COMPUTERIZED AUTOMATIC SYSTEMS

ENCRYPTION KEY MANAGEMENT IN A WIRELESS MESH NETWORK

Halyna Vlakh-Vyhrynovska, PhD,As.-Prof., Rudyy Yuriy, PhD student
Lviv Polytechnic National University, Ukraine;e-mail: halyna.i.vlakh-vyhrynovska@lpnu.ua

https://doi.org/10.23939/istcmtm2024.04.

Abstract. Wireless MESH networks are important for providing communication in conditions where communication infrastructure is limited or absent. In such networks, encryption key management plays an important role in ensuring the security of data transmission. The work analyzes existing approaches to encryption key management in wireless networks and proposes a method for optimal key update. The method was tested on the LoRa EBYTE SX1262 chip with the ESP-WROOM-32. Algorithms for predicting update time were used.

Key words: Key, Wireless mesh networks, LoRa, Esp32, Cryptography.

1. Introduction

Wireless MESH networks are becoming increasingly important in the modern world, especially in conditions where traditional communication infrastructure is limited or completely absent. These networks allow for communication in remote areas, during emergencies, as well as in situations of military conflict. Considering the open nature of the wireless communication channel and the possibility of any device participating in communications, ensuring the security of data transmission is a critically important task [1-3].

A key component of security in such networks is effective encryption key management. This includes the generation, distribution, storage, and updating of keys. These keys are necessary to protect data from unauthorized access and to ensure the integrity of information. Despite numerous studies and developments in this field, existing approaches still have a number of drawbacks, such as high computational complexity, the need for frequent key updates, and the limited resources of devices used in MESH networks [4-7].

2. Drawbacks

In current implementations of key management in wireless MESH networks, there are several drawbacks:

– Some methods require frequent key updates to maintain security, which can lead to a significant increase in network traffic and additional load on its components.

– Devices used in MESH networks often have limited resources, such as memory and processing power, which complicates the use of complex cryptographic algorithms and key management procedures.

– The open nature of wireless communication channels and widely known protocols make MESH networks vulnerable to various types of attacks, such as interception, spoofing, and message replay.

– In decentralized MESH networks, there is no central controlling authority, which complicates the process of centralized key management and network security monitoring.

– Building efficient routing schemes to reduce communication channel load is a challenging task, especially under conditions of dynamic changes in the network [7].

3. Goal

The purpose of this work is the development and implementation of an optimized encryption key management system for wireless MESH networks that meets modern security and performance requirements.

4. Encryption keys Management in a Wireless MESH Network

Wireless networks use both symmetric and asymmetric cryptographic algorithms to ensure the security of data transmission.

Symmetric cryptographic algorithms use the same key for both encryption and decryption of data. This means that both parties involved in the data exchange must have access to the same secret key. AES is the most common symmetric algorithm used in wireless networks due to its high level of security and efficiency. AES supports keys of lengths 128, 192, and 256 bits [8].

Asymmetric cryptographic algorithms use a pair of keys: a public key for encrypting data and a private key for decrypting it. The public key can be known to everyone, while the private key is kept secret.

ECC (Elliptic Curve Cryptography) provides a similar level of security as RSA but uses smaller keys, making it more efficient for devices with limited resources. It is used for data encryption, key exchange, and digital signatures in wireless sensor networks and IoT.

Cryptochips ATAES132A from Microchip Technology are an important component in ensuring the security of Internet of Things (IoT) devices [9-12].

These chips are designed to provide secure authentication, data encryption, and key management,

making them ideal for use in various IoT applications. The ATAES132A uses unique secret keys for device authentication, providing protection against tampering (Fig.1). The chip supports various cryptographic operations, such as HMAC (Hash-based Message Authentication Code) for message authentication. The chip supports AES-128, which provides a high level of encryption for protecting data privacy in CTR (Counter) and CBC (Cipher Block Chaining) modes.

The ATECC608A cryptochip from Microchip Technology is a powerful tool for ensuring the security of Internet of Things (IoT) devices. It provides a high level of security through the use of hardware cryptographic capabilities, including authentication, data encryption, and key management. The chip supports ECC (Elliptic Curve Cryptography) cryptographic operations for secure authentication, AES-128 and AES-256 for data encryption, and SHA256 for generating hash functions that ensure data integrity. The built-in authentication algorithms allow for verification of the authenticity of devices connecting to the IoT network, preventing unauthorized devices from connecting.

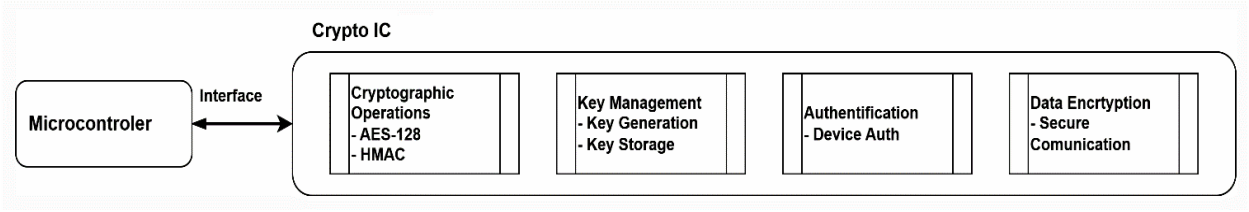

Fig.1. Cryptochips ATAES132A

The use of cryptochips allows for the implementation of access control to network resources based on cryptographically protected identifiers.

The ATECC608A generates a key pair (private and public) using ECC. The private key is stored inside the chip and never leaves it, providing a high level of protection. The public key can be sent to a CA (Certificate Authority) for certificate issuance. Other devices in the network can verify the authenticity of the device using the certificate issued by the CA. ECDSA (Elliptic Curve Digital Signature Algorithm) is used for digital signatures that confirm the authenticity of messages sent by the device. ECDH (Elliptic Curve Diffie-Hellman) is used to establish secure sessions between devices, allowing for secure data exchange.

Let us consider the Ajax alarm systems as one of the most technologically advanced security systems that utilizes modern cryptographic methods to ensure a high level of data protection and secure communication between devices. Ajax uses the AES-128 (Advanced Encryption Standard) algorithm for encrypting data transmitted between sensors and the central unit (hub). AES-128 provides a high level of security due to the complexity of the algorithm and the key length. The encryption is performed in Cipher Block Chaining (CBC) mode, which increases resistance to cryptographic attacks due to each ciphertext block's dependence on all previous blocks [13].

For the initial key exchange between devices, the asymmetric algorithm RSA (Rivest-Shamir-Adleman) is used. RSA allows for the secure exchange of encryption keys without the need for physical key transfer [14].

After the initial RSA key exchange, the system can use ECDH for exchanging session keys, providing a fast and secure key exchange with lower computational costs. To verify the integrity and authenticity of messages transmitted between devices, Ajax uses HMAC. This ensures that the message has not been altered during transmission.

The transmission of a new encryption key, encrypted with the recipient's individual key, is used to ensure secure key exchange between the sender and a specific recipient. The main idea is that the new encryption key is encrypted with the recipient's individual (public) key, ensuring that only the recipient can decrypt it. The sender generates a new session encryption key (e.g., AES-128). The new encryption key is encrypted using the recipient's public key (e.g., using RSA or ECC). The encrypted new key is transmitted to the recipient through the communication channel. The recipient receives the encrypted key and decrypts it using their private key. The recipient can now use the new session key for encrypting and decrypting subsequent messages.

The transmission of a new encryption key, encrypted with a group update key, is used to update encryption keys in a group of devices in a Mesh network (fig. 2). The new encryption key is encrypted with the group update key, which is known to all devices in the group. This allows for the simultaneous update of encryption keys for all devices in the group. The administrator or central device generates a new session encryption key. The new encryption key is encrypted with the group update key, which is known to all devices in the group. The encrypted new key is transmitted to all devices in the group through a secure communication channel. Each device in the group receives the encrypted key and decrypts it using its copy of the group update key. The devices can now use the new session key for encrypting and decrypting subsequent messages.

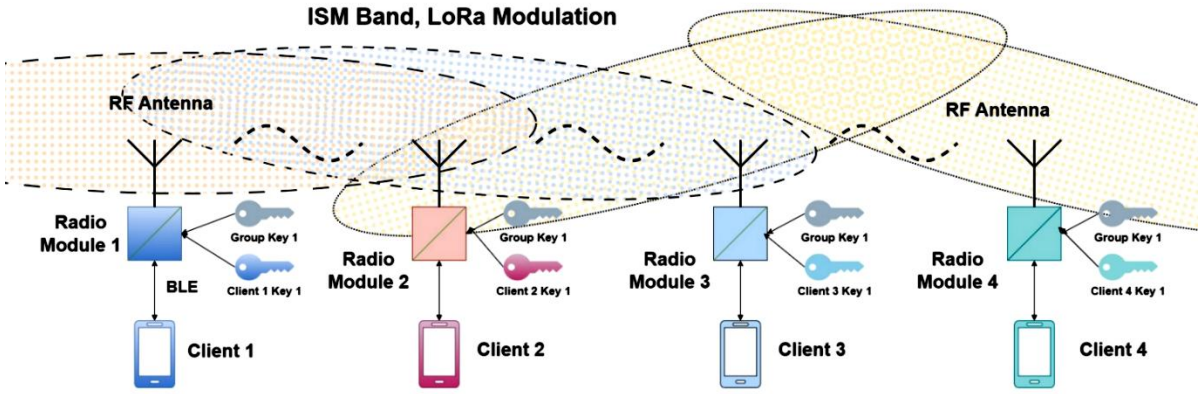

Fig.2. Mesh network

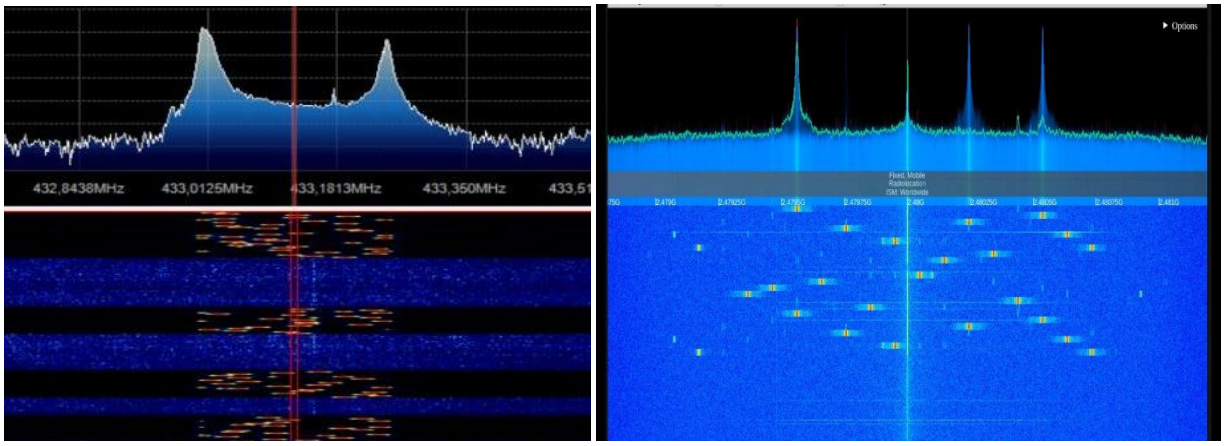

Fig 3. Normal frequrency vs hopping

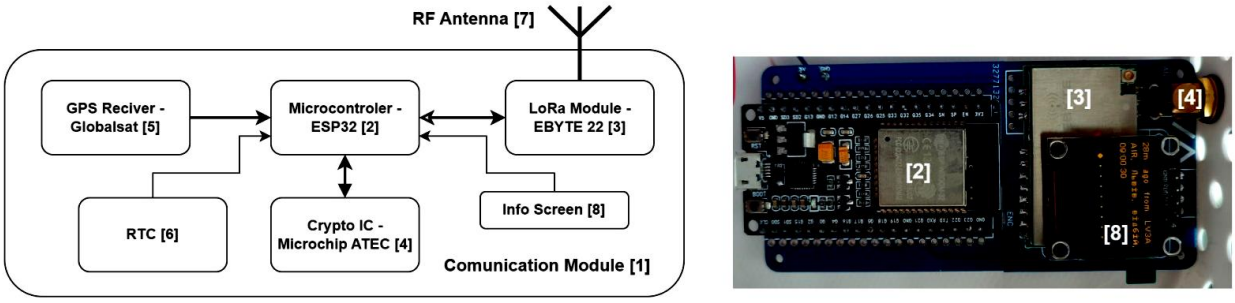

Fig. 4. Test Hardware Platform

When distributing encryption keys, we may encounter the problem of radio frequency noise in certain areas of the network (see Figure 3). This can be due to both random interference and the consequences of deliberate suppression of radio frequencies by electronic warfare means. To increase the likelihood of data packet delivery to all participants in the network, we can introduce a retransmission coefficient as well as message prioritization. Pseudo-random frequency hopping methods may also be used.

To test the algorithms, the open-source software Meshtastic was used as a base [15]. The hardware platform consists of the ESP32 Wroom, the EBYTE E22 400M30S radio module with the SX1268 transceiver, and a GPS receiver (fig. 4).

An important role in this scheme is played by the precise synchronization of the time of all nodes. GPS provides very accurate timestamps, allowing all nodes in the network to have coordinated time. This is critically important for coordinating actions such as data transmission, key updates, and performing cryptographic operations. Timestamps are used to determine the moments for updating encryption keys. This allows for automatic and synchronous updates of keys across the

network, ensuring high security. The developed service allows: viewing the list of all devices, their status, power availability, parameters of stored keys, last exchange time, and signal level.

In the key management menu, we see the current network key, the validity period, and additional keys that will come into effect after a certain time.

It is possible to formulate a request for a new key, specify the duration of its action, the condition, or whether additional confirmation is needed for the readiness of the entire network to work with the new keys (needed in cases where some nodes do not receive the new key)

Also indicate the security level, namely: whether the key should be transmitted encrypted with a group key or an individual key (this is necessary in case of loss or compromise of one of the nodes). During the transitional period, when the transition to new encryption keys is taking place, the possibility of decrypting messages with the old key remains. The time should not exceed 5 seconds. After that, the old key is destroyed in the storage.

"After confirmation from all nodes regarding the update of the encryption keys, the network can switch to the new keys. Advantages of the system:

1. Remote updating of message encryption keys
2. Setting time frames for key validity
3. Confirmation of updates from end nodes
4. Logging of all events
5. Profitability: reduction of costs associated with changes, absence of physical connection to end devices, and reduction in man-hours for maintenance personnel."

Let’s consider the process of secure key management in low-power IoT networks.

Encryption key updates in LoRaWAN [7].

LoRaWAN (Long Range Wide Area Network) is a specification for low-power wireless communications that uses LoRa technology to provide large coverage. Updating encryption keys in LoRaWAN is important to maintain network security [16].

The main types of keys in LoRaWAN include AppKey (Application Key), which is responsible for generating other keys during the device activation process; NwkSKey (Network Session Key), which is used to ensure data integrity and authenticity at the network layer, and AppSKey (Application Session Key) for encrypting data at the application layer.

The Activation by Personalization (ABP) method assumes that the keys are set manually and remain static throughout the device's entire operation.

ABP has security limitations, in that keys are never updated, which can lead to their compromise. If the keys are compromised, the network administrator must manually change the keys on the device and server. translation of the text into English.

The Over-the-Air Activation (OTAA) method implies that during activation, the device initiates the process of connecting to the network through the Join Request message. The device sends a Join Request message that contains the DevEUI, AppEUI, and DevNonce. The network server responds with a Join Accept message, which is encrypted with an AppKey.

After successful activation, NwkSKey and AppSKey are generated using the AppKey. These keys are used to secure further communications between the device and the server. The keys can be automatically updated when the device is reactivated via OTAA, which ensures high security.

Regular re-activation of devices using OTAA ensures frequent key updates, which reduces the risk of compromise. The DevNonce is a unique number that is used during each OTAA activation to prevent old Join Request messages from being reused. Keys should be stored in a secure location on the device, such as in a secure memory or cryptographic module, which provides an additional layer of protection against physical attacks.

Key Management in NB-IoT.

NB-IoT (Narrow Band Internet of Things) is a cellular communication standard that offers a reliable way to connect many low-power devices over long distances, allowing them to exchange small amounts of data [17].

In NB-IoT, key management is of great importance for secure communication between devices and the network. The process consists of several stages, starting with device authentication and key generation.

In the first stage, each NB-IoT device is identified using a unique IMSI (International Mobile Subscriber Identity), which is stored on a SIM card or eSIM.

Next, the device authentication process is performed using the network operator's Authentication Center (AuC), which maintains the IMSI along with the corresponding Ki keys.

Ki (subscriber authentication key), which is required for generating authentication and encryption keys, is stored both on the device's SIM card and in the AuC.

During the authentication process, the network sends a random number to the device, known as RAND (Random Number).

Using Ki and RAND, the network generates an authentication response (XRES) along with the encryption key (CK) and the integrity key (IK)

Authentication.

The network sends RAND and a request for device authentication. The device uses KiK_iKi and RAND to calculate the responses XRES, CK, IK using an algorithm.

The device sends XRES back to the network.

The network calculates the expected XRES and compares it with the one received from the device. If they match, the device is authenticated.

Data Encryption.

The key K_ASME (Access Security Management Entity Key) is generated from CK and IK and is used for further generation of access-level protection keys.

The key K_eNB (eNodeB Key) is generated from K_ASME and is used to protect the data transmitted between the device and the base station.

The key KRRC (Radio Resource Control Key) is used for encrypting signaling at the RRC level.

The key KUPenc (User Plane Encryption Key) is used for encrypting user data.

Keys can be updated periodically or at the request of the network operator to ensure additional security. Key updates can be performed through the re-authentication process, which eliminates the need to regenerate RAND and compute new keys.

It can be said that the process of secure key management in low-power IoT networks is extremely important for protecting data and maintaining confidentiality. This includes regular key updates and device authentication, which helps prevent compromise.

5. Conclusions

Key management for encryption in wireless MESH networks is a critically important aspect of ensuring the security and reliability of network communications. The methods and recommendations discussed in the article emphasize the need for a comprehensive approach to key generation, exchange, storage, and recovery. The ability to remotely update a specific end device with confirmation is an important element of the network's flexibility and manageability. This allows for a prompt response to potential threats and ensures the relevance of cryptographic protection.

Group updates via broadcast to all end devices, with confirmation of the update, is an effective method for mass key updating. This ensures the synchronization of all devices in the network, enhancing overall security.

Establishing the lifetime of keys and their updates based on GPS synchronization allows for maintaining high accuracy and consistency of updates. Researching the transitional stages of security key updates helps avoid data loss and ensures the continuity of network operation.

The use of cryptographically secure algorithms, such as RSA, ECC, or DH, is the foundation of reliable encryption. Keys must be sufficiently long to prevent brute-force attacks, which increases the overall level of security.

Secure storage of keys on nodes is critically important to protect against physical attacks. Using hardware modules for key storage can significantly enhance the level of security.

This article examines various aspects of encryption key management in wireless MESH networks and provides recommendations for their use, emphasizing the importance of reliable and effective key management to ensure the security and reliability of network communications.

6. Gratitude

The authors express their gratitude to the staff of Department of Information and Measuring Technologies for help in their work.

7. Mutual claims of authors

The authors have no claims against each other.

References

[1] Hoa, L. V., & Vinh, T. Q. Real-time Key Management for Wireless Mesh Network. Journal of Telecommunication, Electronic and Computer Engineering (JTEC), Vol. 10 No. 2-6, 2018, pp.13–18. Retrieved from https://jtec.utem.edu.my/jtec/article/view/4362

[2] Gianni D'Angelo, Francesco Palmieri. A hound-inspired pre-hybridized genetic approach for router placement in wireless mesh networks. Applied Soft Computing. Vol 166, 2024, p.112159, doi.org/10.1016/j.asoc.2024.112159

[3] R. Imam, Q.M. Areeb, A. Alturki, F. Anwer Systematic and critical review of RSA based public key cryptographic schemes: past and present status. IEEE Access, 9, 2021, pp. 155949-155976, doi.org/10.1109/ACCESS.2021.3129224

[4] Abdmeziem, M. R., & Ahmed Nacer, A., & Deroues, N.M. Group key management in the Internet of Things: Handling asynchronicity. Future Generation Computer Systems, Volume 152, 2024, pp. 273-287, doi.org/10.1016/j.future.2023.10.023

[5] Bruno Ramos-Cruz, JavierAndreu-Perez, Luis Martínez. The cybersecurity mesh: A comprehensive survey of involved artificial intelligence methods, cryptographic protocols and challenges for future research. Neurocomputing. Volume 581, p.1274277, 2024, doi.org/10.1016/j.neucom.2024.127427

[6] Osama A. Khashan a, Rami Ahmad b, Nour M. Khafajah. An automated lightweight encryption scheme for secure and energy-efficient communication in wireless sensor networks. Ad Hoc Networks. Vol 115, 2021, p. 102448, doi.org/10.1016/j.adhoc.2021.102448

[7] Abdmeziem, M. R., & Ahmed Nacer, A., & Deroues, N.M (2024). Group key management in the Internet of Things: Handling asynchronicity. Future Generation Computer Systems, Volume 152, pp. 273-287, doi.org/10.1016/j.future.2023.10.023

[8] Xueping Yan, Lin Tan, Hong Xu, Wenfeng Qi. Improved mixture differential attacks on 6-round AES-like ciphers towards time and data complexities. Journal of

Information Security and Applications,Volume 80, 2024, p. 103661, doi.org/10.1016/j.jisa.2023.103661

[9] Miguel Antonio Caraveo-Cacep a b 1, Rubén Vázquez-Medina a 1, Antonio Hernández Zavala. A survey on low-cost development boards for applying cryptography in IoT systems. Internet of Things. Vol 22, 2023, p. 100743, doi.org/10.1016/j.iot.2023.100743.

[10] Antonio Muñoz, Ruben Ríos, Rodrigo Román, Javier López. A survey on the (in) security of trusted execution environments. Computers & Security. Vol 129, 2023, 103180, doi.org/10.1016/j.cose.2023.103180

[11] Kendall Niles a, Jason Ray a, Kenneth Niles a, Andrew Maxwell a, Anton Netchaev. Monitoring for Analytes through LoRa and LoRaWAN Technology. Procedia Computer Science. Vol 185, 2021, pp. 152-159, doi.org/10.1016/j.procs.2021.05.041

[12] Inc, Atecc608a, secure element to secure authentication. Accessed on 27.10.2022. www.microchip.com/en-us/product/ATECC608A

[13] Yasmeen Shaher Alslman1, Ashraf Ahmad1, Yousef AbuHour. Enhanced and authenticated cipher block chaining mode. Bulletin of Electrical Engineering and Informatics. Vol.12(4), 2023, pp. 2357-2362 doi:10.11591/beei.v12i4.5113

[14] R. Imam, Q.M. Areeb, A. Alturki, F. Anwer Systematic and critical review of RSA based public key cryptographic schemes: past and present status. IEEE Access, 9, 2021, pp. 155949-155976, doi.org/10.1109/ACCESS.2021.3129224

[15] Nil Llisterri Giménez, Joan Miquel Solé, Felix Freitag. Embedded federated learning over a LoRa mesh network. Pervasive and Mobile Computing. Vol 93, 2023, p.101819, doi.org/10.1016/j.pmcj.2023.101819

[16] Kun-Lin Tsai, Li-Woei Chen, Fang-Yie Leu, Chuan-Tian Wu. Two-Stage High-Efficiency Encryption Key Update Scheme for LoRaWAN Based IoT Environment. Computers, Materials and Continua. Vol 73, №1, 2022, pp. 547-562, doi.org/10.32604/cmc.2022.026557

[17] S. Anbazhagan, R.K. Mugelan. Next-gen resource optimization in NB-IoT networks: Harnessing soft actor–critic reinforcement learning. Computer Networks. Vol 252, 2024, p.110670, doi.org/10.1016/j.comnet.2024.110670