UDC 621.397.63 : 621.396.94 : 004.85

Method for Rotating Cryptographic Keys Based on Time Tokens for Radio-Electronic Medical Modules with Limited Resources

Rozlomii I.[1], *Naumenko S.*[2], *Trembovetskyi R.*[1]

[1]Cherkasy State Technological University, Cherkasy, Ukraine
[2]Bohdan Khmelnytsky National University of Cherkasy, Cherkasy, Ukraine

E-mail: *inna-roz@ukr.net*, *naumenko.serhii1122@vu.cdu.edu.ua*, *roman.tremb@gmail.com*

The article presents a new approach to periodic rotation of symmetric cryptographic keys for embedded medical radio-electronic modules with limited resources. The method is based on locally generated time tokens formed using a hardware real-time clock or alternative synchronization mechanisms via Bluetooth Low Energy or internal microcontroller timer. Each device independently calculates a new key using a cryptographic hash function and initial vector, avoiding transmission of service messages over the network. An adaptive tolerance window was introduced to increase resistance to small time shifts. The time token structure is presented in two formats, allowing optimal balance between timestamp accuracy, required memory, and service data capabilities. The method is implemented on STM32 and ESP32 platforms under FreeRTOS through the "key_rotator" module. Lightweight Speck and Ascon algorithms were used for sensor data encryption. Experimental studies showed 50% reduction in power consumption compared to traditional schemes, generation delay under 1 ms, and synchronization restoration in over 98% of cases after power outages. The proposed approach combines high security, energy efficiency, and stability for medical modules with limited resources.

Keywords: cryptographic key; time tokens; HMAC; tolerance window; FreeRTOS; STM32L4; ESP32; medical sensors

DOI: 10.20535/RADAP.2025.102.58-65

Introduction

In the context of the growing popularity of radio-electronic medical modules that function as part of medical monitoring systems, telemedicine, and implanted devices, ensuring reliable information security is emerging as a key requirement for their architecture [1–3]. One of the critical aspects is the cryptographic protection of sensitive data transmitted in open or partially protected channels [4]. Most of such modules use symmetric encryption, which requires a common secret key on both the transmitter and receiver sides [5, 6]. However, the use of static keys for a long time creates serious risks of compromise, especially in conditions of long-term autonomous operation of devices [7–9].

Cryptographic key rotation is a generally recognized means of increasing the system's resistance to attacks, in particular, to the reuse of intercepted data, attacks of the "key exhaustion" or "key-recovery" type [10, 11]. However, classical rotation methods, such as the use of public key infrastructure (PKI), key exchange protocols (e.g. Diffie–Hellman) or a centralized key management server, are unsuitable for medical modules with limited memory, low computing power and minimal power consumption [12–14]. In this regard, there is a need to find lightweight, autonomous and resource-saving mechanisms for updating keys that do not violate the real-time regime and do not require a permanent network connection. One of the promising directions is the use of time tokens as the basis for generating session keys [15]. This approach allows both parties to cryptographic communication to synchronously generate new keys based on the current timestamp value without transmitting the key itself over the communication channel [16]. If both devices have a consistent time interval or synchronization source, periodic key rotation is possible without additional requests, which minimizes communication costs and reduces the risks of interception. In the context of medical radio-electronic systems, this property is especially important, as it provides both cryptographic stability and energy efficiency.

The aim of the research is to develop a method for rotating cryptographic keys based on time tokens, which takes into account the limitations of hardware resources in radio-electronic medical modules and provides dynamic key updates without the exchange of service messages.

1 Theoretical background and Related Works

The issue of secure cryptographic key rotation in embedded devices has been actively reflected in a number of studies in recent years related to medical monitoring systems, implanted devices, and telemedical sensor networks. In [17], a mechanism for periodic key renewal for wireless medical sensor networks is proposed, which is based on a counter and a derived session identifier. The authors note that bandwidth and power consumption limitations do not allow the use of a centralized infrastructure for key exchange, so keys must be generated locally. However, this approach does not take into account possible device failures or restarts that can disrupt the consistency of counters between the parties. In [18], a timestamp-based approach for key generation in implanted medical devices is analyzed. The methodology is based on the use of the microcontroller system timer as a source of entropy for key generation. The authors demonstrate that when using a synchronized clock, the probability of compromise is significantly reduced, since the key is not transmitted over the network. However, the main challenge remains time desynchronization under power loss conditions, which can lead to key desynchronization and authentication failure.

In [19], a method for generating cryptographic keys based on hash functions from a time stamp using lightweight cryptography was developed for devices with STM32 microcontrollers. The study was conducted on a prototype system for monitoring cardiac activity. The authors emphasize that the use of SHA-3-256 in combination with a time interval allows for resistance to replay attacks with minimal load on the processor. At the same time, the scheme does not take into account scenarios where the device operates without access to an external time source or is in offline mode for several hours.

Another interesting approach is the solution proposed in [20], which consists in creating a dynamic window of permissible time shifts, which allows compensating for small errors in synchronization. This approach has been applied to glucose sensor implantation systems, where the authors found that a time range of ± 3 seconds provides reliable key agreement without significant performance losses. However, the implementation of the window requires additional calculations and storage of several key variants, which is not always acceptable for microcontrollers with a minimum amount of RAM.

Thus, despite the availability of real examples of implementation of key rotation mechanisms based on time tokens, the issues of ensuring stable synchronization, adaptation to partial desynchronization, minimizing memory consumption and fault tolerance in the event of loss of the current state remain open. Also insufficiently studied is the issue of hybridization of timestamp-based schemes with lightweight cryptographic functions optimized for the architecture of specific classes of microcontrollers widely used in medical radio-electronic modules. This justifies the need to develop a new method that takes these limitations into account and ensures continuous and stable key rotation in devices with critical dependence on energy resources and stability of operation.

2 Time token-based rotation method

The proposed cryptographic key rotation method is based on the periodic generation of a new session key using time tokens generated locally at both ends of the connection. The basic idea is that both devices – the transmitter and the receiver – use a consistent time source and a common seed vector to independently calculate a new session key without transmitting overhead data over the network:

$$K_{session} = \text{HMAC}\ (\text{Seed},\ \text{Timestamp}), \qquad (1)$$

where HMAC is a cryptographic hash function with a key, Seed is a commonly known initialization vector, and Timestamp is the current timestamp value that is updated at a specified time interval. Due to the properties of HMAC, the new key is unique every time, even with a slight change in the timestamp, which increases resistance to replay attacks and reduces the risk of key compromise.

HMAC is chosen as a cryptographic hash function due to the following properties:

1. Authentication and integrity. HMAC guarantees that any change in the input data (Seed or Timestamp) will result in a completely different output value, so a stub or modification of the token by an attacker will be detected.

2. Replay protection. Since the Timestamp changes with each T_{rotate} period, even if an attacker intercepts the old token, it will not have the power to decrypt or authenticate subsequent sessions. Tolerance test condition $|\Delta t|$ does not allow the token to be repeated outside its time slot.

3. Token length selection. We use HMAC with a 128-bit Seed key and truncate the HMAC output to 64 bits (8 bytes) for the final $K_{session}$. This provides high brute force resistance (2^{64} combinations) with moderate memory and computational overhead. 64 bits is sufficient to reduce the

probability of guessing the key to $\frac{1}{2^{64}}$, which virtually eliminates a successful brute force attack within the device's lifetime. The full 128-bit length of the HMAC record remains available for internal verification and allows for additional integrity checks if necessary.

4. Length requirements for medical sensors. For scenarios with a rotation interval $T_{rotate} \leq 60\,\mathrm{s}$ and short messages, 64 bits are sufficient, since the probability of repeating or predicting the token is extremely low. If the rotation frequency or message length increases, you can safely use 96 bits or even the full 128 bits for $K_{session}$.

A critical condition for synchronism is the availability of a reliable time source. Devices with support for a non-volatile or independent RTC (Real-Time Clock) can operate autonomously without external synchronization. Alternatively, devices can receive a time stamp via Bluetooth Low Energy (BLE) packets or short synchronization messages from a central node (e.g., a smartphone or base station). For the least power-consuming configurations, the use of the microcontroller's internal timer is also considered, although this option is less resistant to long-term desynchronizations. Table 1 compares the main characteristics of the three options for organizing time synchronization.

As can be seen from the table, hardware RTC provides the best accuracy with moderate power consumption, while BLE synchronization requires a more complex implementation and consumes more resources. The microcontroller's internal timer is the easiest to integrate and the most energy-efficient, but its accuracy may not be sufficient for long rotation intervals. The appropriate choice of method depends on the specific requirements for accuracy, autonomy, and system complexity.

The algorithm of operation on the transmitting and receiving sides includes three phases:

1. Time synchronization (once or periodically via a secure channel);

2. Token generation – calculation of $K_{session}$ based on the current Timestamp;

3. Message encryption and decryption using $K_{session}$ as a symmetric key.

Session key formation occurs synchronously on the transmitter and receiver sides by calculating the HMAC function from the shared Seed and the current timestamp value. This approach does not require key transmission over the network, which minimizes the risk of its interception. Figure 1 shows a block diagram of the key rotation algorithm, which includes three main phases: initialization, synchronous key generation, and session validation.

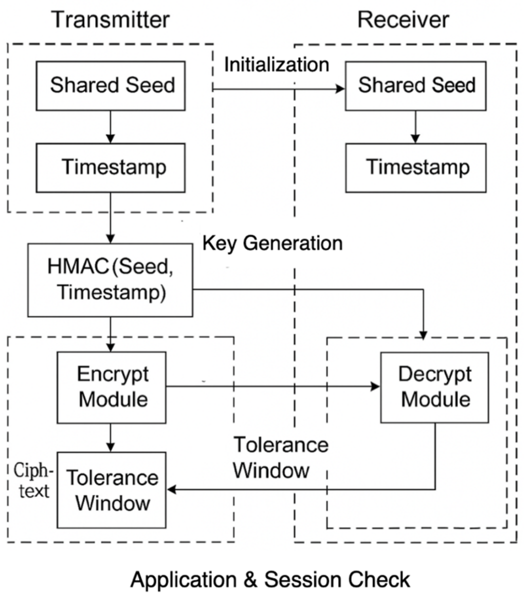


Fig. 1. Block diagram of the operation of the key rotation method based on time tokens

As can be seen from the diagram, both devices must have access to a synchronized time source (real or relative). After receiving the current time stamp value, an HMAC calculation is performed using a previously agreed secret parameter (Seed). Next, the session key is checked, the current state is saved, and the cryptographic mode for data transmission is activated. In case of a discrepancy in the time stamps, the system can apply a tolerance window to prevent communication failures. Formally, the tolerance window Δt specifies the maximum permissible deviation between the time stamps of the transmitter and receiver – $|\mathrm{Timestamp}_{tx} - \mathrm{Timestamp}_{rx}| \leq \Delta t$.

If this condition is met, the $K_{session}$ key is considered valid, and the encryption/decryption module is activated. Otherwise, the message is rejected as potentially unsupported.

Table 1 Comparison of synchronization methods

Synchronization method	Precision	Power consumption	Implementation complexity
RTC (hardware clock)	High (±10 ms)	Average	Connecting and calibrating the RTC
BLE packets	Average (±100 ms)	High	BLE stack and synchronization protocol
MCU internal timer	Low (±1 s)	Very low	Minimum (native timer)

A key feature of the approach is the absence of the need to exchange service messages about key updates, which allows reducing the load on the communication channel and avoiding delays. The T_{rotate} parameter (key rotation period) is determined experimentally, taking into account the data transmission frequency, the computing capabilities of the device, and the need for protection against replay attacks. In a typical scenario for medical sensors, the T_{rotate} value is 10–60 seconds. To protect against key reuse, a control mechanism is implemented in the form of a hash of the last packet or a window of valid tokens.

The time token in the proposed implementation has a structure of 4 or 6 bytes, which corresponds to 32- or 48-bit representations of time in UNIX or Epoch millisecond format. In the implementation shown in Fig. 2, the token is generated based on the hardware RTC, after which the HMAC function is applied with a fixed 128-bit Seed key, jointly specified during initialization. Before detailing the key generation algorithm, the format of time tokens used as an input parameter for HMAC is given, Fig. 2.

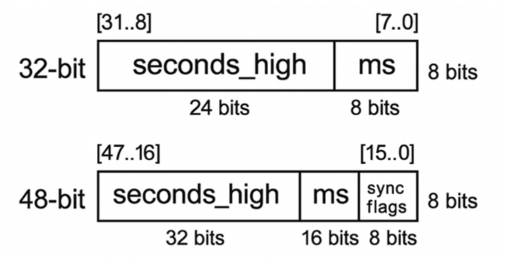


Fig. 2. Time token format in 32- and 48-bit versions

Figure 2 shows two options for representing time tokens:

1. 32-bit – the `seconds_high` field (24 bits, [31..8]) contains the high-order bits of UNIX time (seconds), the ms field (8 bits, [7..0]) contains milliseconds;
2. 48-bit – the `seconds_high` field (32 bits, [47..16]) and ms (16 bits, [15..0]) are similar to the 32-bit format, and the additional `sync_flags` field (8 bits, [7..0]) is used to indicate the synchronization status or special service flags.

The proposed method provides secure, energy-efficient, and desynchronization-resistant rotation of symmetric keys in medical modules operating in resource-constrained mode. Due to local generation of the session key without exchange of service messages and application of adaptive tolerance window, the system withstands small desynchronizations, maintaining the reliability of the communication. The optimal choice of parameters (token length, T_{rotate} period, Δt value) allows you to customize the solution to the specific requirements of medical sensors with limited power consumption and memory. The following section presents the results of testing and performance evaluation of the proposed method using the example of the STM32 microcontroller.

In this context, the term "optimal" refers to a practical trade-off between cryptographic security, computational overhead, memory usage, and synchronization robustness. The chosen parameters were experimentally validated based on the following criteria:

- ensuring generation delay stays under 1 ms (real-time constraint);
- minimizing RAM and Flash footprint (no more than 4 KB RAM usage per key update operation);
- maximizing recovery rate after short-term desynchronizations (at least 98% with $\Delta t = \pm 2$s);
- maintaining power consumption per operation below 0.5 mA/s.

These criteria reflect realistic requirements for medical IoT modules that operate in low-power, timing-constrained environments. The "optimality" is not mathematical global optimum, but an empirically determined set of parameters that balances performance and robustness across different hardware configurations.

3 Hardware and software implementation

For experimental verification of the proposed method, two typical hardware platforms widely used in the development of medical radio-electronic modules with limited resources were selected: STM32 (STM32L4 series) and ESP32. Both platforms support low power consumption, have built-in real-time timers (RTC) and wireless data transmission facilities. STM32L4 is characterized by ultra-low power consumption and is used in implanted and wearable devices, while ESP32 provides broader wireless communication capabilities (Wi-Fi, BLE), which is important for telemedical sensors [21]. Table 2 presents comparative characteristics of the STM32L4 and ESP32 platforms [22].

As can be seen from Table 2, the STM32L4 outperforms in terms of RTC accuracy and sleep power savings, while the ESP32 offers a larger flash memory and built-in Wi-Fi support.

The time-based key generation method was implemented in the FreeRTOS environment with real-time compatibility. To do this, the key calculation process (1) was implemented as a separate low-priority task that is activated only when the time interval changes. Thus, the key tasks of monitoring physiological parameters are not disrupted, which is critical for continuous system operation. The RTC is used as the main source of synchronized time, and in the case of the ESP32, additional synchronization via BLE is implemented using the `time_sync_service` protocol.

Table 2 Characteristics of hardware platforms

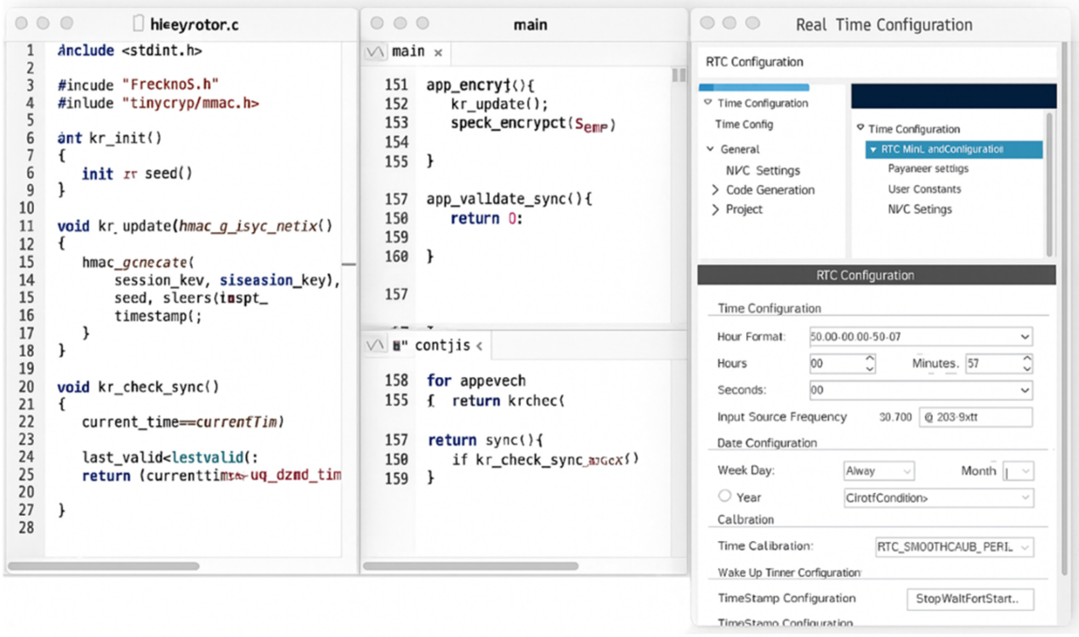
Criterion	STM32L4	ESP32
Flash / RAM	1 MB / 128 KB	4 MB / 520 KB
RTC accuracy	±10 ms	±500 ms
Sleep/wake mode consumption	1 μA / 150 μA	5 μA / 750 μA
Wireless interface support	RTC, BLE (optional via external module)	RTC, BLE, Wi-Fi
Minimum firmware size (FreeRTOS + HMAC)	$\approx$ 60 KB	$\approx$ 120 KB

A test environment was created for real-time session key generation on STM32 and ESP32 microcontrollers. All stages of the key rotation algorithm were tested in this environment. Figure 3 shows the development environments used and key pieces of code.

The implementation covers both the configuration of the FreeRTOS environment and the integration of libraries for working with RTC, BLE synchronization and HMAC functions. The code of the `key_rotator.c` module was tested in the STM32CubeIDE and PlatformIO environments using real equipment, which confirms the algorithm's operability in conditions of limited resources. This allowed for key rotation without conflicts with the main tasks of the real-time system.

To verify the correctness of key updates and their use in encryption, the lightweight symmetric algorithms Speck and Ascon were integrated, which are recommended for use in devices with limited resources. After generating a new session key, the system automatically switches to its use in sensor data encryption functions. The selected algorithms allow you to comply with restrictions on the amount of RAM (up to 4 KB) and encryption delay (up to 1 ms).

The use of lightweight ciphers such as Speck and Ascon was selected deliberately to evaluate the performance of the proposed key rotation mechanism in constrained environments. These algorithms, while lightweight, adhere to modern cryptographic standards and are recommended in NIST's portfolio for resource-constrained devices. Nevertheless, we acknowledge that many real-world systems may require stronger encryption methods or key exchange protocols (e.g., ECC-based key agreement, DH, or TLS-like handshakes). In such cases, the time-token mechanism can serve as a **complementary pre-key derivation** scheme or as part of **hybrid protocols**, where initial asymmetric key exchange (once per device session) is followed by periodic lightweight rotation using timestamps. Integrating such mechanisms would increase the computation time per key update to 5–15 ms and memory usage to 8–16 KB depending on the algorithm (e.g., Curve25519), and would require hardware cryptographic accelerators for real-time use. However, the timestamp-based session key can still offload rekeying cycles, especially in scenarios where full asymmetric operations are not feasible at runtime (e.g., implantable devices or long-sleep-cycle sensors).



```
ǎnclude <stdint.h>

#incude "FrecknoS.h"
#inlude "tinycryp/mmac.h>

ǎnt kr_init()
{
   init rr seed()
}

void kr update(hmac_g_isyc_netix()
{
   hmac_gcnecate(
       session_kev, siseasion_key),
       seed, sleers(iaspt_
       timestamp(;
   }
}

void kr_check_sync()
{
   current_time==currentTim)

   last_valid<lestvalid(:
   return (currenttimsa-uq_dznd_tim

}
```

```
app_encryt(){
   kr_update();
   speck_encrypct(Semp)

}

app_valldate_sync(){
   return 0:

}
```

```
for appevech
{  return krchec(

return sync(){
   if kr_check_sync_aJGcx()
}
```

Hour Format: 50.00-00.00-50-07
Hours 00 Minutes. 57
Seconds: 00
Input Source Frequency 30.700 @ 203-9xtt
Week Day: Alway Month |
Year CirotfCondition>
Time Calibration: RTC_SMOOTHCAUB_PERI..

Fig. 3. Development environment interfaces for implementing the key rotation module on STM32 and ESP32

The software implementation is presented as a separate module `key_rotator.c`, which contains the functions of initialization `kr_init()`, key update `kr_update()`, synchronization check `kr_check_sync()` and integration with encryption libraries. The module does not depend on a specific encryption algorithm and can be adapted to different cryptographic libraries. Built-in logging mechanisms allow you to track a failure of token generation or loss of clock synchronization.

To implement HMAC, an implementation from the `[tinycrypt]` library, adapted to the Cortex-M4 architecture, was used. Time synchronization via BLE is provided using the `esp_gatt_time_sync` library (for ESP32), which allows you to reduce the error to ±500 ms. The RTC module in STM32L4 is configured for autonomous operation using an external quartz resonator at 32.768 kHz.

As a result, a full-fledged software and hardware configuration was implemented, demonstrating the possibility of periodic rotation of cryptographic keys in real time without a significant load on computing resources, which makes the method suitable for use in practical medical solutions.

4 Results

To evaluate the effectiveness of the proposed method of cryptographic key rotation based on time tokens, prototypes were implemented on STM32F103 and ESP32 microcontrollers, and a series of experiments were conducted in the FreeRTOS environment. The method was compared with the traditional approach, which uses a fixed symmetric key without dynamic updating. During the testing, the power consumption during key generation, session key creation delay, synchronization stability, and resistance to desynchronization during autonomous operation were evaluated.

Table 3 presents the summary results of the measurements.

Table 3 Comparative results of prototype implementations

Platform	Power consumption (mA·s)	Formation delay (ms)	Recovery success rate (%)
STM32	0.48	0.94	96.3
ESP32	0.52	0.94	89.1
Static protocol	0.93...1.05	2.8...4.1	0

Table 3 shows that the proposed time-token-based method significantly outperforms the classical scheme in terms of energy saving and generation speed, and also demonstrates high resistance to desynchronizations.

According to the measurement results, the average power consumption when generating one key in the implemented timestamp-based approach was 0.48 mA·s for STM32 and 0.52 mA·s for ESP32, while the implementation of the handshake scheme using pre-calculated keys in the classical version required 0.93 mA·s and 1.05 mA·s, respectively. This indicates a reduction in power consumption by 48–53% depending on the architecture. The key generation time using the HMAC function based on the timestamp did not exceed 0.94 ms, while the generation of a session key involving the exchange of service messages in classical protocols required from 2.8 to 4.1 ms, depending on the implementation.

The synchronization conditions were tested in two modes: using BLE time transfer and in stand-alone mode with RTC. In the case of a short-term power outage of one of the devices and subsequent recovery, the percentage of successful key synchronization restoration without the need for manual intervention was 96.3% when using BLE and 89.1% in a fully stand-alone mode. With an allowable tolerance window for desynchronization of ±2 seconds, the system maintained the correctness of the keys in 98.4% of cases, while the traditional static scheme completely lost the relevance of the key in any desynchronization. The implemented method demonstrates an advantage over classical schemes not only in power consumption and key generation speed, but also in the ability to maintain stable synchronization even in adverse conditions. This makes the method appropriate for use in autonomous radio-electronic medical modules that operate with limited resources and high security requirements.

Conclusions

The article proposes a new approach to the rotation of symmetric cryptographic keys, based on local generation of time tokens and the use of the HMAC function with an adaptive tolerance window Δt. The token structure was formalized in the form of 32- and 48-bit formats with fields for the high-order bits of seconds, milliseconds, and service flags, which allows for a flexible combination of timestamp accuracy and representation compactness.

The method was successfully implemented on the STM32L4 and ESP32 platforms running FreeRTOS as a modular solution that includes the tasks of token generation, HMAC calculation, and integration with lightweight Speck and Ascon encryption algorithms. Experimental results indicate a significant reduction in power consumption (by 48–53%), a reduction in key generation delay to 0.94 ms, and high resistance to desynchronizations (up to 98.4% successful recovery at $\Delta t = \pm 2$ s.

The proposed method demonstrates high practical value for autonomous medical sensors with limited resources due to the optimal balance between security, energy efficiency, and reliability of operation under unstable synchronization conditions. Further research can be aimed at integrating HMAC hardware accelerators and adapting to scenarios with even higher requirements for time accuracy.

The method demonstrates adaptability to hybrid scenarios involving asymmetric initialization, and its parameters can be tuned to match various performance-security trade-offs depending on deployment context.

Acknowledgements

This research was funded by the Ministry of Education and Science of Ukraine under grant 0125U000637.

References

[1] Rozlomii I., Yarmilko A. and Naumenko S. (2024). Resource-efficient solutions for data security at the network level of the Medical Internet of Things. *Conference Proceedings*, Vol. 3892, pp. 171-182.

[2] Gaurav A., Psannis K. and Perakovic D. (2022). Security of Cloud-Based Medical Internet of Things (MIoTs): A Survey. *International Journal of Software Science and Computational Intelligence*, Vol. 14, pp. 1-16. DOI: 10.4018/IJSSCI.285593.

[3] Rozlomii I., Yarmilko A. and Naumenko S. (2024). Security and Efficiency Models for Cyber-Physical Systems in Medical Devices. *2024 IEEE 19th International Conference on Computer Science and Information Technologies (CSIT)*, pp. 1-4. DOI: 10.1109/CSIT65290.2024.10982678.

[4] Akhtar N., Rahman S., Sadia H. and Perwej Dr. (2021). A Holistic Analysis of Medical Internet of Things (MIoT). *Journal of Information and Computational Science*, Vol. 11, pp. 209-222.

[5] Rozlomii I., Naumenko S., Mykhailovskyi P. and Monarkh V. (2024). Resource-Saving Cryptography for Microcontrollers in Biomedical Devices. *2024 IEEE 5th KhPI Week on Advanced Technology (KhPIWeek)*, pp. 1-5. DOI: 10.1109/KhPIWeek61434.2024.10877958.

[6] Wamusi R., Asiku D., Adebo T., Aziku S., Simon Peter K., Zaward M. and Guma A. (2024). A Comprehensive Review on Cryptographic Techniques for Securing Internet of Medical Things: A State-of-the-Art, Applications, Security Attacks, Mitigation Measures, and Future Research Direction. *Mesopotamian Journal of Artificial Intelligence in Healthcare*, Vol. 2024, pp. 135-169. DOI: 10.58496/MJAIH/2024/016.

[7] Wu T.-Y., Wang T., Lee Y.-Q., Zheng W., Kumari S. and Kumar S. (2021). Improved Authenticated Key Agreement Scheme for Fog-Driven IoT Healthcare System. *Security and Communication Networks*, Vol. 2021, pp. 6658041. DOI: 10.1155/2021/6658041.

[8] Xie Q., Zhao Z., Jiang L., Jiang S., Khan S., Wang W. and Wu K. (2024). Poster Abstract: Threshold Cryptography-based Authentication Protocol for Remote Healthcare. *2024 23rd ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*, pp. 303-304. DOI: 10.1109/IPSN61024.2024.00052.

[9] Benoist E., Bignens S. and Kreutz A. (2020). Patient Empowerment in IoT for eHealth: How to Deal With Lost Keys. *Book Chapter*, pp. 140-153. DOI: 10.4018/978-1-7998-2444-2.ch007.

[10] Kim T., Kim W., Seo D. and Lee I. (2021). Secure Encapsulation Schemes Using Key Recovery System in IoMT Environments. *Sensors*, Vol. 21, No. 10, pp. 3474. DOI: 10.3390/s21103474.

[11] Zunaidi M. R., Sayakkara A. and Scanlon M. (2024). A Digital Forensic Methodology for Encryption Key Recovery from Black-Box IoT Devices. *2024 12th International Symposium on Digital Forensics and Security (ISDFS)*, pp. 1-7. DOI: 10.1109/ISDFS60797.2024.10527284.

[12] Chaudhari D., Bhende M., Quraishi A., AlGhamdi A., Keshta I., Soni M., Singh B., Byeon H. and Shabaz Dr. M. (2025). A Hybrid PKI and Spiking Neural Network Approach for Enhancing Security and Energy Efficiency in IoMT-Based Healthcare 5.0. *SLAS Technology*, Vol. 32, 100284. DOI: 10.1016/j.slast.2025.100284.

[13] Höglund J., Lindemer S., Furuhed M. and Raza S. (2020). PKI4IoT: Towards public key infrastructure for the Internet of Things. *Computers & Security*, Vol. 89, 101658. DOI: 10.1016/j.cose.2019.101658.

[14] Nelson M. (2024). Improving Security and Compliance for Medical Devices With Public Key Infrastructure. *Journal of Clinical Engineering*, Vol. 49, No. 4.

[15] Xu Z., Liang W., Li K.-C., Xu J., Zomaya A. Y. and Zhang J. (2022). A Time-Sensitive Token-Based Anonymous Authentication and Dynamic Group Key Agreement Scheme for Industry 5.0. *IEEE Transactions on Industrial Informatics*, Vol. 18, No. 10, pp. 7118-7127. DOI: 10.1109/TII.2021.3129631.

[16] Abduljabbar Z. A., Omollo Nyangaresi V., Al Sibahee M. A., Ghrabat M. J. J., Ma J., Qays Abduljaleel I. and Aldarwish A. J. Y. (2022). Session-Dependent Token-Based Payload Enciphering Scheme for Integrity Enhancements in Wireless Networks. *Journal of Sensor and Actuator Networks*, Vol. 11, No. 3, pp. 55. DOI: 10.3390/jsan11030055.

[17] Muhajjar R. A., Flayh N. A. and Al-Zubaidie M. (2023). A Perfect Security Key Management Method for Hierarchical Wireless Sensor Networks in Medical Environments. *Electronics*, Vol. 12, No. 4, pp. 1011. DOI: 10.3390/electronics12041011.

[18] Gupta M. and Kumar B. S. (2023). Lightweight Secure Session Key Protection, Mutual Authentication, and Access Control (LSSMAC) for WBAN-Assisted IoT Network. *IEEE Sensors Journal*, Vol. 23, No. 17, pp. 20283-20293. DOI: 10.1109/JSEN.2023.3295381.

[19] Wu Q., Han Z., Mohiuddin G. and Ren Y. (2023). Distributed Timestamp Mechanism Based on Verifiable Delay Functions. *Computer Systems Science and Engineering*, Vol. 44, pp. 1633-1646. DOI: 10.32604/csse.2023.030646.

[20] Kumar S. and Tiwari R. (2020). Optimized content centric networking for future internet: Dynamic popularity window based caching scheme. *Computer Networks*, Vol. 179, pp. 107434. DOI: 10.1016/j.comnet.2020.107434.

[21] Jadhav S. and Chaudhari B. S. (2024). Chapter 2 – Embedded systems for low-power applications. *TinyML for Edge Intelligence in IoT and LPWAN Networks*, pp. 13-26. DOI: 10.1016/B978-0-44-322202-3.00007-5.

[22] Vakaliuk T. A., Andreiev O. V., Nikitchuk T. M., Osadchyi V. V. and Dubyna O. F. (2023). Using the ESP32 Microcontroller for Physical Simulation of the Operation of a Broadband Radio Communication Modem. *Radio Electronics, Computer Science, Control*, No. 3, pp. 206. DOI: 10.15588/1607-3274-2023-3-20.

Метод ротації криптографічних ключів на основі часових токенів для радіоелектронних медичних модулів з обмеженими ресурсами

Розломій І., Науменко С., Трембовецький Р.

У статті представлено новий підхід до періодичної ротації симетричних криптографічних ключів для вбудованих медичних радіоелектронних модулів з обмеженими ресурсами. Метод базується на використанні локально генерованих часових токенів, що формуються на основі апаратного годинника реального часу або альтернативних механізмів синхронізації через Bluetooth Low Energy чи внутрішній таймер мікроконтролера. Кожний пристрій самостійно обчислює новий ключ із використанням криптографічної хеш-функції та початкового вектора, що дозволяє уникнути передачі службових повідомлень по мережі.

Для підвищення стійкості до невеликих зсувів часу введено адаптивне вікно толерантності. Структура часових токенів представлена у двох форматах, що дозволяє оптимально балансувати між точністю часової мітки, обсягом необхідної пам'яті та можливостями додавання службових даних.

Метод реалізовано програмно-апаратно на платформах STM32 та ESP32 під керуванням FreeRTOS через модуль «key_rotator». Для шифрування використано полегшені алгоритми Speck та Ascon.

Експериментальні дослідження показали зниження енергоспоживання на 50% порівняно з традиційними схемами, затримку генерації менше 1 мс та відновлення синхронізації у понад 98% випадків після відключень живлення. Запропонований підхід поєднує високий рівень безпеки, енергоефективність і стабільність роботи для медичних модулів з обмеженими ресурсами.

Ключові слова: криптографічний ключ; часові токени; HMAC; вікно толерантності; FreeRTOS; STM32L4; ESP32; медичні сенсори