responsible for deciding whether to permit the device to join. When the device is not permitted to join, a request to remove the device is initiated from the parent (Figure 2). However, if the device is permitted to join, then the behavior of the trust center will vary based on whether the device possesses a preconfigured link key.

The policy set by the trust center governs the procedure for handling new devices and determines if a device should be equipped with a preconfigured link key. In the event that a new device lacks a preconfigured link key, it will be incapable of joining the network. The trust center is able to decide if the key will be the well-known default link key (ZigBee Alliance09), a pre-shared installation code, or a dynamically negotiated link key. In order for a device to connect to the network, the trust center encrypts the network key with the device's preconfigured link key (Figure 3). All ZigBee 3.0 and ZigBee application profiles necessitate the use of a preconfigured link key for joining.

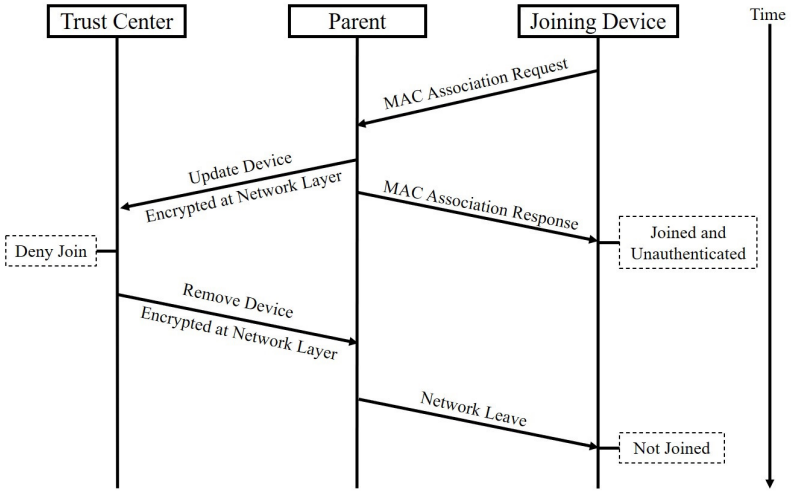


Figure 2. Denied access to join the network.

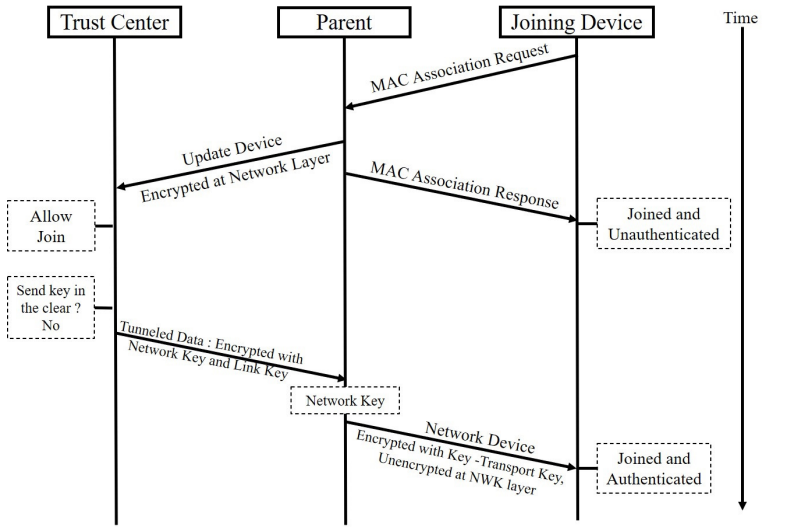


Figure 3. Joining using a preconfigured trust center link key.

3. A Simulator for Investigating the Secure Connectivity of End Devices within a ZigBee Network

In this study, an improved version of the ZigBee network simulator developed by [28] is introduced. The improvement lies in the capability of the simulator to demonstrate the secure connection process of end nodes to a ZigBee network. To enable this feature, a new function has been added to the end nodes, allowing them to decide whether they are accompanied by an installation code or not (Figure 4).

Once the end node information has been filled in, the coordinator network needs to be initialized using the new "ZigBee_Addressing" tab. After the network has been initialized, details about the coordinator and the end nodes that will connect to the network are provided (Figure 5 (1)). A unique extended unique identifier (EUI-64) is generated for each ZigBee device (Figure 5 (2)). The network ID and address of each device in the corresponding network are presented. Initially, the network ID is determined by the coordinator and only the coordinator has an address in the network. The other devices receive an address in the corresponding network after successfully establishing a secure connection with the coordinator/trust center (Figure 5 (3)). The device type is also displayed, which can be a ZigBee coordinator, a ZigBee router, or a ZigBee end device (Figure 5 (4)). If the corresponding device does not have an installation code, then a series of zeroes is shown (Figure 5 (5)). The channel on which all devices will operate is visualized (Figure 5 (6)). An indication of whether the end device is

connected to the network is displayed (Figure 5 (7)). The sequence number of the network key generated by the trust center (Figure 5 (8)), the network key itself (Figure 5 (9)), and an encrypted version of this key according to the AES algorithm (Figure 5 (10)) are presented. When selecting the device that will initiate the process of connecting to the ZigBee network and clicking on the "Join Request" button (Figure 5 (11)), a visualization of the secure connection process with the ZigBee coordinator is generated (Figure 6).

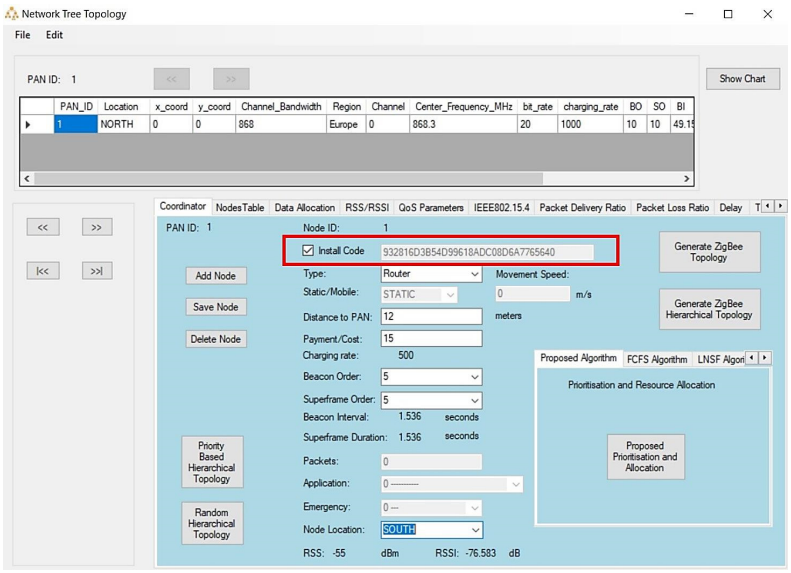


Figure 4. Installation code assignment to the end device.

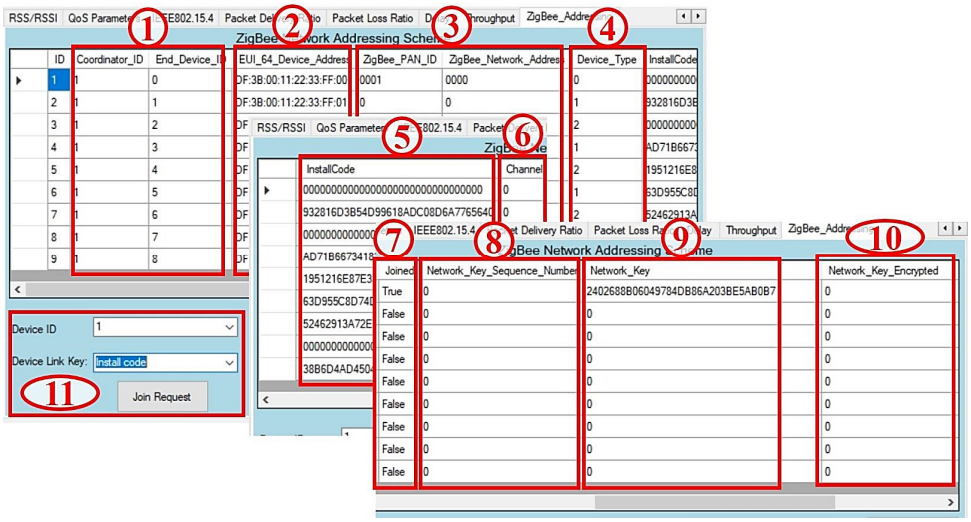


Figure 5. Network initialization.

The depiction of the procedure for linking an end device to a ZigBee network showcases communication between the end device, its parent node, and the trust center. The visualization illustrates the identification of the initiating end device for the connection process (Figure 6 (2)), along with the identification of the parent node responsible for forwarding requests to the trust center (Figure 6 (1)). Additionally, it provides information regarding the permission status of the end node to connect to the network.

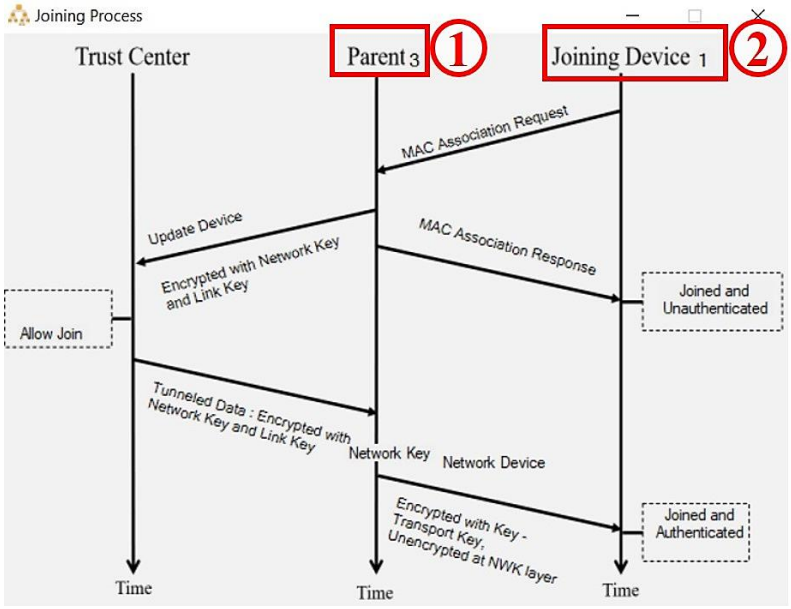


Figure 6. Depiction of the connection process.

4. Results Obtained from the Experiment

To validate the simulator's operation, an experiment was conducted involving one coordinator and eight nodes. All the nodes connected to the ZigBee network are stationary, with some having the install code and others not (Figure 7). The remaining parameters of the end nodes are identical.

Coordinator | NodesTable | Data Allocation | RSS/RSSI | QoS Parameters | IEEE802.15.4 | Packet Deliver

	Static_Mobile	Movement_Speed	Node_Location	Install_Code
	STATIC	0	SOUTH	932816D3B54D99618ADC08D6A7765640
	STATIC	0	EAST	00000000000000000000000000000000
	STATIC	0	WEST	AD71B66734182D164DB4A3104B437350
	STATIC	0	SOUTH	1951216E87E3554727E8D96A0BA7234B
	STATIC	0	SOUTHEAST	63D955C8D74D33250BC4A6CB6165A4...
	STATIC	0	SOUTHWEST	52462913A72E7064BDC210252ED8A230
	STATIC	0	EAST	00000000000000000000000000000000
▶	STATIC	0	WEST	38B6D4AD4504EB8947BC18616A76EE...

Figure 7. Endpoint configurations.

As part of the network initialization process, a unique EUI-64 address is allocated to each device for identification purposes (Figure 8 (1)). The ZigBee coordinator, which also functions as the trust center, is the sole entity within the network. It defines the ZigBee network ID (0001 in the experiment) and is identified by the ZigBee network address—0000. Additionally, it generates a random network key and assigns a sequence number to this key for tracking the current network key. This setup enables end nodes to securely connect to the ZigBee network (Figure 8 (2)).

Any device that does not receive authorization from the trust center and does not possess the correct network key will not be able to join the network. The experiment indicates that initially, no end device is connected to the network.

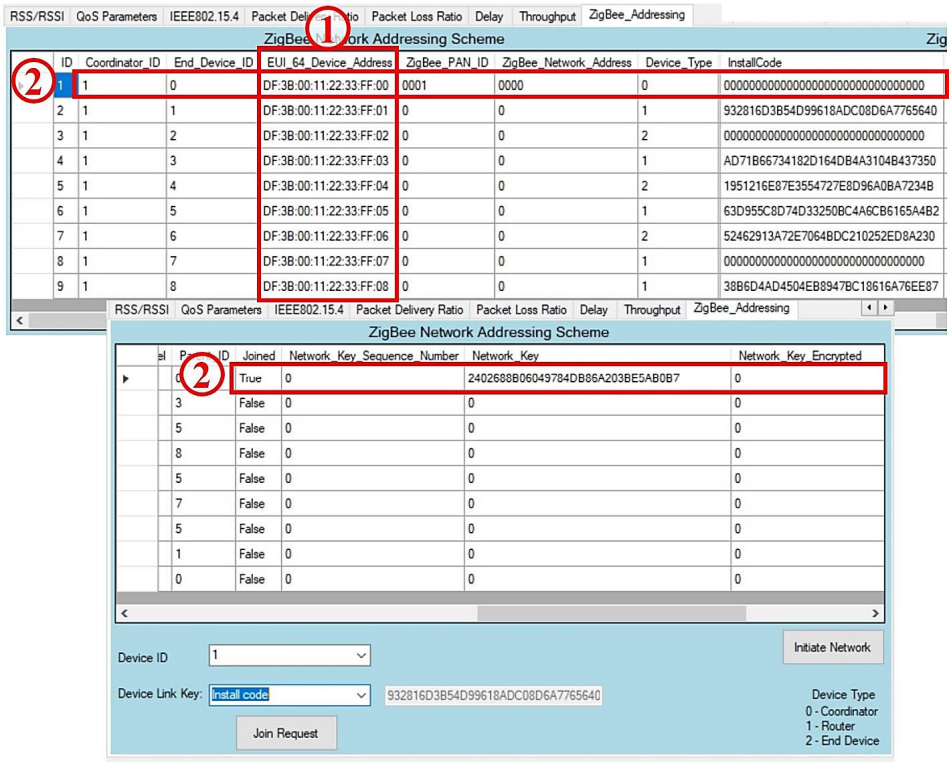


Figure 8. Network initialization waiting for incoming connection requests.

To establish a connection and simulate this operation, it is necessary to select the device's identifier that will be connected to the network and choose the encryption key type for the network key within the network. Once the "Join Request" button is selected, the ZigBee network simulates the secure connection process by sending a connection request to the trust center. To facilitate the end device's connection establishment and information transmission at the network level, it is essential to obtain a network key from the trust center. This network key is transmitted in an encrypted form. In the simulator, the trust center has the option to encrypt the network key using a well-known key or an installation code. Devices equipped with an installation code are classified as trusted since the trust center receives information about them through this code. On the other hand, devices without an installation code are labeled untrusted. The trust center encrypts the network key for devices with installation code data and sends it to the end device, facilitating a network connection. Devices lacking a reconfigured installation code will be considered untrusted by the trust center, potentially resulting in denied network access. In cases where these devices have been recognized by the trust center, the network key will be transmitted in encrypted form using a well-known key. If the trust center receives a connection request from a device without an install code, then it has the option to request additional information from the end device. If the authentication process is successful, then the device will be granted access to the network; however, if authentication fails, then the request will be rejected. The simulator enables visualization of this process through a diagram illustrating the messages exchanged between the end device and the trust center. Following the experiment conducted, when an end device bearing ID 7 (Figure 9 (1)) attempts to establish a connection with the ZigBee network, it is initially added to the network but categorized as unauthenticated by the trust center (Figure 9 (2)). Due to the absence of an install code, the trust center views this device as a potential security risk to the network and consequently refuses it access (Figure 9 (3), (4)).

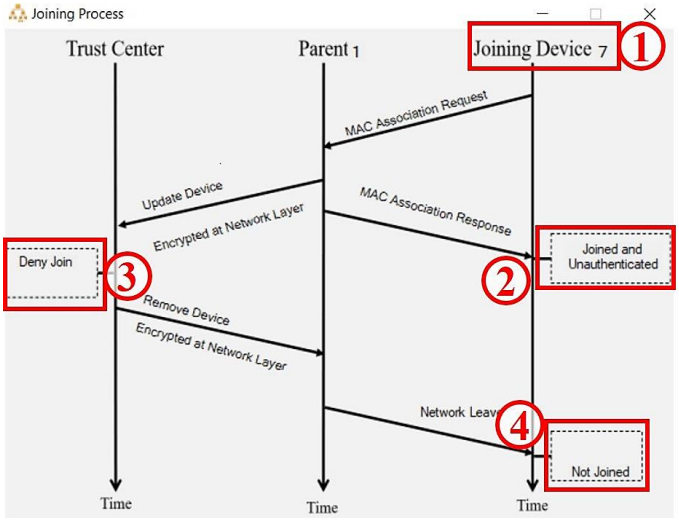


Figure 9. The process of denying access to the network.

Based on the conducted experiment, when an end device bearing ID 1 (Figure 10 (1)) attempts to join the ZigBee network, it is initially added to the network but classified as unauthenticated by the trust center until access is granted (Figure 10 (2)). Due to the presence of an install code on this device, the trust center recognizes it as a trusted device and grants it access (Figure 10 (3)). Therefore, the trust center ensures the encryption of the network key by employing the device's exclusive install code, and subsequently transmits it to the device (Figure 10 (4)). Upon receiving an encrypted network key from the trust center, the end device decrypts it using its install code. This allows the device to successfully connect to the network as a recognized device, resulting in a status change to "Joined". With the network key, the end device gains the ability to decrypt incoming network packets and securely send its own encrypted packets.

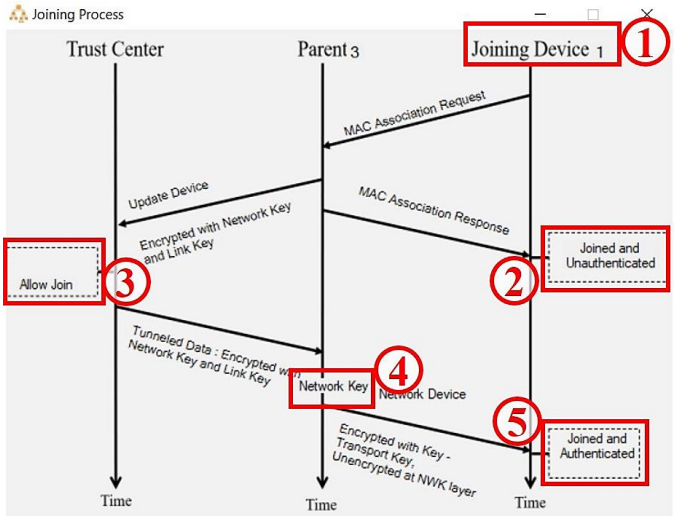


Figure 10. The process of establishing a secure connection.

Following the authentication and connection of selected devices to the coordinator, they are informed about the PAN ID (Figure 11 (1)) of the ZigBee network and a specific ZigBee network address (Figure 11 (2)).

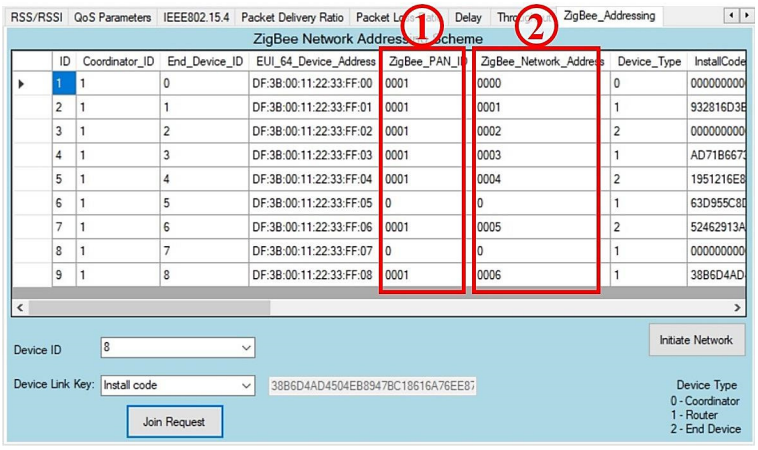


Figure 11. Authenticated devices on the network.

The simulator displays the network key and its encrypted version for devices that have established a secure connection with the trust center. It is evident from the experiment that devices with an install key configured to encrypt the network key have different encrypted versions for each device. Conversely, devices that use a well-known key for encrypting the network key have the same encrypted version across all devices (Figure 12), resulting in a security vulnerability during encryption.

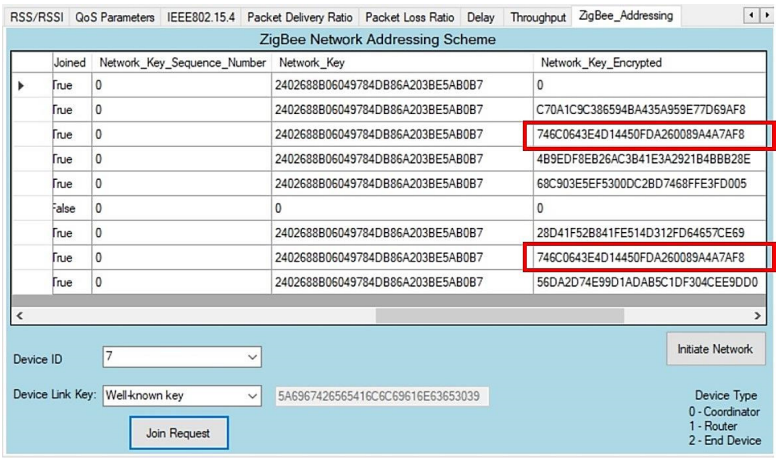


Figure 12. Receiving the network key and its encryption.

5. Conclusions

The presented paper showcases an enhanced simulation product that aims to explore the secure connection procedure between an end device and a ZigBee network. This product offers a visual representation of the utilization of various keys for implementing ZigBee protection, incorporating the AES algorithm for standard encryption. The developed simulation product makes it possible to visually present the process of controlling the access of end devices in a ZigBee network from the trust center. The process of securely establishing a connection of an end device on a network with the exchange of cryptographic keys to encrypt packets at the network layer is considered. The simulator makes it possible to examine the dynamic change of network nodes and address information according to the ZigBee standard. The product serves as an effective tool for studying the secure connection process through simulation and can also be utilized for training purposes, as it visually demonstrates this process in accordance with established standards. The experiments conducted and documented in this paper substantiate this assertion.

Author Contributions: Conceptualization, M.Y. and V.A.; methodology, M.Y., A.H. and V.A.; software, A.H.; validation, A.H. and M.Y.; investigation, M.Y., A.H., V.A. and H.V.; data curation, M.Y., A.H., V.A. and H.V.; writing—original draft preparation, M.Y.; writing—review and editing, V.A. and H.V.; visualization, M.Y. and A.H. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data are contained within the article.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Ali, S.M.; Rahu, M.A.; Karim, S.; Jatoi, G.M.; Sattar, A. Internet of Things (IoT), Applications and Challenges: A Comprehensive Review. *JIICET* **2024**, *1*, 20–27.
2. Dhar, S.; Khere, A.; Dwivedi, A.D.; Singh, R. Securing IoT devices: A novel approach using blockchain and quantum cryptography. *Internet Things* **2024**, *25*, 101019. [CrossRef]
3. Zahariev, P.Z.; Hristov, G.V.; Iliev, T.B. Study on the Impact of Node Density and Sink Location in WSN. In *Technological Developments in Networking, Education and Automation*; Elleithy, K., Sobh, T., Iskander, M., Kapila, V., Karim, M., Mahmood, A., Eds.; Springer: Dordrecht, The Netherlands, 2010; pp. 539–542. [CrossRef]
4. Stoyanov, I.S.; Iliev, T.B.; Mihaylov, G.Y.; Evstatiev, B.I.; Sokolov, S.A. Analysis of the Cybersecurity Threats in Smart Grid University of Telecommunications and Post, Sofia, Bulgaria. In Proceedings of the International Symposium for Design and Technology in Electronic Packaging (SIITME), Iasi, Romania, 25–28 October 2018. [CrossRef]
5. Perazzo, P.; Righetti, F.; Manna, L.M. Performance evaluation of Attribute-Based Encryption on constrained IoT devices. *Comput. Commun.* **2021**, *170*, 151–163. [CrossRef]
6. Muneeswari, G.; Varun, S.S.; Hegde, R.; Priya, S.S.; Shermila, P.J.; Prasanth, A. Self-diagnosis platform via IOT-based privacy preserving medical data. *Meas. Sens.* **2023**, *24*, 100636. [CrossRef]
7. Panagiotou, P.; Sklavos, N.; Darra, E.; Zaharakis, I. Cryptographic system for data applications, in the context of internet of things. *MICPRO* **2020**, *72*, 102921. [CrossRef]
8. Riad, K.; Cheng, J. Adaptive XACML access policies for heterogeneous distributed IoT environments. *Inf. Sci.* **2021**, *548*, 135–152. [CrossRef]
9. Li, B.; Feng, Y.; Yang, W.; Liu, G. Research on AI security enhanced encryption algorithm of autonomous IoT systems. *Inf. Sci.* **2021**, *575*, 379–398. [CrossRef]
10. Jafar, A.A. Blockchain-based Lamport Merkle digital signature: Authentication tool in IoT healthcare. *Comput. Commun.* **2021**, *170*, 200–208. [CrossRef]
11. Lokendra, V.; Debasis, D. SCAB-IoTA: Secure communication and authentication for IoT applications using blockchain. *J. Parallel Distrib. Comput.* **2021**, *154*, 94–105. [CrossRef]
12. Saeideh, S.; Seok-Bum, K.; Ali, M. A low-cost fault-attack resilient AES for IoT applications. *Microelectron. Reliab.* **2021**, *123*, 114202. [CrossRef]
13. Salim, M.M.; Kim, I.; Doniyor, U.; Lee, C.; Park, J.H. Homomorphic Encryption Based Privacy-Preservation for IoMT. *Appl. Sci.* **2021**, *11*, 8757. [CrossRef]
14. Musthafa, A.S.; Preya, A.J.; Alneyadi, F.M.; Alattas, N.S.; Hassan, G.E.; Zia, H. Safeguarding IoT Device Deployment in Healthcare: Analysis and Strategies for Enhanced Security and Privacy. In Proceedings of the 2023 4th International Conference on Communications, Information, Electronic and Energy Systems (CIEES), Plovdiv, Bulgaria, 23–25 November 2023; pp. 1–7. [CrossRef]
15. Ali, A.; Pasha, M.F.; Ali, J.; Fang, O.H.; Masud, M.; Jurcut, A.D.; Alzain, M.A. Deep Learning Based Homomorphic Secure Search-Able Encryption for Keyword Search in Blockchain Healthcare System: A Novel Approach to Cryptography. *Sensors* **2022**, *22*, 528. [CrossRef]
16. Patil, K.S.; Mandal, D.I.; Rangaswamy, D.C. Hybrid and Adaptive Cryptographic-based secure authentication approach in IoT based applications using hybrid encryption. *Pervasive Mob. Comput.* **2022**, *82*, 101552. [CrossRef]
17. Martin, P.; Part, S. Internet of things-blockchain integration: A robust data security approach for end-to-end communication. *Indones. J. Electr. Eng. Comput.* **2023**, *32.2*, 1050–1057. [CrossRef]
18. Anita, S.; Saravanan, S.; Chandrasekar, A. Trust management based multidimensional secure cluster with RSA cryptography algorithm in WSN for secure data transmission. *Meas. Sens.* **2023**, *29*, 100889. [CrossRef]
19. El-Hadedy, M.; Guo, X.; Yoshii, K.; Cai, Y.; Herndon, R.; Banta, B.; Hwu, W. RECO-ASCON: Reconfigurable ASCON hash functions for IoT applications. Integration. *Integration* **2023**, *93*, 102061. [CrossRef]

20. Abikoye, O.C.; Oladipupo, E.T.; Imoize, A.L.; Awotunde, J.B.; Lee, C.-C.; Li, C.-T. Securing Critical User Information over the Internet of Medical Things Platforms Using a Hybrid Cryptography Scheme. *Future Internet* **2023**, *15*, 99. [CrossRef]
21. Solomon, G.; Zhang, P.; Brooks, R.; Liu, Y. A Secure and Cost-Efficient Blockchain Facilitated IoT Software Update Framework. *IEEE Access* **2023**, *11*, 44879–44894. [CrossRef]
22. Ahmad, R.; Rajendran, J.; Ismail, W. Parallel-pipelined-memory Blowfish FPGA-based radio system with improved power-throughput for secured IoT network. *Ain Shams Eng. J.* **2024**, *15*, 102625. [CrossRef]
23. Zianab, K.M.; Mazin, A.M.; Karrar, H.A.; Dilovan, A.Z.; Abdullah, L.; Haydar, A.M.; Jan, N.; Radek, M. A Metaverse Framework for IoT-Based Remote Patient Monitoring and Virtual Consultations Using AES-256 Encryption. *Appl. Soft Comput.* **2024**, *158*, 111588. [CrossRef]
24. Jiby, J.P.; Shelbi, S.; Bharat, B. Temporal ECDSA: A timestamp and signature mask enabled ECDSA algorithm for IoT client node authentication. *Comput. Commun.* **2024**, *216*, 307–323. [CrossRef]
25. Eyad, M.H.; Samer, A.; Amer, A.; Omar, A.S. Implementing and developing multi-stage cryptography technique for low-cost long-range communication system. *Bull. Electr. Eng. Inform.* **2024**, *13*, 264–276. [CrossRef]
26. Connectivity Standards Alliance. Zigbee Specification, Revision 23. 15 March 2023. Available online: https://csa-iot.org/all-solutions/zigbee/ (accessed on 14 April 2024).
27. Silicon Labs. AN1233: Zigbee Security, Rev. 0.4. Available online: https://docs.silabs.com/zigbee/7.4.2/zigbee-security-overview/ (accessed on 14 April 2024).
28. Haka, A.; Dinev, D.; Aleksieva, V.; Valchanov, H. A study of ZigBee Networks in Experimental Environment and Simulation. In Proceedings of the International Conference on Communications, Information, Electronic and Energy Systems (CIEES 2022), Veliko Tarnovo, Bulgaria, 24–26 November 2022. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.