İSTANBUL UNIVERSITY CERRAHPAŞA

ORIGINAL ARTICLE
Electrica 2024; 24(3): 589-600

Implementing Enhanced Security in the Zigbee Joining Process Using Elliptic Curve Diffie–Hellman Algorithm

Ngonidzashe Gwata, Vipin Balyan

Department of Electronic Engineering, Cape Peninsular University of Technology, Cape Town, South Africa

Cite this article as: V. Balyan and N. Gwata, "Implementing enhanced security in the Zigbee joining process using elliptic curve Diffie–Hellman (ECDH) algorithm," *Electrica*, 24(3), 589-600, 2024.

ABSTRACT

The surge in Internet of Things (IoT) growth has generated an amplified need for secure communication protocols, underscoring the importance of identifying vulnerabilities in prominent IoT communication protocols such as Zigbee. This study thoroughly assesses papers that explore security issues, such as vulnerabilities, attacks, and countermeasures in Zigbee networks. It is then identified that the key vulnerability lies in key transportation and management within Zigbee networks. To address this issue, an enhancement is proposed, which focuses on reinforcing security during key transportation by integrating Elliptic Curve Diffie–Hellman (ECDH) encryption during the network joining process. This solution aims to securely protect the network key against unauthorized access or manipulation. The proposed method is evaluated in terms of its computational, energy, and communication overhead. The results demonstrate that the suggested approach imposes minimal additional demand, resulting in improved overall security with minimal resource requirements. This research contributes to IoT security by providing an enhanced approach to safeguarding Zigbee networks, ultimately reinforcing the safety measures for IoT devices and ensuring secure data transmission.

***Index Terms*—**IoT, Elliptic Curve Diffie–Hellman, network security, Zigbee

I. INTRODUCTION

The Internet of Things (IoT) is experiencing a soaring expansion with approximately 15.14 billion connected IoT devices as of 2023[1]; furthermore, it is predicted that this number will increase to exceed 25 billion by the year 2030. Zigbee, reported as widely used [2], requires heightened security due to increasing IoT and device demand, especially when transmitting sensitive data.

The Zigbee protocol is continually evolving, and in April 2023, the Connectivity Standards Alliance (CSA) unveiled the latest update—Zigbee PRO 2023. This new release includes advanced security measures that highlight device safety and compatibility within IoT development. Following [3], the key features of this upgrade involve improved mechanisms for safeguarding networks during onboarding and operation stages to combat current security risks.

However, the challenges persist, as the upgraded protocol continues to suffer from a wide range of security weaknesses. This is attributed to the incapacity of constrained wireless sensor network devices to employ conventional security protocols like asymmetric cryptographic mechanisms, which are resource intensive and unsuitable for wireless sensor networks [4]. Even though the latest Zigbee version does not address all the vulnerabilities, the protocol still receives little attention from the research community [5]. Through their study, [6] found certain researchers touch upon Zigbee but neglect the aspect relating to vulnerability or potential security breaches. Hence, there is a need to conduct further research in this area and increase research efforts to develop alternative security solutions.

Zigbee utilizes the Advanced Encryption Standard (AES), a highly secure symmetric encryption algorithm widely accepted and expected to remain secure beyond 2030 [7]. The vulnerability in Zigbee lies not in the cryptographic algorithm but in key management, involving the generation, storage, distribution, and security of keys needed for encryption and decryption processes.

In this study, the researchers propose integrating the Elliptic Curve Diffie–Hellman (ECDH) Algorithm into the Zigbee joining process. This results in a lightweight Zigbee security solution

Corresponding author:
Ngonidzashe Gwata

E-mail:
ngonidzashegwt@gmail.com

Received: December 22, 2023

Revision Requested: May 22, 2024

Last Revision Received: June 14, 2024

Accepted: June 29, 2024

Publication Date: September 9, 2024

DOI: 10.5152/electrica.2024.23188

that is suitable for low-powered and hardware-limited devices, leading to enhanced security for both IoT and Wireless Sensor Network (WSN) applications.

In summary, the research contributes to:

1. Analyze security weaknesses that may render the Zigbee communication protocol unsuitable for transmitting private and sensitive information.
2. Identifying the methods by which attackers can compromise a system through exploiting flaws in the Zigbee specification.
3. Incorporating ECDH into the Zigbee join process to secure the transmission of link keys.

The content of this paper is organized in the following manner: Section 2 presents a concise review of the studies related to the proposed scheme. Section 3 entails an overview of crucial elements required to comprehend Zigbee's security framework and network model. Section 4 reveals the flaws in the Zigbee standard. The ECDH technique along with its integration into the Zigbee standard during the device join process is elaborated in detail under Section 5. Results obtained from the authors' proposed security solution will be discussed under Section 6, followed by concluding remarks towards wrapping up this paper.

II. RELATED WORK

A. Security Features by Zigbee Alliance

The following discusses the security features introduced by the Zigbee Alliance and highlights their drawbacks.

1) Zigbee Link Keys

Link keys can be generated between devices in a Zigbee network. The network key (NWK) is used to encrypt the link keys when they are transmitted. However, the use of NWK for link key protection poses a security risk. An attacker with access to the NWK can obtain link keys. Although link keys enable unique encryption of messages between two devices, the mechanism by which these keys are exchanged does not prevent other devices, malevolent or not, from accessing them.

2) Installation Code

While the installation code is intended to facilitate secure data transmission during device joining, it is vulnerable as it is often physically exposed. Even if an attacker gains access after joining, there is no forward secrecy, allowing for decryption of recorded transmissions.

3) Touchlink

This enables nearby devices to communicate, even if they are not on the same network. However, [8] found a security risk, extracting the network key from 130 meters away during touchlink operations. This allowed unauthorized actions like factory resets, highlighting a potential security threat in Zigbee applications.

B. Public Key Infrastructure (PKI)-based suggestions

1) Certificate-based Model

The primary vulnerability in Zigbee arises from its use of symmetric cryptography. Misra et al. [9] suggested the implementation of a public key infrastructure (PKI) to provide each device with a unique manufacturer-signed certificate, along with private and public keys. However, this approach falls short in distinguishing legitimate devices from numerous other certified ones within an open ecosystem featuring multiple vendors, thus making it easier for attackers to include fake or malicious items. With many legitimately certified tools operating simultaneously, hackers can find it easy to modify software features on target tools or even establish fraudulent certifications that hide malware infections. Additionally, limitations exist concerning installing all vendor root certificates onto Zigbee devices during production—which restricts their usefulness when interacting with non-root-certificate compatible peers as well as limits their capabilities too due to resource constraint issues.

2) Certificateless Model

Allakany et al. [4] reported that Lee et al. [10] have developed a lightweight mutual authentication protocol specifically for IoT, utilizing symmetric encryption instead of complex methods like asymmetric encryption to suit the limited capacity of constrained devices. However, they argued that, while this scheme facilitates RFID tag-reader authentication, its cryptographic measures' effectiveness appears neglected in their work analysis. [4] also reported that Kulkarni et al. [11] have introduced a secure routing protocol for Zigbee networks that includes aggregated MAC as the authentication codes. Researchers in [4] further propounded that the proposed method guarantees end-to-end authentication by enforcing it at each route step, but its reliance on only two keys may not guarantee communication confidentiality. They also said that the study solely focuses on analyzing security concerns about the MAC aspect of Zigbee and overlooks other potential attack vectors or vulnerabilities within the protocol itself in detail.

To enhance authentication and security, [12] recommended utilizing ECDH for distributing NWK in the Zigbee network. Yet it is unclear how secret keys function as NWK since they must be distributed to all devices for broadcast message decryption. Furthermore, [13] discovered ECDH impractical for NWK generation. The coordinator (also Trust Center) refused to accept the packet containing the public key of the joining node.

III. FUNDAMENTALS OF ZIGBEE

Zigbee is a wireless communication standard designed for low-power, short-range applications on the Internet of Things (IoT) domain. It uses the IEEE 802.15.4 standard for physical and data link layers. Illustrated in Fig. 1, the Zigbee protocol stack comprises four layers: the Network layer (NWK), Application layer (APL), Physical layer, and medium access control (MAC) layer. The Physical and MAC layers are under the regulation of IEEE 802.15.4, while the Zigbee standard presides over the network and application layers.

A. Zigbee Security

1) Security Framework in Zigbee Protocol

The security framework of Zigbee aligns with the IEEE 802.15.4 standard. Zigbee's security services encompass the establishment of keys, their distribution, and the protection of data frames through the utilization of Advanced Encryption Standard (AES) cryptography.

2) Zigbee Security Keys

The Zigbee standard relies on three distinguished 128-bit symmetric keys that have significant roles to play.

i. The Network Key is crucial for secure broadcast communication among nodes.
ii. The Link Key is utilized in unicast.

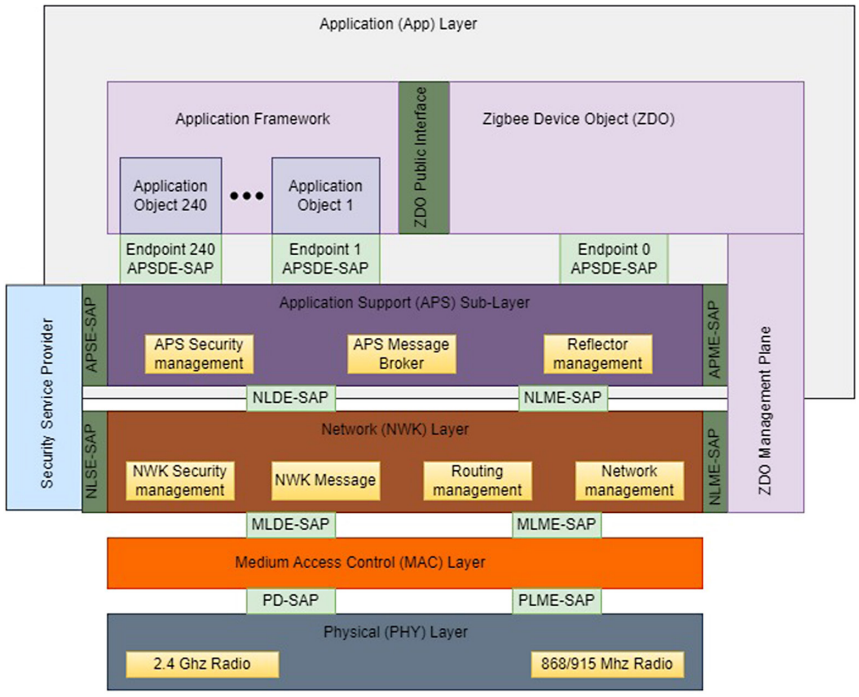

Fig. 1. Zigbee stack.

3) Key Management in Zigbee

Zigbee offers a variety of key management mechanisms:

i. **Pre-installation:** Manufacturers embed the key into the device during production.
ii. **Key Establishment:** Link keys are generated from a master key using a one-way function rooted in the SKKE protocol. Devices must possess the master key obtained through pre-installation, key transport, or user input to engage in this process.
iii. **Key Transport:** The network device requests a key from the Trust Center, and the key-load key secures the transportation of the master key.

An installation code is a random value loaded during production and encrypts the initial network key transport. The code may be presented in hexadecimal or encoded form, such as through barcodes or QR codes on packaging. However, it should be noted that anyone with physical access to the device packaging can obtain the install code itself—which would subsequently grant them the ability to decrypt any transported messages associated with the network key (NWK). As there is no guarantee of forward secrecy utilizing this method either; even if an attacker is not present at join-time—they can still record encrypted transmissions until obtaining the installation code.

Alternatively, the coordinator can generate and distribute link keys encrypted using the NWK. Because the link key is protected by the NWK, an attacker with the NWK can access the link key and undertake attacks. Although link keys enable unique encryption of messages between two devices, the mechanism by which these keys are exchanged does not prevent other devices, malevolent or not, from accessing them. Sending link keys encrypted with the NWK defeats the purpose.

These flaws related to key management highlight the necessity of reassessing how devices in the network share and utilize secret keys. A potential solution is implementing ECDH integration during device joining, which can establish a secure exchange mechanism via a shared link key between any pair of communicating devices without relying on NWK confidentiality.

4) Joining Procedure of Node

To join a WPAN, a device scans channels to find potential coordinators, sends an association request upon selection, and awaits confirmation. A data request is then used to check acceptance, and the coordinator responds, confirming or denying the association. Refer to Fig. 2 for detailed steps on device pairing in this network system.

IV. ZIGBEE SECURITY VULNERABILITIES

The following are some of the weaknesses in the Zigbee standard.

A. Replay and Injection Attacks

Attackers can capture plaintext MAC and network addresses, and Personal Area Network ID (PAN ID) from Zigbee traffic, enabling them to impersonate network devices and disrupt communication. The steps that an attacker could take in the normal operation of Zigbee networks are detailed in [14]. Manipulation is achieved by retransmitting the packets, making them appear as if they originated from a device in a network. Consequently, the network may interpret the malicious traffic as legitimate. In a study by [15], a replay attack on a standard Zigbee network was executed successfully. This attack utilized a program called killerBee, which allowed the authors to inject previously captured frames into the Zigbee network.

B. Weakness in NWK Transmission

Some Zigbee devices transmit network keys (NWK) in plain text or encrypted with GMK during initial communication. This has been identified as Zigbee's main vulnerability in earlier studies by [16] and [17]. [16] conducted a successful packet sniffing experiment to

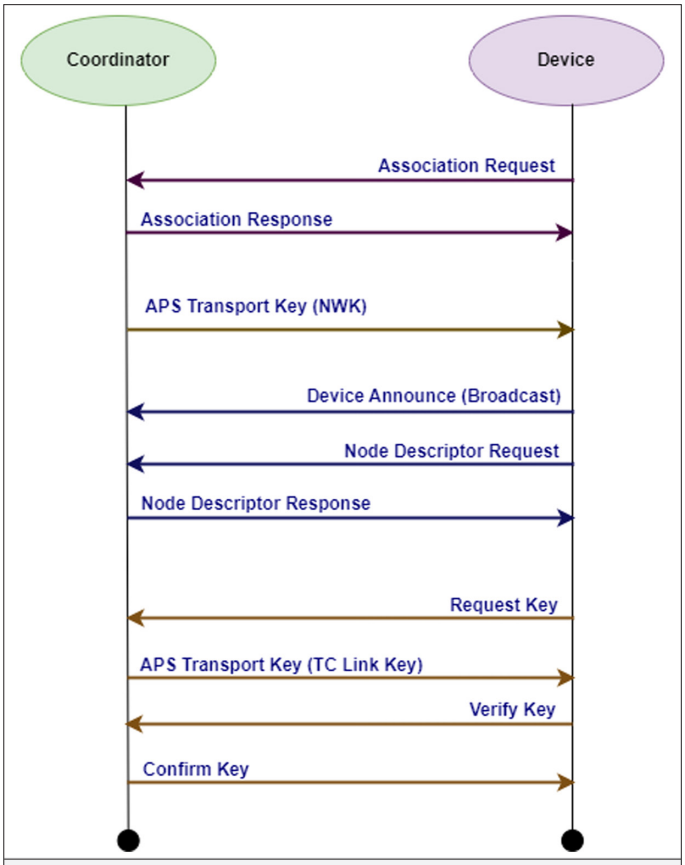

Fig. 2. Joining procedure.

obtain the NWK key when a device joined the network. This NWK key enabled hackers to control the network, add harmful devices, and replicate other devices' parameters, enabling their malicious device to join by deceiving the Trust Center into considering it a legitimate reconnection.

C. Forward Security Flaw

The Zigbee security model fails to properly address the need for secure revocation. As a result, even after a node leaves the network, it can still access the connectivity because it retains the master and link-keying material. Extracting security keys is possible, making data extraction a real threat [18].

D. Denial of Service (DoS)

In [19]'s study, they demonstrated a Denial of Service (DoS) attack on Zigbee networks. By injecting malicious packets without monitoring network traffic, they overwhelmed the network with dummy messages. A 2-minute attack led to network freezing, and a 10-minute attack caused a network crash. These experiments underscore the real-world dangers of DoS attacks.

E. Malware Distribution Through Smart Lamp

Researchers demonstrated an attack using a compromised Zigbee-enabled Philips Hue lamp to distribute malware across a network [20].

F. Worm Infection

Researchers created a worm that can destroy Philips Hue lightbulbs throughout entire cities by exploiting hardcoded symmetric encryption keys in Zigbee devices [21].

G. Passive Sniffing

Researchers in [22] were able to determine the type and status of smart home devices even when their communication using Zigbee is encrypted; this finding could potentially lead to inadvertent disclosure of user information.

V. THEORETICAL FRAMEWORK

This research proposes a solution to improve the security of Zigbee networks by employing an Elliptic Curve Diffie–Hellman (ECDH) key exchange scheme during the join procedure. This is accomplished by switching the order of key exchange so that the link key goes before the NWK. The NWK in the transport key message is then encrypted using the link key as opposed to TCLK.

A. ECDH Key Exchange in Zigbee

ECDH Key Exchange enables two parties with elliptic-curve key pairs to establish a shared secret across an unsecured channel without transmitting the key itself.

Incorporating it into Zigbee protocol, two Zigbee devices Dev1 and Dev2, can share a secret key as follows:

- Dev1 and Dev2 have the parameters of the elliptic curve pre-installed, including the generator point G, and n, an integer associated with the curve.
- Dev1 will generate a random private key d_1 in the range [1, n−1] by taking a point on the curve and computing a public key; $Q_1 = d_1 \times G$.
- Dev2 will do the same and generate its public key Q_2 from its private key d_2: $Q_2 = d_2 \times G$.
- Dev1 and Dev2 exchange their public keys.
- Dev1 will then use Dev2's public key and its private key to calculate:

$$sharekey\_dev1 = d_1 \times Q_2 = d_1 \times d_2 \times G.$$

- Dev2 will then use Dev1's public key and its private key to determine:

$$sharekey\_dev2 = d_2 \times Q_1 = d_2 \times d_1 \times G.$$

Since d2 and d1 are integers, $d1 \times d_2 = d_2 \times d_2$, hence, for the properties of the elliptic curves, sharekey_dev1 = sharekey_dev2 = S.

As a result, dev1 and dev2 share a secret key, S, which is used as the link key for secure end-to-end communication in the network. Fig. 3 shows the updated join procedure with ECDH. The coordinator and the joining device exchange values Q_1 and Q_2 during the joining process. These values can be included in the association request and response commands, eliminating the need for any additional messages in the joining operation. After acquiring each other's values, the shared secret key S is generated, allowing the coordinator to send the NWK symmetrically in the transport key message encrypted by S. Keeping the NWK inaccessible to eavesdroppers.

An attacker who can listen in on the exchange cannot obtain the value of the key S and hence cannot extract the NWK. Since the key S is adopted as the link key in the proposed model, each ZC, ZR, or ZED must generate a unique encryption key for each connection they establish. This technique has the following advantages:

- Protection from attackers with the TCLK.
- Inability of an opponent with the NWK to compromise devices due to messages being encrypted with link keys.
- Devices being unable to eavesdrop on conversations between other devices in the network.

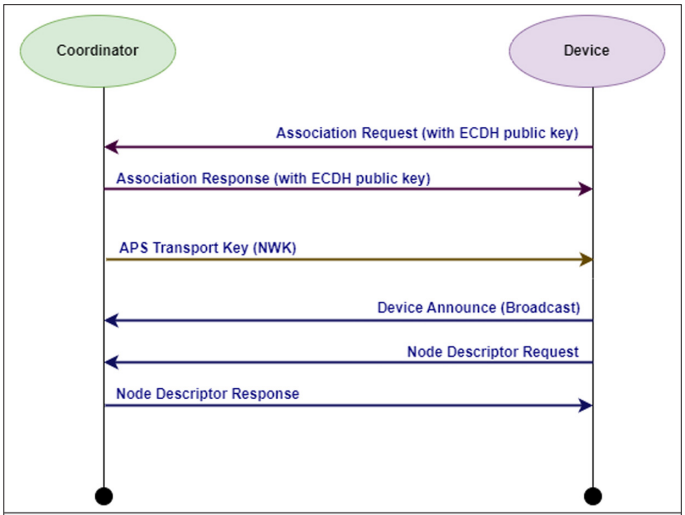

Fig. 3. Join procedure with ECDH.

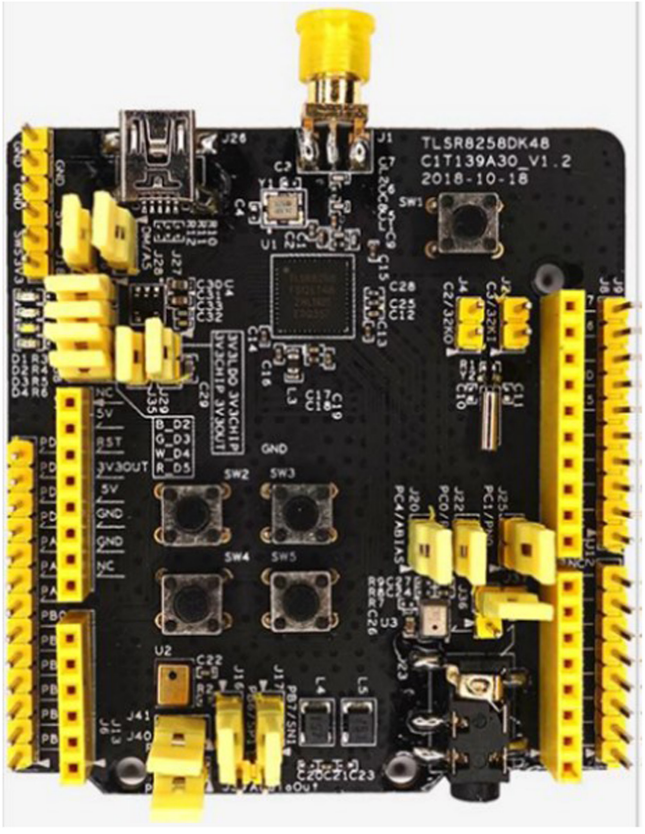

Fig. 4. B85 board.

Fig. 5. B85 dongle.

B. Implementation

This research uses the B85 Development Board and B85 Dongle, development kits based on Telink Semiconductor's TLSR8258 SoC, for the implementation of Zigbee, Figs. 4 and 5.

The target board's firmware was downloaded using an adapter that was built around a Telink TLSR8266-based burning EVK, Fig. 6. It uses a mini-USB interface to connect to the PC and a serial wire debug interface to communicate with the target board's microcontroller.

C. Software Tools

The experiment's and solutions' firmware were created using the Telink IDE and the Telink Zigbee SDK was installed. The development boards are programmed and debugged using Telink IDE for TC32, an Eclipse-based Integrated Development Environment (IDE).

Programming of the boards was done using the Telink Burning and Debugging Tool (BDT).

Performing complex mathematical operations such as elliptic curve point multiplications involved in ECDH requires intense computation. This becomes challenging for devices that are resource-constrained and operate on low-power microcontrollers. However, this challenge has been addressed by making use of an

Fig. 6. B85 board.

optimized ECDH library specifically designed for 8-bit, 16-bit or 32-bit microcontrollers.

D. Experiments

The code was written to work per the proposed solution.

A B85 EVK board is configured as ZC, one B85 USB dongle is configured as ZR, and another as ZED. Secp128r1, a 128-bit prime field Weierstrass curve, was used for the ECDH protocol's public-key algorithm solution. The ECDH C library used is a small and fast ECDH and ECDSA implementation called the micro-ecc.

In this proposed approach, ECDH is used to generate a secret that is used as a link key. The generator point, G, and an integer associated with the curve, n, are stored in the ZC and ZR. The ECDH drive provided by micro-ecc generates public and private keys and computes the symmetric key from the generated public key and shared secret.

The joining process flow for the ZC and ZR joining to form a network is depicted in Figs. 7 and 8. The process begins with the ZC allowing devices to join its network. The ZR generates a private key d_r and computes a public key Q_r before sending the association request. The public key is then included in the association request payload. When the ZC receives an association request, it reads and stores Q_r, generates its private key d_c, and calculates the public key Q_c. The ZC then includes its public key as the payload when sending the association response. The ZR will read the coordinator's public key after receiving the response. At this point, both devices compute the symmetric key S using the predefined secp128r1 curve with basepoint G, as well as the public and private keys. If the procedure is successful, they will have the same symmetric key.

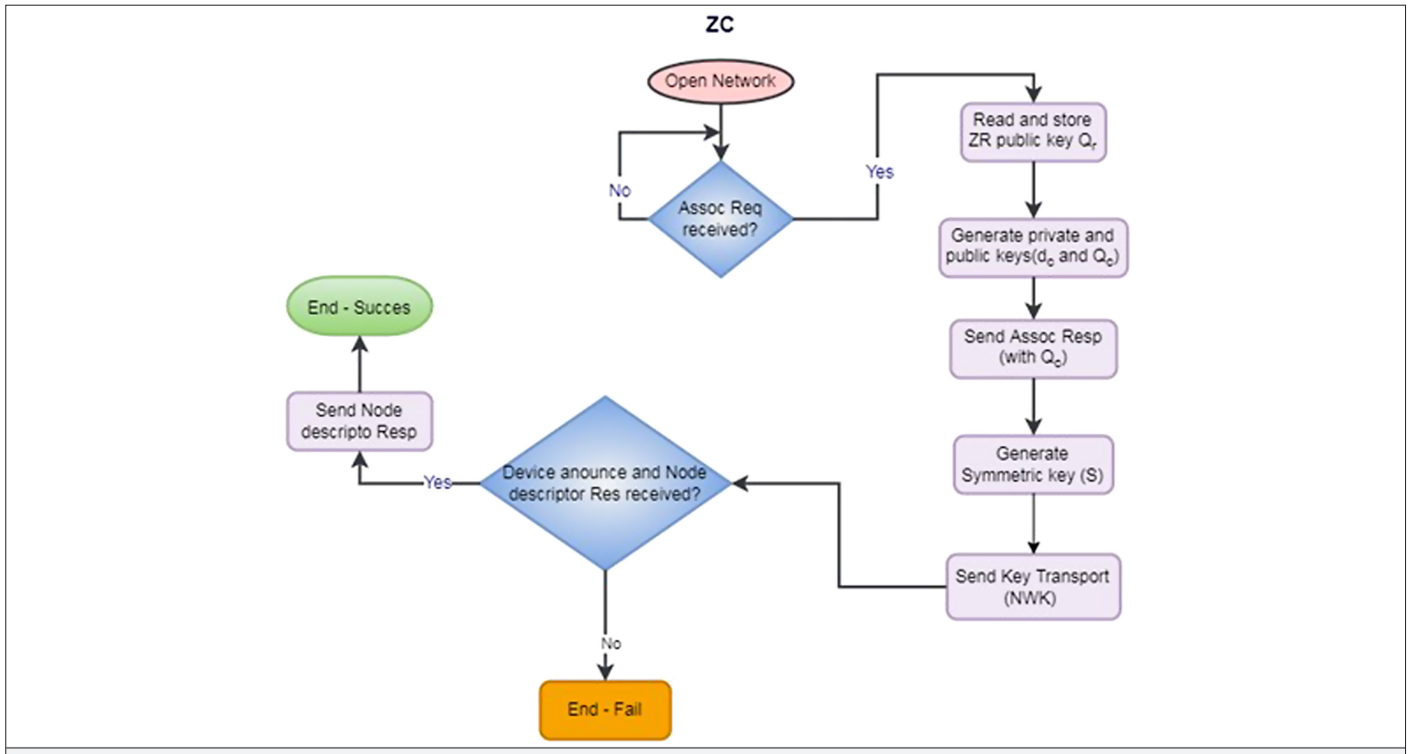

Fig. 7. Zigbee Coordinator creating network.

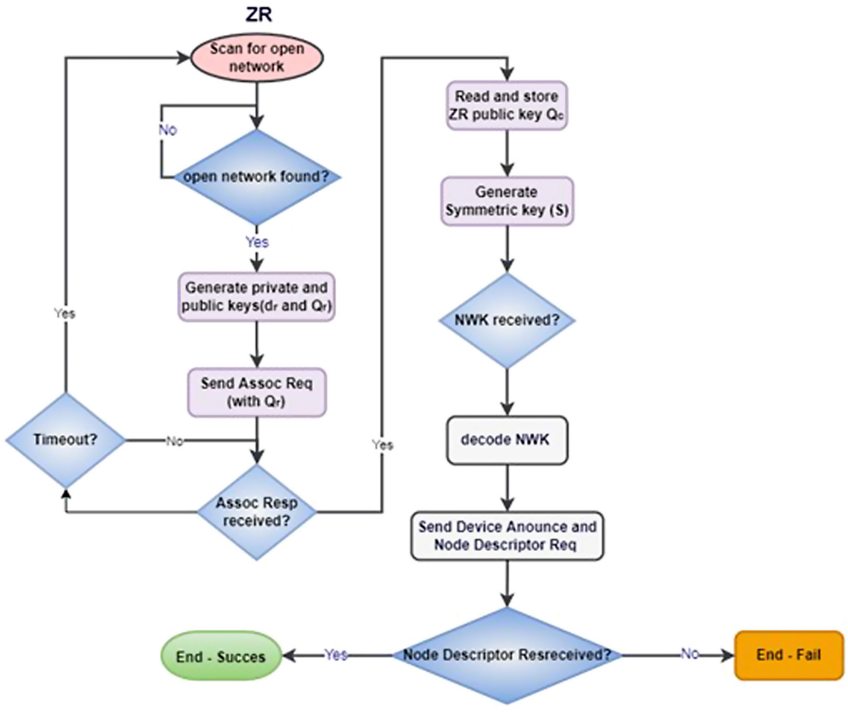

Fig. 8. Zigbee Router joining network.

The ZC coordinator then encrypts the NWK in the key transport command with the symmetric key S. Upon receiving the key transport command, because it has the same symmetric key as the coordinator, the router can decrypt the key transport payload and retrieve the NWK. The joining process continues, with the ZR broadcasting the device announce command and sending a node descriptor request. The ZC concludes the process by responding to the node descriptor request.

If the devices do not generate the same symmetric key, the router won't be able to decrypt the NWK, and other network devices will not process its device announcement broadcast and node descriptor requests. As a result, the join is unsuccessful.

E. Method for Setting up a Zigbee Network

The Zigbee network was set up as a star network topology, which provides centralized security.

The boards were configured as follows to create nodes:

- B85 Dongle 1, as a Gateway (Coordinator)
- B85 Board, as a Light (Router)
- B85 Dongle 2, as a Switch (End Device)

When the SW1 button on the gateway board is pressed, it instructs the ZC to open or close the network. When a gateway is powered on, if it is a new device, it will begin to build a network and allow devices to join. If it is a device that has previously established a network, the network will be restored, and the permit join state can be activated by pressing the button.

When the router is powered on as a new device with the gateway's network open, the network join will begin automatically. If the device was previously connected to the network, the network will be restored. When the endpoint is turned on, a similar process occurs; if it is a new device, the network join is initiated automatically.

When the end point's button (SW1) is pressed, a message is sent to the gateway. This endpoint serves as a switch. In turn, the gateway will send a toggle command to the router. In this arrangement, the router is a light node. When the router receives the toggle command, it will turn an LED on or off. This procedure is used to determine whether the proposed solution will allow Zigbee devices to function normally in the network.

VI. RESULTS AND DISCUSSION

This section covers the findings of the experiments for the suggested approach to improving Zigbee security through incorporating ECDH in the join process.

A. Computational Overhead

Only an ECDH secret key is included in both the end device's association request message and the coordinator's association response message. The computation time for ECDH point multiplication has been determined to be approximately 275.8 milliseconds.

To measure the computation time for ECDH (Elliptic Curve Diffie–Hellman) point multiplication, the following steps were performed:

- First, Timer0 of the TLSR8261 MCU was configured into mode 3 (Tick mode) to measure time in high-resolution increments. The timer registers were initialized and set to appropriate prescaler values to achieve a resolution of 1 microsecond, with the initial Tick value of Timer0 set to zero.
- Next, Timer0 was enabled immediately before the ECDH point multiplication began, with the timer tick value increasing by 1 on each positive edge of the system clock.

- To measure the computation time, the timer value was captured at the end of the point multiplication, yielding a measured value of 275,804 ticks.
- Finally, the computation time was calculated by multiplying the tick count by the timer resolution, resulting in a total of 275.8 milliseconds for the ECDH point multiplication.

It should be noted that this time presupposes that both the coordinator and the end device can compute ECDH point multiplication at the same time, avoiding any overhead duplication associated with this operation. It is essential to keep in mind, however, that these timings are dependent on specific hardware performance characteristics.

B. Memory Overhead

Extra memory space is needed for storing the constants and codes specifically related to ECDH. Fig. 9 illustrates memory sizes after compiling a sample gateway code, excluding the ECDH algorithm. In Fig. 10, the compilation includes the ECDH algorithm and associated constants. The augmented memory size reflects the additional space required for storing both the constants and code segments essential for ECDH implementation.

The highlighted "text" and "bss" sections report the sizes of distinct program segments. The "text" section refers to the code segment, containing executable instructions. The "bss" section represents the data segment, which includes uninitialized global and static variables. The introduced constants occupy 22,218 – 22,034 = 186 bytes, while the algorithm code adds another 175,372 – 169,704 = 5668 bytes of code space (Table I).

C. Communication Overhead

The proposed solution sends ECDH public keys in the association request and response messages. This has a total communication overhead of 16 bytes, which falls within Zigbee's maximum payload of 127 bytes [23,24]. Considering that Zigbee's standard data rate is 250 Kbit/sec and that more than 517 bytes will be transmitted during the entire join process, the resulting communication overhead remains relatively minimal.

D. Energy Overhead

In this experiment, the software executes a loop of ECDH point calculations to determine the energy consumption of one ECDH point calculation. An oscilloscope probe is connected to a shunt resistor to measure the potential difference generated by the current flowing through it, as shown in Fig. 11. Using the known resistance of the shunt resistor and the measured voltage, the current is calculated using Ohm's law.

As illustrated in Fig. 12, the mean voltage, determined as 2.9 mV, is the average voltage during the oscilloscope's acquisition period. The

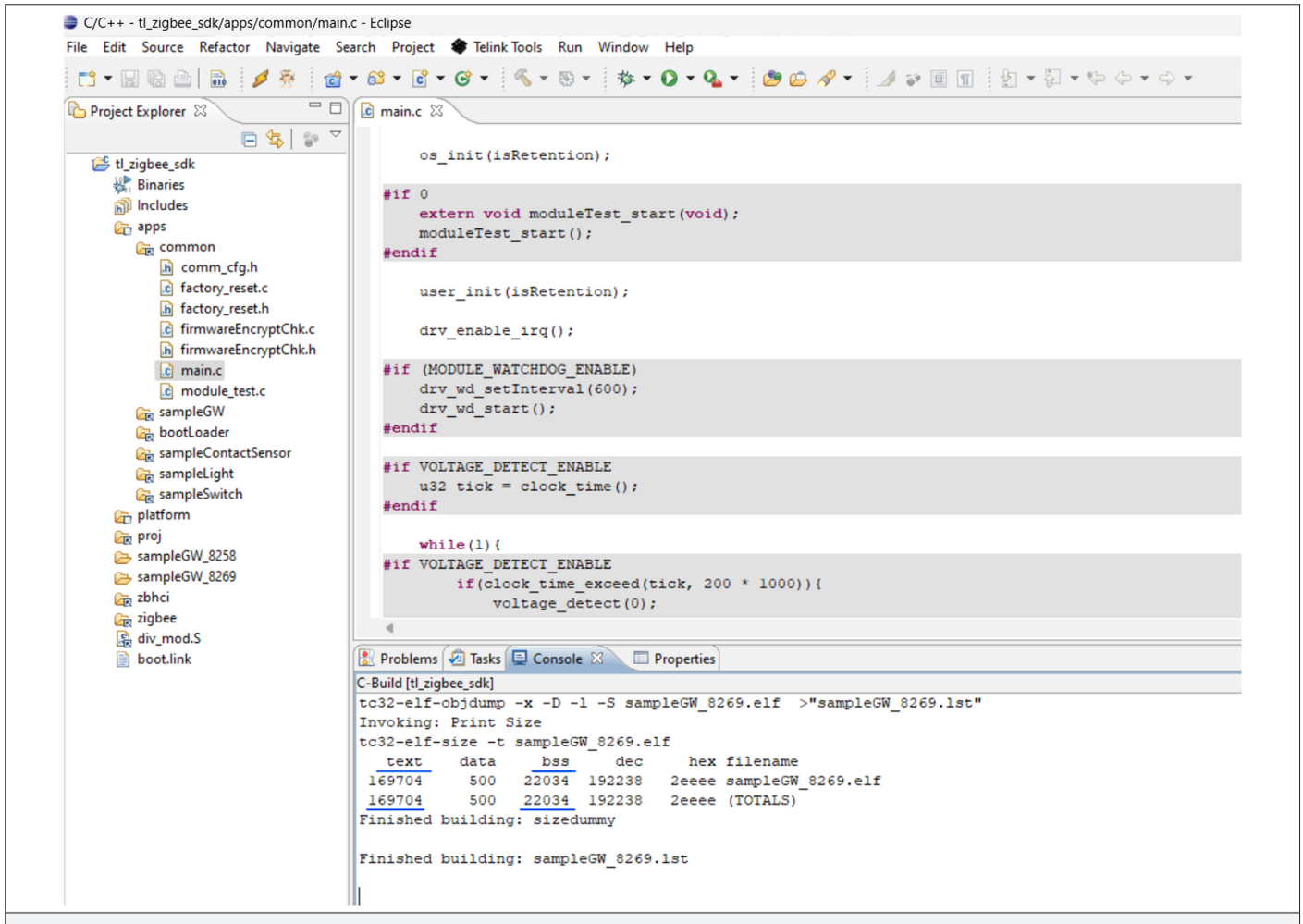

Fig. 9. Code size without ECDH.

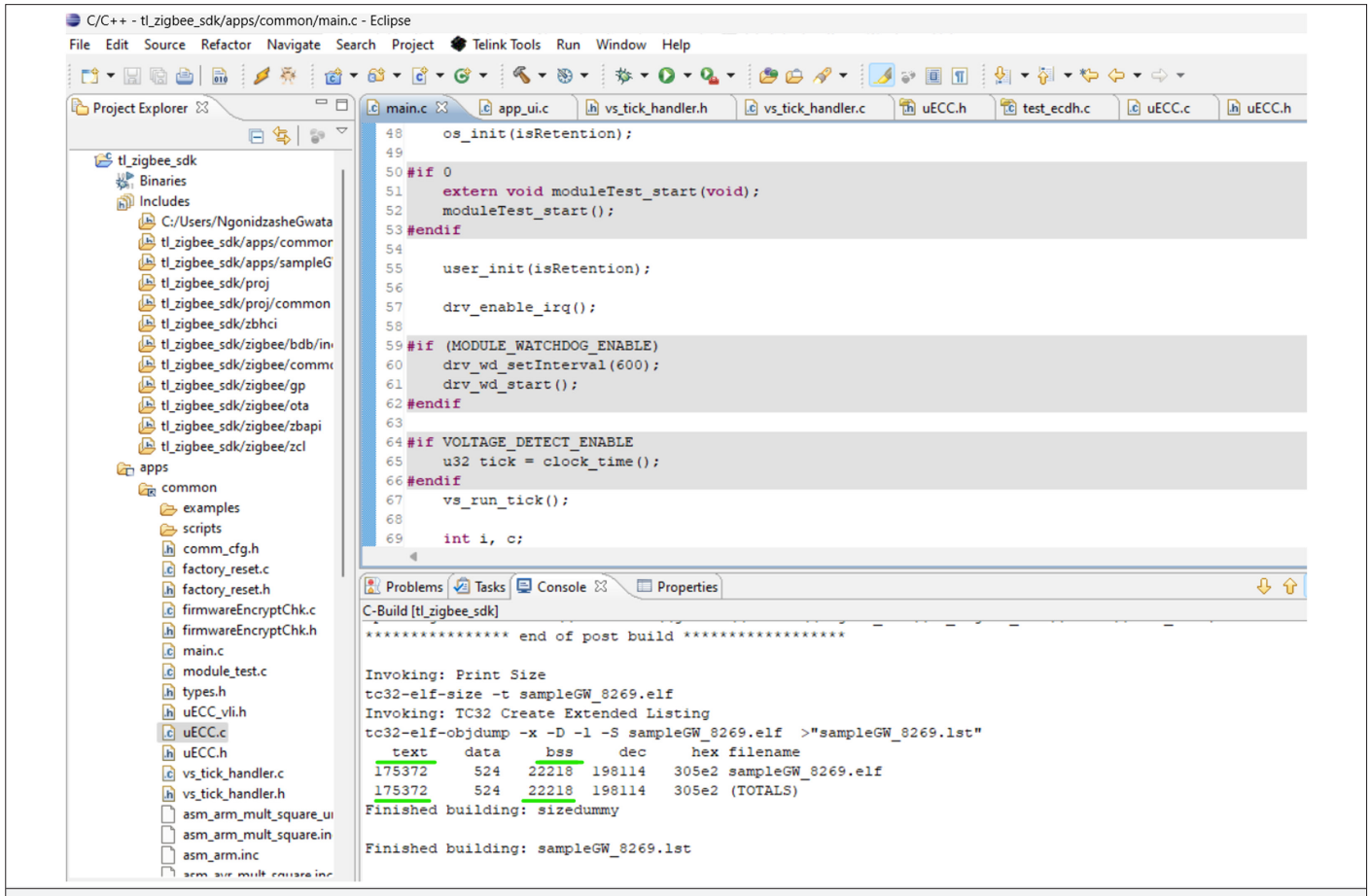

Fig. 10. Code size with ECDH.

total period, representing the time for the ECDH algorithm to run, is 275.8 milliseconds.

The calculations are as follows:

Average Current: 0.0029 V/10 Ω = 0.29 mA

Total Current Consumption: 0.29 mA × 275.8 ms = 79.982 mA.ms

Power: 3 V × 0.00029 A = 0.87 mW

Energy Used: 0.87 mW × 0.00007661 hours = 66.6507 nWh

Thus, 66.6507 nWh is the energy consumed in a single ECDH point calculation.

The average energy consumption for a 5-volt battery-powered Zigbee device during the joining process is around 60 mW on the Atmel microcontroller [25]. In the proposed solution, two ECDH multiplications are performed, that consume an additional 0.87 mW. This extra load accounts for less than 2% of the overall energy used during the join.

TABLE I. OPERATION OVERHEAD

Total Operation Overhead	
Total time overhead	275.8 ms
Memory overhead (code)	5.6 KB
Memory overhead (constants)	186 bytes
Communication overhead	17 + 17 bytes

E. Comparison With Existing Proposals

The new protocol is compared with earlier researchers' proposals, assessing various aspects, including:

- The additional messages needed.
- Whether a Certification Authority (CA) or Distributed Authority (DA) is necessary.
- Operation, memory, and communication overhead.

F. Extra Messages

The solution presented in this work does not necessitate the use of additional messages. Consequently, compared to other methods, it has lower communication overhead. Certain approaches, such as those described in [12] and [26], add extra messages. For example, these references have 3 and 4 more messages than the current Zigbee protocol, respectively.

G. CA/DA Requirement

This solution offers a certificateless model, eliminating the need for a centralized CA or DA like [9] and [26]. Unlike protocols relying on CA/DA, using such models would significantly delay the joining process. Additionally, it can be difficult to set up a single authority for Zigbee networks with multiple vendors because of issues with mutual trust.

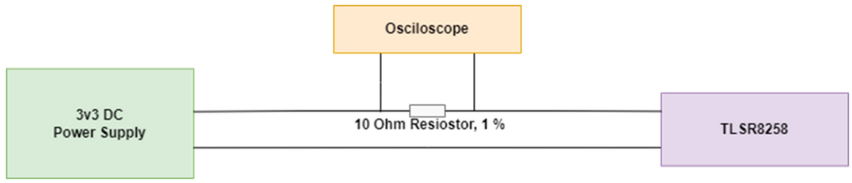

Fig. 11. Oscilloscope connection

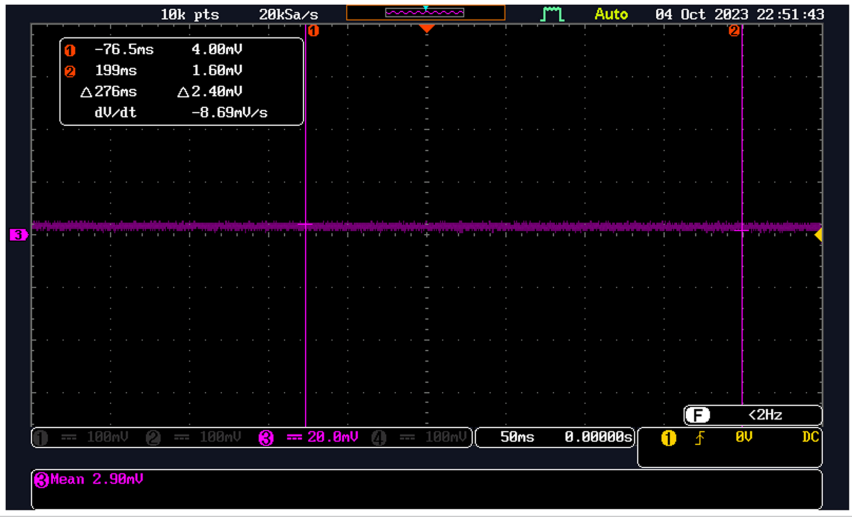

Fig. 12. Oscilloscope voltage measurement

H. Operation, Memory, and Communication Overhead

This solution offers improved efficiency in terms of operation, communication, code, and constant memory overheads. It outperforms the current solutions reviewed in this paper. Additionally, its smaller memory footprint makes it a more cost-effective and practical choice for implementation.

I. Potential Challenges

Implementing the proposed ECDH-based security solution in real-world IoT environments can present several challenges and considerations. This includes the following:

- In large-scale IoT deployments, the increased computational and communication overhead associated with ECDH might impact overall network performance
- Existing Zigbee networks often include a mix of devices with varying capabilities. Integrating ECDH into these networks requires ensuring that older devices, which may not support ECDH, can still operate effectively. This may necessitate updating the firmware of existing devices. This process can be complex and may introduce risks of firmware bugs or failures.

J. Security Robustness

The strength of the security enhancement provided by ECDH needs to be assessed. Future research may evaluate the robustness by testing the system against various attack vectors such as key interception, replay attacks, and unauthorized access attempts. Additionally, cryptographic analysis can be performed to assess the strength of ECDH encryption. The effectiveness of the ECDH integration will then be determined by its ability to withstand these attacks and protect the network data.

VII. CONCLUSION

Previous studies have identified several security vulnerabilities in Zigbee networks. These include unauthorized access, message interception, and replay attacks. Such weaknesses pose significant risks to the confidentiality and integrity of data transmitted through Zigbee networks. This study comprehensively examined these security issues while devising an efficient solution that effectively addresses them while maintaining compatibility with low-power, low-cost devices

To secure the Zigbee join process without imposing excessive costs or energy consumption on the system, ECDH was implemented. This modification encrypts the network key using ECDH-generated security, resulting in an effective and efficient method for key transportation.

This research contributes to IoT security by offering an enhanced approach to safeguarding Zigbee networks. This reinforces the safety measures for IoT devices and secure data transmission. This contribution is crucial for industries reliant on secure digital communication, offering a practical solution to current cybersecurity challenges.

Future research could focus on how to detect malicious devices in the network and determine appropriate actions following a security breach. This can be through monitoring the behavior of devices to detect any anomalies or deviations from expected patterns. Malicious devices may exhibit unusual communication patterns, such as excessive messaging or unexpected commands. Machine learning algorithms can be used to establish a baseline of normal network behavior and detect deviations. These models can help

identify suspicious activities that could indicate the presence of a malicious device.

In this paper, it is concluded that the security of Zigbee can be improved. Additionally, a method to secure network key transmission has been proposed, which can be adopted on every device within the Zigbee network. This advancement has the potential to improve data protection in various industries, making digital communications safer.

Availability of Data and Materials: The data that support the findings of this study are available on request from the corresponding author.

Peer-review: Externally peer-reviewed.

Author Contributions: Concept – N.G., V.B.; Design – N.G.; Supervision – V.B.; Resources – V.B.; Data Collection and/or Processing – N.G., V.B.; Analysis and/or Interpretation – N.G.; Literature Search – N.G., V.B.; Writing – N.G.; Critical Review – N.G., V.B.

Declaration of Interests: The authors have no conflict of interest to declare.

Funding: The authors declared that this study has received no financial support.

REFERENCES

1. E. Curryer, "IoT in 2023 and beyond," *Tech.Informed*. [Accessed: Oct. 25, 2023]. [Online]. Available: https://techinformed.com/iot-in-2023-and-beyond/.
2. *DFRobot*, *"Comparison of Wireless Technologies: LoRaWAN and Zigbee, WiFi, NB-IoT - DFRobot."* [Accessed: Oct. 25, 2023]. [Online]. Available: https://www.dfrobot.com/blog-1646.html.
3. *PRNewswire*, *"Zigbee PRO 2023 Improves Overall Security While Simplifying Experience."* [Accessed: Oct. 25, 2023]. [Online]. Available: https://www.prnewswire.com/news-releases/zigbee-pro-2023-improves-overall-security-while-simplifying-experience-301795113.html.
4. A. Allakany, A. Saber, S. M. Mostafa, M. Alsabaan, M. I. Ibrahem, and H. Elwahsh, "Enhancing security in ZigBee wireless sensor networks: A new approach and mutual authentication scheme for D2D communication," *Sensors (Basel)*, vol. 23, no. 12, art. no. 12, 2023. [CrossRef]
5. M. Ren, X. Ren, H. Feng, J. Ming, and Y. Lei, "Security analysis of Zigbee protocol implementation via device-agnostic fuzzing," *Digital Threats*, vol. 4, no. 1, 1–24, 2023. [CrossRef], p. 9:1–9:24.
6. R. Saker, and O. Abu Issa, "Assessing the security of a ZigBee smart HomeNetwork," 2024. [Online]. Linnaeus University. Available: https://lnu.diva-portal.org/smash/get/diva2:1837833/FULLTEXT01.pdf.
7. E. Barker, "Recommendation for key management Part 1: General," *Natl Inst. Stand. Technol. NIST SP*, p. 800-57pt1r4, 2020. [CrossRef]
8. P. Morgner, S. Mattejat, Z. Benenson, C. Müller, and F. Armknecht, *"Insecure to the touch: attacking ZigBee 3.0 via touchlink commissioning," in Proceedings of the 10th ACM Conference on Security and Privacy in Wireless and Mobile Networks*. Boston MA: ACM, 2017, pp. 230–240. [CrossRef]
9. S. Misra, S. Goswami, C. Taneja, and A. Mukherjee, "Design and implementation analysis of a public key infrastructure-enabled security framework for ZigBee sensor networks," *Int. J. Commun. Syst.*, vol. 29, no. 13, pp. 1992–2014, 2016. [CrossRef]
10. J.-Y. Lee, W.-C. Lin, and Y.-H. Huang, "A lightweight authentication protocol for Internet of Things.","," presented at the International Symposium on Next-Generation Electronics, ISNE 2014, 2014, pp. 1–2. [CrossRef]
11. S. Kulkarni, U. Ghosh, and H. B. Pasupuleti, *Considering Security for ZigBee Protocol Using Message Authentication Code*, 2015, pp. 1–6. [CrossRef]
12. K. Choi, M. Yun, K. Chae, and M. Kim, "An enhanced key management using ZigBee Pro for wireless sensor networks," in The International Conference on Information Network, 2012, pp. 399–403. [CrossRef]
13. H. Kadhum, "Enhancing Zigbee security for industrial implementation," 2020. [Online]. Kth Royal Institute of Technology. Available: http://kth.diva-portal.org/smash/get/diva2:1460818/FULLTEXT01.pdf.
14. X. Wang, and S. Hao, *"Don't Kick Over the Beehive: Attacks and Security Analysis on Zigbee," in Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*. Los Angeles CA USA: ACM, 2022.
15. F. Farha, and H. Ning, "Enhanced timestamp scheme for mitigating replay attacks in secure ZigBee networks," in IEEE International Conference on Smart Internet of Things (SmartIoT), 2019, pp. 469–473. [CrossRef]
16. X. Fan, F. Susan, W. Long, and S. Li, "Security analysis of Zigbee," 2017. [Online]. Available: https://courses.csail.mit.edu/6.857/2017/project/17.pdf.
17. S. Khanji, F. Iqbal, and P. C. K. Hung, "ZigBee security vulnerabilities: Exploration and evaluating," in 10th International Conference on Information and Communication Systems (ICICS). Irbid, Jordan: IEEE Publications, 2019, pp. 52–57. [CrossRef]
18. T. Goodspeed, "Extracting keys from second generation Zigbee chips," 2009. [Online]. Available: https://paper.seebug.org/papers/old_sebug_paper/Meeting-Documents/Blackhat-USA2009/BHUSA09-Goodspeed-ZigbeeChips-PAPER.pdf.
19. S. M. Rana, M. R. Hoque, and M. H. Kabir, *Evaluation of Security Threat of ZigBee Protocol to Enhance the Security of ZigBee Based IoT Platform*, vol. 11, no. 1, Jul. 2018, pp. 37–35.
20. T. Anderson, "Time to patch your lightbulb? Researchers demonstrate Philips Hue exploit." [Accessed: Sep. 16, 2022]. [Online]. Available: https://www.theregister.com/2020/02/05/time_to_patch_your_lightbulb_researchers_demonstrate_philips_hue_exploit/.
21. E. Ronen, A. Shamir, A.-O. Weingarten, and C. O'Flynn, "IoT goes nuclear: Creating a ZigBee chain reaction," in IEEE Symposium on Security and Privacy (SP), 2017, pp. 195–212. [CrossRef]
22. R. Li, W. Zhang, L. Wu, Y. Tang, and X. Xie, "ZPA: A smart home privacy analysis system based on ZigBee encrypted traffic," *Wirel. Commun. Mob. Comput.*, vol. 2023, pp. 1–16, 2023. [CrossRef]
23. K. MacKay, "micro-ecc," 2023. [Accessed: May 12, 2023]. [Online]. Available: https://github.com/kmackay/micro-ecc.
24. Zigbee Alliance, "Zigbee specification," 2017. [Online]. Available: https://csa-iot.org/wp-content/uploads/2022/01/docs-05-3474-22-0csg-zigbee-specification-1.pdf.
25. Atmel, *AT03663: Power Consumption of ZigBee End Device*, 2015.
26. P. Tedeschi, S. Sciancalepore, A. Eliyan, and R. Di Pietro, "LiKe: Lightweight Certificateless key agreement for secure IoT communications," *IEEE Internet Things J.*, vol. 7, no. 1, pp. 621–638, 2020. [CrossRef]

Vipin Balyan received his BE degree in electronics and communication with honours in 2003, his MTech degree in electronics and networking from LaTrobe University, Bundoora, Melbourne, Australia in 2006. He received PhD degree in Efficient Single Code Assignment in OVSF based WCDMA Wireless Networks in 2013. He is currently working as a senior lecturer in Department of Electrical, Electronics and Computer Engineering of Cape Peninsula University of Technology, Bellville Campus, Cape Town, South Africa. His research area includes CDMA, OFDM, 5G, Visible light communication and Fuel Cells. He is a National Research Foundation (NRF) rated researcher, he received departmental Teaching Excellence Award in 2022, University Bronze award for publication in 2020, Platinum Award in 2021 for publication and Bronze award for post graduate supervision in 2022. He received a funding of more than 2 million rands (equivalent to 1 crore rupees) from Hydrogen South Africa.

Ngonidzashe Gwata is an electrical engineering master's student at the Cape Peninsula University of Technology (CPUT) and has a strong foundation in electronic engineering. He graduated with a BTech degree in electronic engineering from the Harare Institute of Technology in 2012, after which he focused his research on embedded systems. His particular interest lies within RF communication. Currently pursuing his master's degree, Ngonidzashe shows great commitment to expanding his knowledge and contributing valuable insights to the intersection between embedded systems and RF communication technologies.