Digital Signature Method to Overcome Sniffing Attacks on LoRaWAN Network

Original Scientific Paper

Rahayu Indah Lestari
Telkom University, School of Computing
Jl. Telekomunikasi no. 1, Bandung, Indonesia
rahayuindahlestarii@student.telkomuniversity.ac.id

Vera Suryani*
Telkom University, School of Computing
Jl. Telekomunikasi no. 1, Bandung, Indonesia
verasuryani@telkomuniversity.ac.id

Aulia Arif Wardhana
Telkom University, School of Computing
Jl. Telekomunikasi no. 1, Bandung, Indonesia
auliawardan@telkomuniversity.ac.id

Abstract *– LoRa or Long Range with LoRaWAN technology is a protocol for low-power wireless networks. The absence of an encryption process on the data payload becomes a challenge for the LoRaWAN network. When the process of sending messages is running inter devices, sniffing might occur, thereby reducing the confidentiality aspect of the data communication process. This paper optimized the digital signature method to secure messages sent by LoRaWAN network devices, along with Advanced Encryption Standard (AES) algorithm and Ed25519 algorithm. AES was used for message encryption, while Ed25519 was used for signature purposes. The aim of applying digital signatures in this paper was to verify that the payload data sent was original and not changed during the transmission process and to ensure data confidentiality. The addition of security mechanisms to the LoRaWAN network, such as the process of encryption, decryption, and verification results, has caused some overheads. The overhead caused by the usage of a digital signature is also analyzed to ensure that the digital signature is feasible to be implemented in LoRa devices. Based on the experimental results, it was found that there was an increase in the size of memory usage and some additional processing delay during the deployment of digital signatures for LoRa devices. The overall overhead caused by implementing digital signatures on the LoRa devices was relatively low, making it possible to implement it on the LoRa network widely.*

Keywords*: LoRaWAN, sniffing, digital signature, AES, overhead*

1. INTRODUCTION

Internet of Things (IoT) refers to a communication paradigm to build the interactions between machines without any human interference [1]. IoT networks can be classified based on their physical radio layer, achievable bit rate, and power consumption or communication range. A network that operates remotely, use low power, and is able to tolerate low bit rates tend to use network technology such as LoRa [2].

LoRaWAN is a network infrastructure based on the Long Range (LoRa) radio modulation technology with some security flaws [3]. Payload data is not protected by encryption, making it subject to the sniffer.

Sniffing is the process of snooping the data packet on a network system. Some of which can monitor and capture all passing network traffic, regardless of who will receive the packet. During the sniffing process, it is potential to emerge an attack on LoRaWAN devices when receiving data from other devices.

The lack of security protection on LoRaWAN networks, which makes sniffing activities susceptible to attacks, is the main research problem of this paper. The digital signature is utilized to anticipate any attacks that sniffing operations may induce. This method aims to enhance the authentication aspect during data transmission of LoRaWAN communication. Furthermore, the purpose

of this digital signature is to ascertain that the content being transmitted does not change until they reach the recipient; thus, the receiver may be confident that the message received is genuinely original from the sender [4]. The digital signature is not a new method, but it is an alternative solution to encountering sniffing attacks on LoRa networks. Digital signatures are well suited to identifying valid users involved in the LoRa network's data communication process. Digital signatures are frequently used in software distribution, financial transactions, and other situations where modification or forgery must be detected.

The encryption algorithm of the digital signature implemented in this paper Advanced Encryption Standard (AES), and Ed25519 algorithm. The AES algorithm was used considering that it is lightweight and efficient in both software and hardware, and it can be applied to the digital signature method [5]. For encryption and decryption purposes, the AES variant applied were AES 128 and AES 256. The Ed25519 algorithm was selected for the signature process because it applies the Curve25519 algorithm in which the algorithm is compatible and found more efficient to be applied to the digital signature method. The overhead computation is the performance parameter of the proposed method in this research. The computed overhead consists of the change of the data payload size, the memory usage of the device executing the application, the RAM utilization, and the response processing time between the sender and receiver.

This study was conducted to investigate the usage of digital signature to prevent data sniffing in LoRa devices during data communication. The addition of a security mechanism to the network will certainly produce computational overhead on the system. This computational overhead was computed to determine how many additional resources are required when a digital signature is utilized. The parameters of the system's overhead analysis are payload length, memory usage, RAM usage, and response processing time [6]. This evaluation aims to determine the feasibility of applying digital signatures to LoRa devices.

The remaining sections of the paper are structured as follows. Section 2 provides a brief summary of relevant works or the current state of the art about other techniques for addressing security vulnerabilities in the Lora Network. In Section 3, the architecture of the proposed approaches is explained. In Section 4, the authors assess the proposed solution and show the experiment results. Section 5 concludes with a summary of the investigation's findings.

2. RELATED WORKS

Many studies have been conducted using digital signature and other methods in preventing the attack on the LoRaWAN network devices caused by the sniffing process. Table 1 depicts the comparison of related studies regarding this problem. Paper [7] compares traditional and new methods to deal with selective jamming attacks. The new methods suggested are game-theoretic approaches and the usage of machine learning. These two new methods significantly impact the detection of selective jamming attacks rather than the traditional ones.

Due to the growing usage of LoRa and the expansion of IoT devices, the paper [4] explains how to avoid sniffer activities on wireless sensor networks, particularly in LoRa networks. The AES and MAC algorithms are implemented in the LoRa network to protect data during transmission. The overhead analysis of IoT constrained devices class 0, and class 2 was also explored in this paper, with the findings indicating that these two algorithms could be applied to these devices. The network architecture in this research was still a local network. As a result, it is believed that additional study would enable this technology to be implemented on the LoRaWAN network, allowing data to be accessible over the internet.

Paper [8] analysis of LoRaWAN and its future directions focused on the threat of LoRaWAN, such as physical capture of end devices, sniffing gateways, and self-replay processes. These threats required particular attention from developers and organizations implementing LoRa networks. The problems that occurred were about the comprehensive security risks of the protocol and the way to find solutions to these security risks. Hence, the results and advantages obtained are the creation of a threat catalog for LoRaWAN by conducting discussions and analysis from the perspective of scale, impact, possibilities of each threat, and the drawbacks that may have an impact on several relevant network device security threats.

Paper [9] entitled Onboarding and Software Update Architecture for IoT was focused on Ed25519 as a derivative of the signature of EdDSA scheme. The Ed25519 algorithm applied a symmetric key using SHA- 512, a member of the SHA-2 family in the hashing process. The result showed that EdDSA provided attack resistance equal to 128-bit symmetric ciphers, using a 16-byte public key and a signature key of 64 bytes for the Ed25519 algorithm. This paper is used as a reference for designing the Lora system in this research.

Meanwhile, paper [10] discusses experimental tests focusing on the energy efficiency and security of LoRaWAN end devices (WisNode RAK811 and Seeeduino SX1301). In the security aspect, the experiment depicted that Activation By Personalization (ABP) mode is a more energy-efficient solution that comes at the sacrifice of security. Due to the lack of a join method, the ABP mode exchanges fewer messages. ABP offers an additional security risk because it relies on counter values maintained in memory and is unable to renew session keys. The end device will go into an out-of-sync condition and become useless if there is a problem retaining or reading these settings. WisNode devices are more vulnerable to physical memory assaults due

to this security feature. An OTAA system that provides secure session keys to safeguard communication is an option to secure these devices.

The authors of [11] stated that a dual-blockchain structure could be used to secure a LoRa-based information system. The algorithms used in this research are decentralized to reduce the dependency on a centralized server. Blockchain is also utilized for securely updating IoT device firmware using LoRa as a communication protocol [12].

Encrypt then Sign was the digital signature approach used in this research because when communication is exploited by a third party, the sender and receiver of the message can determine who is exploiting the message. Due to the fact that the signature key retrieved no longer belongs to the message's sender but rather to the sniffer party, the application of the Encrypt then Sign approach drastically reduces the likelihood of message exploitation. Using the Sign then Encrypt approach, when a sniffer exploits a message from the sender and forwards it to the receiver, the received message still has the sender's signature key. This is because the sniffing party only modifies the message from the decryption process in plaintext and not the signature key of the message's sender. Consequently, the sender and receiver cannot identify the sniffing party who compromised the message.

Table 1. Comparison with previous methods

Reference	Attack/Vulnerability Type	Techniques	Security Aspect
[7]	Selective jamming	Game-theoretic approaches and reinforcement machine learning methods	Integrity
[4]	Sniffing	Advanced Encryption Standard (AES) and Message Authentication Code (MAC)	Confidentiality, Integrity
[8]	Device Cloning, Self-Replay, Rogue End-Device	Tamper-resistant hardware, Public Certificate, End-to-End Encryption	Confidentiality, Integrity
[9]	Software update, MiTM	Elliptic curve cryptography (Curve25519), authenticated key establishment, and a public key encryption	Authentication
[10]	Remote access of IoT device	Over- The-Air Activation (OTAA) and exchanging keys	Authentication
[11]	Information asymmetry	Blockchain-based LoRa-IS combined with contract theory	Authentication
[12]	Software update, MiTM	Blockchain-based system for securely updating IoT device firmware	Privacy, Authentication
This paper	Sniffing	Digital signature using Advanced Encryption Standard (AES) algorithm	Authentication, Integrity

3. PROPOSED SYSTEM

This research used two 868MHz LoRa Shield Module devices and two Arduino Mega 2560 devices as node 1 sender and node 2 receiver. A Raspberry Pi device for LoRaWAN acted as a sniffer. The programming language used was C++ with the data type sent as string data. The attack scenario was conducted by testing a man-in-the-middle attack for the sniffing process. This attack was tested before the encryption algorithm was deployed and after the signature process was implemented. The goal is to determine whether or not the payload data sent has been modified. AES 128 with 128-bit key length and AES-256 with 256-bit key length were implemented in the experiment. Moreover, Ed25519 algorithm was deployed for the signature implementation. Table 2 shows the hardware specifications and scenarios used in the experiment. Fig. 1 illustrates the overall system architecture, where node 2 as the receiver would only react if the received data contained the same ID found in node 1 as the sender. If node 2 successfully received the message sent, then the node 2 device as the receiver would send an acknowledgment to node 1 informing that the message received was valid.

Fig. 2 illustrates the flowchart of sensor node 1 as a sender. The first process was to connect the sender to node 2. The plaintext message would be added with ID and message digest when the connection was established. This plain text was then encrypted to produce cipher text. The subsequent step would be checking the key used for the signing process. If the signing process is successful, the signature key and ciphertext message will be merged and delivered to the node 2 receiver.

Fig. 3 portrays the flowchart of sensor node 2 as the receiver. First, node 2 must be connected to sensor node 1 as the message sender device. If successfully connected, then node 2 would check the messages. Furthermore, the checking process was conducted for a total length of 80 bytes message, where 64 bytes was the length of the signature key, and 16 bytes was the length of the ciphertex message. If the value of message length was equal, i.e. 80 bytes; the following process is splitting the signature key and ciphertext message. The decryption procedure then required for the inversion of ciphertext into plaintext. Before beginning the decryption procedure, node 2 would match the signature key of node 1.

Table 2. Hardware and Scenario Specifications

No	Scenario	Hardware
1	Device 1 (communicate with device 2)	Arduino Mega 2560 Rev3 with Dragino LoRa Shield
2	Device 2 (communicate with device 1)	Arduino Mega 2560 Rev3 with Dragino LoRa Shield
3	Sniffing Device	Raspberry Pi 3 model B with Dragino LoRa Hat

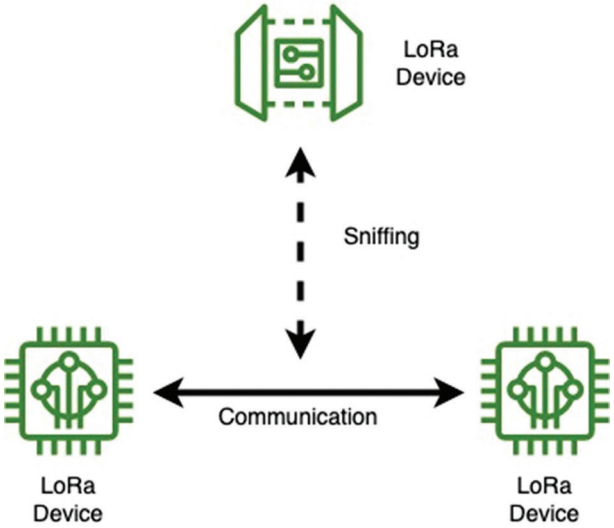

Fig. 1. System architecture

After discovering that the key is identical to the one used by node 1 during the signing procedure, decryption would be performed. The results obtained from the decryption process are sender ID, message digest, and plaintext; therefore, it is crucial to separate these three results. Following the process of splitting, the three values are stored. After the results have been saved, the receiver will verify that the sender is a legitimate user, not an adversary.

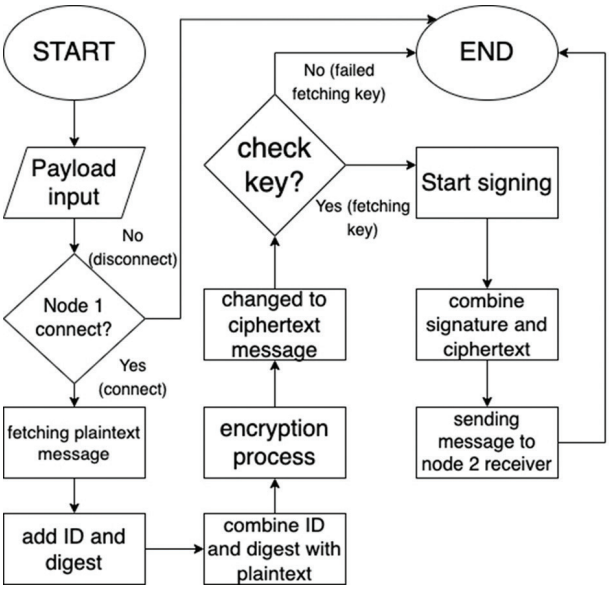

Fig. 2. Sender Flowchart

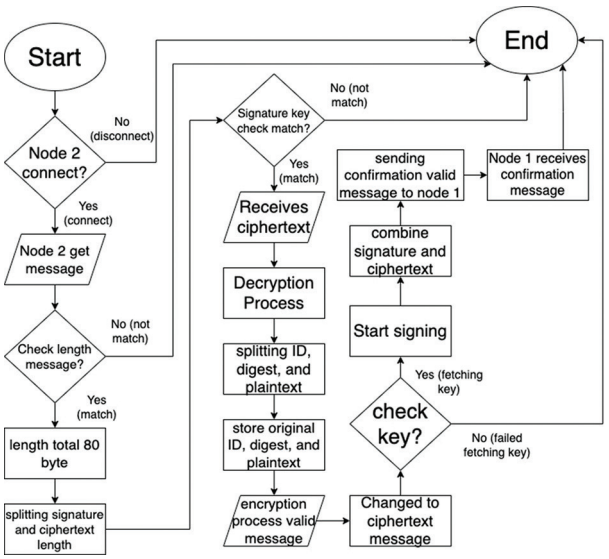

Fig. 3. Receiver Flowchart

4. RESULT AND DISCUSSION

Man-in-the-middle attack testing was applied for the sniffing process during the experiment. Three modes were set up on LoRa devices:

- mode 0 for "MODE_NON_SIGNATURE"
- mode 1 for "MODE_SIGNATURE_AES128"
- mode 2 for "MODE_SIGNATURE_AES256"

The experiment started with mode 0, followed by mode 1 and mode 2. Detail procedures are depicted on flowcharts in Fig. 2 and Fig. 3. Furthermore, Fig. 4 shows the test results of the sniffing process using mode 0 "MODE_NON_SIGNATURE". Mode 1 "MODE_SIGNATURE_AES128" is shown in Fig. 5 and mode 2 "MODE_SIGNATURE_AES256" is shown in Fig. 6.

After conducting the man-in-the-middle test for the sniffing procedure, the following test observed the system's overhead values. Similar 3 modes were utilized to evaluate the sniffing process for calculating the overhead values.

```
There are data has been sniffed.
Received:
30 31 74 65 73 74 69 6e 67
31 32 33 34 00 ab ac 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00
Packet RSSI: -50, RSSI: -106, SNR: 9, Length: 84
current time: 20-07-01 23:07:08:392
There are data has been sniffed.
Received:
52 45 43 45 49 56 45 52 5f
53 52 56 30 31 ba bc 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00
Packet RSSI: -58, RSSI: -106, SNR: 10, Length: 84
current time: 20-07-01 23:07:08:585
```

Fig. 4. MODE NON-SIGNATURE

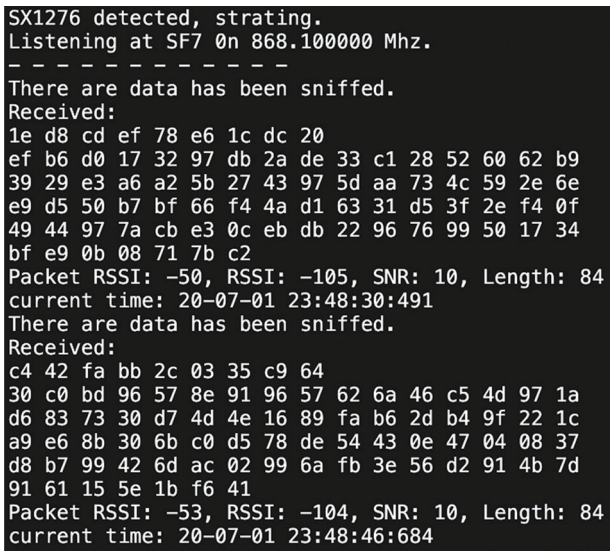

Fig. 5. MODE SIGNATURE AES 128

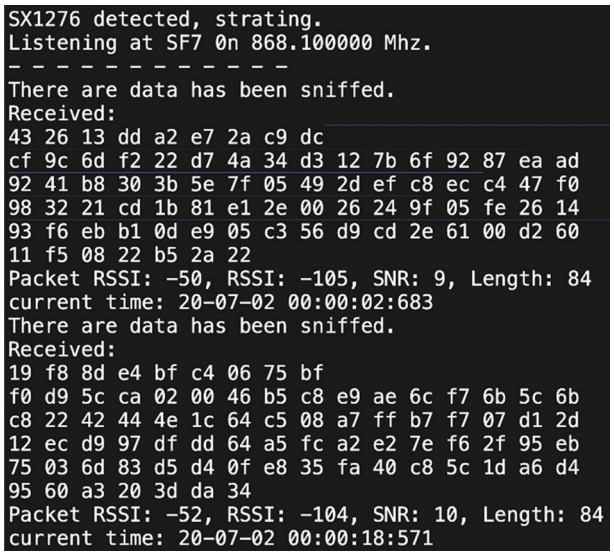

Fig. 6. MODE SIGNATURE AES 256

Overhead testing was done by sending string data from node 1 sender to node 2 receiver. The goal of the overhead analysis was to investigate the payload length, memory utilization, RAM usage, and response processing time characteristics. Table 3 shows the results of the overhead comparison of the three modes used.

As depicted in Fig. 4, the test result in mode 0 showed that the device which acted as a sniffer knew all the original payloads data of the two communicating node devices. LoRaWAN devices are susceptible to attacks and message alterations if the payload data is not encrypted. Meanwhile, for testing mode 1 in Fig. 5 and mode 2 in Fig. 6, it can be seen that the devices acted as the sniffer also knowing all communications between devices. However, the payload data obtained have been encrypted and signed so that the authenticity of the payload data could be well maintained.

This evidence shows that using digital signatures can reduce the potential for attacks due to its encryption process. Furthermore, Table 3 depicted the overhead testing outcomes for each sender and recipient. The first overhead analysis was the analysis of the length of the payload data. Based on the experiment findings shown in Table 3, Table 4 provides a more detailed description of the payload length test value. It can be seen that mode 1 and mode 2 used in the test had additional header data. In mode 0 the size of the payload length used was only 16 bytes, 4 bytes of which were the additional data consisting of 2 bytes of ID sender and 2 bytes digest, and the rest 12 bytes are considered as actual data.

Meanwhile, in mode 1 and mode 2 the payload length used was 80 bytes with 64 bytes were the additional header data, i.e., the signature key and 2 bytes of IDsender, 2 bytes of digest, and 12 bytes of actual data.

Mode	Additional data headers			Real Data	Payload
	Sender ID	digest	signature key		
Mode 0	2	2	0	12	16 bytes
Mode 1	2	2	64	12	80 bytes
Mode 2	2	2	64	12	80 bytes

Table 4. Detail Overhead Test Results

Parameters	Mode 0		Mode 1		Mode 2	
	S	R	S	R	S	R
Payload length	16 bytes	16 bytes	80 bytes	80 bytes	80 bytes	80 bytes
Memory usage	13506 bytes (5%)	12304 bytes (4%)	36770 bytes (14%)	35680 bytes (14%)	37080 bytes (14%)	35998 bytes (14%)
RAM usage	1754 bytes (21%)	1374 bytes (16%)	2564 bytes (31%)	2091 bytes (25%)	2644 bytes (32%)	2171 bytes (26%)
Delay (S to R)	174.64 ms	34.68 ms	16063.53 ms	9884.53 ms	15763.2 ms	9584.58 ms
Delay (R to S)	42.14 ms	169.07 ms	9683.87 ms	6187.79 ms	10004.4 ms	6187.77 ms

S = sender; R = receiver

The message length in mode 0 corresponded to the encryption key used in the AES algorithm, where the total key length was 32 bits. Hence, each AES algorithm was divided by 8 bits, so 128 bits = 16 bytes, and 256 bits = 32 bytes. Using AES 128 or AES 256, the size of the encrypted plaintext was only 16 bytes, independent of the AES algorithm library being used [13]. It is obvious that adding a header to modes 1 and 2 would result in a payload length that was 4 times bigger than it was in mode 0, which only used 12 bytes of actual data and 4 bytes of extra data. Consequently, the resulting payload length was indeed higher. Even though separate digital signature algorithms are utilized for modes 1 and 2, the payload length results produced for both modes are equal. Therefore, memory and RAM utilization on the system rose since the more program functionalities that were implemented, the greater memory and RAM consumption was required.

The fourth overhead analysis was a delay from sender to receiver. Table 4 shows that mode 1 and mode 2 produced a longer response time when the sender sent the payload data to the receiver, with the encryption process and added digital signing. After the message from the sender is successfully received, the receiver will carry out the signature key validation process and provide a key validation response to the sender. The fifth overhead analysis is the delay or response processing time from receiver to sender. The response time used in mode 1 and mode 2 was also longer than mode 0. This is because after the payload data were received, the receiver would confirm to the sender that the payload data received was a valid message. However, before the confirmation process was sent, the data payload must be converted into encrypted form, and the signing process was carried out first. Therefore, from the two results of response processing time testing on LoRaWAN devices, after applying the digital signature method, the transmission time was increased because the device was charged for several extra operations. Based on prior study, if the work cycle was applied to 1%; a node was allowed to send only for 36 seconds/hour or about 36 ms [14]. After establishing a security system which resulting a work cycle of 14%, the highest response processing time in this experiment was 16063.53 ms. Nevertheless, the system's utilization grows more secure.

5. CONCLUSION

Based on the experiment on the sniffing process and overhead calculations that have been conducted, it can be concluded that the digital signature method could secure the message sent between LoRa devices.

The results of the overhead analysis in this study showed that the use of digital signatures produced a high overhead value compared to those without implementation. The experiment results also revealed that this system was more efficiently applied to the AES 128 than AES 256 encryption algorithm. For future work, different security methods can be applied to improve the confidentiality, integrity, and availability aspects of LoRa network. The security method algorithm should be lightweight to be compatible with the characteristics of LoRa devices having limited resources.

6. REFERENCES:

[1] A. A. Laghari, K. Wu, R. A. Laghari, M. Ali, A. A. Khan, "A Review and State of Art of Internet of Things (IoT)", Archives of Computational Methods in Engineering, Vol. 29, No. 3, 2022, pp. 1395-1413.

[2] E. Aras, N. Small, G. S. Ramachandran, S. Delbruel, W. Joosen, D. Hughes, "Selective jamming of LoRaWAN using commodity hardware", Proceedings of the 14th EAI International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services, November 2017, pp. 363-372.

[3] N. Hayati, K. Ramli, S. Windarta, M. Suryanegara, "A Novel Secure Root Key Updating Scheme for LoRaWANs Based on AES DRBG 128", IEEE Access, Vol. 10, 2022, pp. 18807-18819.

[4] P. A. Windya, V. Suryani, A. A. Wardana, "Sniffing Prevention in LoRa Network Using Combination of Advanced Encryption Standard (AES) and Message Authentication Code (MAC)", Proceedings of the International Conference Advancement in Data Science, E-learning and Information Systems, Bali, Indonesia, 13-14 October 2021.

[5] M. A. Mughal, X. Luo, A. Ullah, S. Ullah, Z. Mahmood, "A lightweight digital signature-based security scheme for human-centered internet of things", IEEE Access, Vol. 6, 2018, pp. 31630-31643.

[6] H. Hidayat, P. Sukarno, A. A. Wardana, "Overhead Analysis on the Use of Digital Signature in MQTT Protocol", Proceedings of the International Conference on Electrical Engineering and Informatics, Bandung, Indonesia, 9-10 July 2019, pp. 87-92.

[7] D. Basu, T. Gu, P. Mohapatra, "Security Issues of Low Power Wide Area Networks in the Context of LoRa Networks", arXiv:2006.16554v1, 2020.

[8] I. Butun, N. Pereira, M. Gidlund, "Security risk analysis of LoRaWAN and future directions", Future Internet, Vol. 11, No. 1, 2018, pp. 1-22.

[9] H. Gupta, P. C. Van Oorschot, "Onboarding and Software Update Architecture for IoT Devices", Proceedings of the 17th International Conference on Privacy, Security and Trust, Fredericton, NB, Canada, 26-28 August 2019.

[10] M. Mehic, M. Duliman, N. Selimovic, M. Voznak, "LoRaWAN End Nodes: Security and Energy Efficiency Analysis", Alexandria Engineering Journal, Vol. 61, No. 11, 2022, pp. 8997-9009.

[11] G. Yu et al. "A novel Dual-Blockchained structure for contract-theoretic LoRa-based information systems", Information Processing & Management, Vol. 58, No. 3, 2021, pp. 1-23.

[12] A. Anastasiou, P. Christodoulou, K. Christodoulou, V. Vassiliou, Z. Zinonos, "IoT Device Firmware

Update over LoRa: The Blockchain Solution", Proceedings of the 16th International Conference on Distributed Computing in Sensor Systems, Marina del Rey, CA, USA, 25-27 May 2020, pp. 404-411.

[13] L. A. F. Fernandes, M. M. Oliveira, "Handling Uncertain Data in Subspace Detection", Pattern Recognition, Vol. 47, No. 10, 2014, pp. 3225-3241.

[14] D. Zorbas, K. Abdelfadeel, P. Kotzanikolaou, D. Pesch, "TS-LoRa: Time-slotted LoRaWAN for the Industrial Internet of Things", Computer Communications, Vol. 153, No. October 2019, 2020, pp. 1-10.