Enhancing Cybersecurity in IoT Networks: Exploring Lightweight Key Establishment Using OSCORE and EDHOC

RENETA TRIFONOVA, University of Twente, The Netherlands
SUPERVISOR: DR. ING. MOHAMMED ELHAJJ, University of Twente, The Netherlands

The widespread adoption of IoT networks has revolutionized modern life thanks to their diverse range of applications. With their increased usage, the need to transfer data securely has never been more vital. Often IoT devices are battery-driven and possess limited processing and storage capabilities. In response to those limitations, there has been a shift towards the implementation of resource-efficient and lighter protocols. This research aims to evaluate how lightweight key establishment through Object Security for Constrained RESTful Environments (OSCORE) and Ephemeral Diffie-Hellman Over COSE (EDHOC) can enhance cybersecurity in IoT networks where resources are limited. The study strives to explore different key management approaches and the effects of certificates on security, efficiency, and power consumption. The research methodology consists of the practical implementation of the protocols in a physical system where factors such as the impact of hardware acceleration on computational overhead and overall power consumption will be assessed. The expected contributions are insights into the tradeoffs between security, power consumption, and efficiency in resource-limited IoT environments.

Additional Key Words and Phrases: IoT, CoAP, OSCORE, EDHOC, Hardware Acceleration, TLS, DTLS

1 INTRODUCTION

The Internet of Things (IoT) refers to networks of connected devices that can communicate with each other or to the cloud [1]. Nowadays, these networks have become an essential part of our daily lives since we are always surrounded by smart products. As the number of IoT devices continues to grow, they become more vulnerable to attacks. Therefore, it is vital to secure the transmitted data to prevent malicious activity that could compromise the entire network and cause serious consequences. In order to ensure secure communication in IoT networks, protocols for end-to-end security have been implemented. These protocols ensure that the messages' confidentiality, integrity, and availability remain uncompromised.

1.1 Background

Constrained Application Protocol (CoAP) is a compressed version of the HTTP protocol designed for resource-constrained devices where compatibility and low power are essential according to its documentation [23]. While CoAP provides a high level of communications security, it remains vulnerable to a series of attacks, such as Man-In-The-Middle attacks or Denial-of-Service Attacks. To mitigate such security vulnerabilities, the Internet Engineering Task Force (IETF) has published a new standard, Object Security for Constrained RESTful Environments (OSCORE), for application-level security. According to the protocol documentation, OSCORE provides end-to-end protection between endpoints communicating using CoAP or CoAP-mappable HTTP [11]. Moreover, it supports a wide range of proxy operations as well as translation between transport protocols. As it was designed for resource-constrained devices, its messages are as small as 11-13 bytes, and the protocol encrypts only the data part of the payload, thus decreasing security overhead and increasing bandwidth usage and battery lifetime of the device [22]. To set up OSCORE, a security context needs to be established. In order to do it in an optimized way, the lightweight authenticated key exchange protocol Ephemeral Diffie-Hellman Over COSE (EDHOC) can be used. EDHOC was proposed by the IETF for its many advantages, one of which is that it significantly reduces the number of roundtrips needed for setting up the OSCORE security context [18]. The way the security context is set up can be seen in Figure 1.

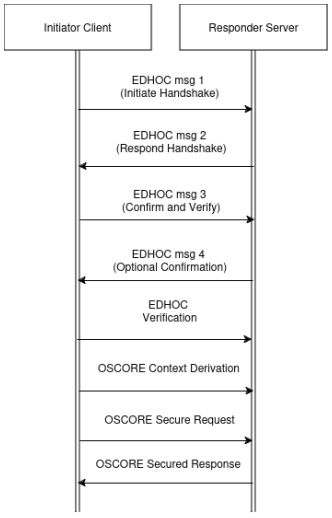

Fig. 1. OSCORE-EDHOC secure context establishment

One of the advantages EDHOC offers is that it uses ephemeral keys, meaning for each key establishment session, different keys are generated, ensuring there is Perfect Forward Secrecy (PFS) in the communication channel [3]. The overall performance and cost of OSCORE and EDHOC-based implementations have been of interest to many researchers as it is crucial to evaluate metrics such as computational overhead, power consumption, and security when it comes to IoT devices. While there have been tests conducted to measure such metrics, there is limited data about how different key management strategies and certificates can affect performance.

TScIT 41, July 5, 2024, Enschede, The Netherlands

Moreover, many articles mention that hardware acceleration can have a high impact on the performance and memory requirements of the protocols, however, there is no sufficient research made to extract concrete quantitative data to support the claim.

The IoT communication networks play a pivotal role in managing complex systems in various domains such as healthcare, smart homes, and industrial automation. As such devices have increased in complexity, new challenges associated with cybersecurity have emerged, especially in environments with limited resources. One of the largest IoT cyber attacks with severe consequences was the Mirai botnet distributed denial-of-service (DDoS) attack [17], where over 175,000 websites were affected. Such attacks which take advantage of weak passwords can be prevented by adopting strong authentication mechanisms. The combination of using OSCORE together with EDHOC ensures end-to-end encryption with verification of both parties before data is exchanged, preventing DDoS attacks. The need for robust authentication and encryption in IoT networks opts for careful reflection on the key management strategies (e.g, Pre-shared Keys (PSK), Raw Public Keys (RPK), or dynamically generated keys) as it is important to consider various metrics, such as latency, throughput, and security in the design of the system. However, at present, there is a lack of data that can help evaluate the suitability and trade-offs of these key management strategies, especially in the context of OSCORE and EDHOC-based systems. Moreover, such networks use certificate-based authentication, although a proper comparison of different digital certificates in terms of critical metrics like resource utilization has yet to be made. This is especially vital for devices with limited computational power, as certificate-based authentication can prove to be resource-heavy.

More ways to improve computational overhead include the use of hardware accelerators, special-purpose hardware structures separated from the CPU [19], as they can enhance the cryptographic efficiency of algorithms such as Advanced Encryption Standard (AES), an algorithm used with OSCORE. However, there is a need for detailed documentation on the tradeoffs between speed and power consumption when hardware acceleration is enabled, both crucial factors to consider for IoT networks.

1.2 Research Questions

Based on the above-mentioned information, the main goal of this work will be to answer the following question:
How do key management strategies, certificates, and hardware accelerators impact the latency, throughput, security, and energy efficiency of IoT devices secured by OSCORE and EDHOC protocols, and how does establishing the security context over DTLS compare?

The main question can be split into the following three subquestions:

RQ1 How do different EDHOC methods of operation impact the security of a system utilizing the OSCORE protocol and its performance metrics, such as latency and memory requirements in IoT networks?

RQ2 How do hardware accelerators influence the computational overhead, energy consumption, and memory usage of IoT devices implementing OSCORE and EDHOC protocols, and what are the associated tradeoffs?

RQ3 How does establishing a security context for OSCORE over DTLS compare in terms of computational overhead to incorporating the EDHOC protocol in the design?

1.3 Structure

The rest of this paper is organized as follows: Section 2 provides an overview of related works. Section 4 presents the obtained data from performing tests on computational overhead with the help of an experimental setup together with an analysis of the security of the system. The following Section 5 contains an analysis of the results, while the conclusions Section 6 summarises the key points discussed. Finally, in Section 7, directions for future work are presented.

2 RELATED WORK

As secure communication is essential within IoT networks, many studies were conducted on advanced lightweight encryption algorithms. In the article [24], various challenges to IoT environments, such as power consumption of devices, limited battery, memory space, performance cost, and security, were listed. The same article pointed out that a constrained device's security can be subject to flexibility, emphasizing the importance of security metrics. The standard protocol for application-layer protocol is Hypertext Transfer Protocol (HTTP). However, it has proven to be heavy and inefficient when implemented on constrained, battery-powered devices [16].

Consequently, a simpler alternative, namely CoAP, was introduced for IoT nodes. There is a variety of methods to achieve secure communication for CoAP, one of which is applying OSCORE. As evaluated in the article [7], compared to other alternatives, such as Datagram Transport Layer Security (DTLS), OSCORE offers many advantages, including the possibility to deploy on non-trusted proxies through object security. Before working with the OSCORE protocol, a secure context needs to be established. This can be done through EDHOC, a lightweight authenticated key exchange protocol. One of EDHOC's advantages over Transport Layer Security (TLS) is the reduced amount of roundtrips [2]. Moreover, the EDHOC handshake process requires only three messages with a fourth optional message, while through DTLS, it can consist of up to 13 [4].

Evidence from a recent study [4] indicates DTLS uses at least x7 times more time on air and x4 times more FLASH and RAM compared to EDHOC. After its standardization, OSCORE's performance, together with EDHOC, was compared to that of different protocols, demonstrating promising results. When OSCORE is used together with the key exchange protocol EDHOC, a times five reduction over DTLS has been demonstrated in terms of transmitted bytes [10]. While there are promising results, it is important thorough testing and valuation are done, as the protocols were introduced recently.

A study by Kim [13] presented a detailed analysis of potential vulnerabilities of the EDHOC protocols, including susceptibility to man-in-the-middle attacks during the key exchange phase. Their findings suggest it is crucial to evaluate security against emerging threats continuously. Another study [12] delved into various key management methods, exploring their effectiveness and cost for IoT devices. Some strategies mentioned were PSK, RPK, and CBOR Web Tokens(CWT). The derived conclusions stated that PSK is commonly

Fig. 2. Comparison between security protocols, based on "Fig 1" from [6]

used as it is efficient and easier to implement, while RPK and CBOR Tokens have a higher computational demand and may be suitable for more powerful devices as they offer more robust security protection. In some studies about OSCORE and EDHOC protocols, [10], pre-shared keys were used for authentication. However, based on the documentation on the protocol, there is no support for PSK as static DH keys are sufficient [21]. It was explained that in initial versions, PSK was an option; however, in later drafts, the decision was retracted [9].

Although there are other ways to perform key sharing, the differences between these methods regarding security and resource usage need further evaluation. In [9] and [7], it is mentioned that hardware acceleration increases the performance of OSCORE and EDHOC. Additionally, the research done on Crypto-Hardware in the Low-end IoT by Santos [20] documents that by utilizing hardware acceleration for cryptographic operations, the overall energy cost and execution time can be significantly reduced. Nevertheless, there is still limited information about the exact impact hardware acceleration can make in terms of latency and memory requirements in a system implementing the OSCORE and EDHOC protocols.

An alternative approach for establishing a secure communication channel between IoT devices is Named Data Networking (NDN). NDN offers confidentiality by using authentication-based names instead of IP addresses for identification [26]. Its high reliability can be brought down to hop-wise transfers and network caches, reducing the need for retransmissions [5]. Therefore, in multiple-hop scenarios, NDN outperforms CoAP over DTLS and OSCORE [5]. Nevertheless, an OSCORE-oriented approach has advantages over NDN as it introduces a smaller security message overhead [5]. For a better overview of the different approaches to secure IoT communications, the following schema can be referred to: Figure 2.

3 METHODOLOGY

In this section, the research methodology will be discussed, addressing the hardware setup, software implementation, and used libraries together with the research metrics that will help answer the research questions.

3.1 Hardware setup

The testing environment consisted of the following components. The experiments utilize the setup shown in Table 1. The way the components were connected can be seen in as shown in Figure 3:

Table 1. Devices for Hardware Setup

Device	Flash Memory	SRAM
2x Raspberry Pi 4B	32GB	4GB
ESP32-S	4MB	520KB
Thinkpad P15v	500GB	16GB
AVHzY USB-Meter C3	/	/

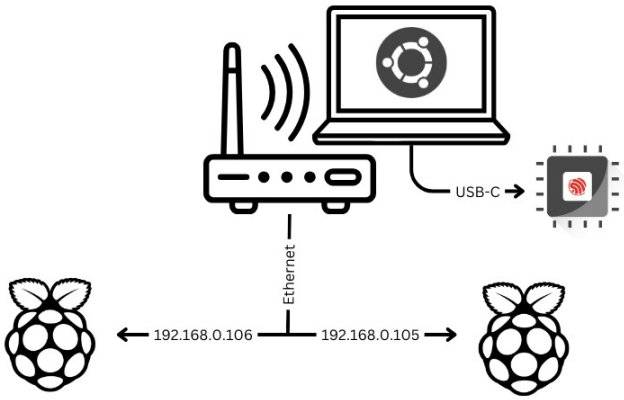

Fig. 3. Hardware Setup

The method used for measuring power consumption was taking the highest measured value throughout the program's execution. If there was a difference between multiple measurements, the average was presented.

3.2 Software setup

3.2.1 Measurements. The amount of flashed bytes was taken to measure overall memory usage on the ESP32 microcontroller. The memory measurements for the Raspberry Pi 4 were acquired through the *usr/bin/time* Linux command together with timing functions in the code for the relevant sections. Note the results depend on the libraries used and their submodules.

3.2.2 uOSCORE-uEDHOC library. The uOSCORE-uEDHOC library was installed on both Pis. The first Pi takes on the role of the initiator client, while the other one is a responder server. They use the library's corresponding sample implementations of both roles. Since the two Pis are connected via the same network, they were also assigned static IPs. There are four supported EDHOC methods: Signature keys for both initiator and responder, Static Diffie-Hellman keys for both initiator and responder and a combination of signature keys and Diffie-Hellman keys. Moreover, Zephyr OS was used to upload the project containing uOSCORE and uEDHOC protocol implementation on ESP32S so that the metrics could be evaluated for a microcontroller of a smaller size.

3.2.3 libcoap library. An additional library, namely libcoap, was used for its OSCORE implementation over DTLS. The library was installed on each Raspberry Pi 4 device, enabling them to communicate with each other via WiFi.

3.2.4 edhoc and libcoap libraries. Libcoap, together with a third library called edhoc, was utilized for evaluating the performance of both OSCORE and EDHOC protocols on ESP32S as well as documenting the impact hardware acceleration has on the overall efficiency of the protocols. For the purpose of performing such tests, the project configuration options for hardware acceleration, namely "Use hardware AES acceleration", "Use hardware SHA acceleration", "Use hardware MPI (bignum) acceleration" were used. Currently, the edhoc library only supports EDHOC method 0 (EDHOC authenticated with asymmetric keys) and cipher suite 0 (AES-CCM-16-64-128, HMAC 256/256, X25519, EdDSA, Ed25519). It is important to mention that the edhoc library is only compatible with an older version of CoAP.

3.3 Research Metrics

A variety of metrics relevant to the computational overhead of resource-constrained devices were measured. As securing IoT communications is a pivotal goal in this research, metrics relating to the security of the implementation were evaluated. Below there is a list of relevant measurements related to the EDHOC and OSCORE protocols:

(1) Power Consumption: The electric current (Amperes) and electric power (Watts) consumed by the devices [24].
(2) Memory usage: the Flash or RAM requirements that need to be satisfied to use the protocols measured in bytes [7].
(3) Computation Time: The time that certain processes take to execute [8]; in this study, it is measured in milliseconds or seconds, depending on the experiment.
(4) Confidentiality: Ensuring no unauthorized individuals have access to the transmitted data [25].
(5) Integrity: The data has not been altered by an unauthorized individual [25].
(6) Authentication: The process of confirming the identity of a device or entity in the system [25].

4 RESULTS

The setup mentioned in Figure 3.1 was used to evaluate the performance and security implications of using OSCORE and EDHOC protocols in resource-constrained IoT environments. The results are organized according to the research questions, providing insights into key management strategies, hardware accelerators' impact, and DTLS comparisons.

4.1 RQ1

The following section will summarise the findings related to answering the first Research Question 1.2. The first research question focuses on comparing the use of Diffie-Hellman Static Keys and Signature Keys as well as CBOR certificate authentication. The ESP32S device was used to obtain data related to the computational overhead of the authentication mechanisms on resource-constrained devices. It is important to mention the results can be dependent on the cryptography engine used.

4.1.1 Power Consumption: Between the three key management methods, there was no visible difference in terms of electric current and power; the measured values were consistently 0.025A and 0.129W. The power consumption was calculated based on the below Equation 1:

$$P = V \times I \tag{1}$$

where:

- P is the power in watts (W).
- V is the voltage in volts (V).
- I is the current in amperes (A).

4.1.2 Computation Time. After comparing the results from tests on the interaction between OSCORE and EDHOC with six different test vectors, two of which used Signature Keys, two which used Diffie-Hellman Static Keys, and two utilized CBOR certificates, the results showed a slight increase in latency when the Diffie-Hellman Static Keys were used. This can be explained by the fact that the Diffie-Hellman key exchange involves additional steps for generating and exchanging the keys in a secure way. In Table 2, the differences can be compared.

4.1.3 Memory Usage. In Table 2, it can be seen the three methods have minimal differences when it comes to the usage of DRAM (Dynamic Random Access Memory) and IRAM (Instruction Random Access Memory). However, the FLASH space used is much higher, which could be explained since the certificate size is larger than the key size as it can contain data such as issuer information, date of validity, subject information, signature, etc. Moreover, parsing and storing the certificates is an additional requirement.

4.1.4 Confidentiality.

- Diffie-Hellman Static Keys: As DH keys provide a strong encryption mechanism that establishes communication over an insecure channel, even if attackers intercept a message, it will not be possible to decrypt the data. As the EDHOC suite uses the AES-CCM-16-64-128 algorithm, confidentiality is protected while keeping the protocols efficient in a constrained environment.
- Signature keys offer strong encryption, although they lack forward secrecy - the keys used to encrypt and decrypt information do not change, making the system vulnerable to retrospective attacks.
- Using PKI (Public Key Infrastructure), CBOR encoded X.509 certificates provide a strong level of confidentiality.

4.1.5 Integrity. Integrity in the system is maintained through the hashing algorithm SHA-256. Both DH-based key exchange and signature keys prevent the data from being tampered with by ensuring no modifications are made to the public keys used.

4.1.6 Authentication. The Diffie-Hellman key exchange poses a vulnerability of a man-in-the-middle attack. By using signature keys, authentication can be performed through digital signatures, confirming the identities of the parties trying to connect.

Table 2. Authentication Methods and measurements

Type	EDHOC Method	Latency (sec)	IRAM (bytes)	DRAM (bytes)	FLASH (bytes)
Client request	Signature Keys	0.154	46656	8756	147456
Client request	Static DH Keys	0.164	46656	8756	147456
Server key derivation	Signature Keys	0.038	46656	8752	147456
Server key derivation	Static DH Keys	0.044	46656	8752	147456
Client authentication	CBOR certificate	2.827	46656	8808	229376
Server authentication	CBOR certificate	2.791	46656	8808	229376

These results indicate that the communication protocols exhibit low latency, which is crucial for time-sensitive IoT applications. The minimal time required for message transmission and acknowledgment highlights the efficiency of the OSCORE and EDHOC protocols in constrained environments.

4.2 RQ2

For the purpose of answering the second research question, the setup involving a laptop and microcontroller ESP32S was used. The microcontroller has support for the following types of hardware acceleration: AES, Random Number Generation (RNG) and SHA256. Previous studies have mentioned the possibility of improvement when such hardware acceleration is involved in the process; more information in Section 2. The performance comparison between enabling and disabling hardware acceleration for the ESP32S in implementing the OSCORE and EDHOC protocols reveals minimal differences in performance, as can be seen from Figure 4. This observation can be attributed to several factors:

- Resource-Efficient Protocol Design: The EDHOC protocol is specifically designed for minimal computational and message overhead, owing to the fact that many resource-constrained devices do not have the capacity to perform intense operations. The protocol employs elliptic curve cryptography, which requires less computational effort and is thus efficient even without hardware acceleration.
- Optimized Library Implementation: The EDHOC library, edhoc,, available at Marco von Raumer's GitLab repository, utilizes the `mbedtls` library for cryptographic operations. This library is optimized for performance, leveraging efficient algorithms and data structures to minimize computational load. The use of `mbedtls_ecdh_calc_secret` together with other optimized functions ensures that the performance of resource-constrained devices remains high even without them having dedicated hardware acceleration.
- Minimal Message Overhead: The protocol's design minimizes the number of messages exchanged and their sizes, which reduces the time and computational resources required for each operation.

4.2.1 Computation time:

- EDHOC Initialization (`edhoc_init_context`): Both configurations took nearly the same time (28 ms).
- Handling CoAP Message (`edhoc_handle_coap_message`): The average time with hardware acceleration enabled was slightly higher (98.33 ms) compared to when disabled (94.67 ms); however, such difference is negligible.
- OSCORE Context Creation (`create_oscore_ctx`): With hardware acceleration enabled, the average time was 54.75 ms, slightly more than the 52.8 ms without it.

4.2.2 Memory Usage: Memory usage metrics can be observed in the following Table 4. There was consistency across both configurations for all operations with slight differences. The version where hardware acceleration was enabled consistently used less flash memory: 503872 bytes in comparison to 508758 for the server and 508707 compared to 513425. The difference is approximately 1%, which in such a system does not make a large impact.

4.2.3 Power Consumption: Power measurements did not show a noticeable difference between the implementation with hardware acceleration and without it.

Table 3. Flash Memory used in bytes with and without Hardware Acceleration (HWA)

Mode	Flash Memory	Flash Memory (Compressed)
Server HWA	785536	503872
Server no HWA	790608	508758
Client HWA	792624	508707
Client no HWA	797760	513425

4.3 RQ3

To compare the security context establishment over DTLS and EDHOC, both the libcoap and uOSCORE-uEDHOC libraries were used. Currently, the libcoap library does not support the EDHOC protocol, and it uses DTLS to establish the security context, while the uOSCORE-uEDHOC library has implemented the EDHOC protocol. The two libraries' implementations were compared through a scenario where the following actions happened: encrypting a request, sending it, receiving a response, decrypting it, and validating the results. The setup used for the tests included the two Raspberry Pi 4 devices, which were connected via WiFi; more information can be found in the methodology Subsection 3.1.

Table 4. Overview of performance comparison between OSCORE and EDHOC and OSCORE and DTLS for security context establishment

Protocol(s)	Prower (W)	Current (A)	Time (s)	RAM (KB)
OSCORE and EDHOC	1.536	0.301	0.046	2732
OSCORE and DTLS	2.235	0.395	2.04	7049

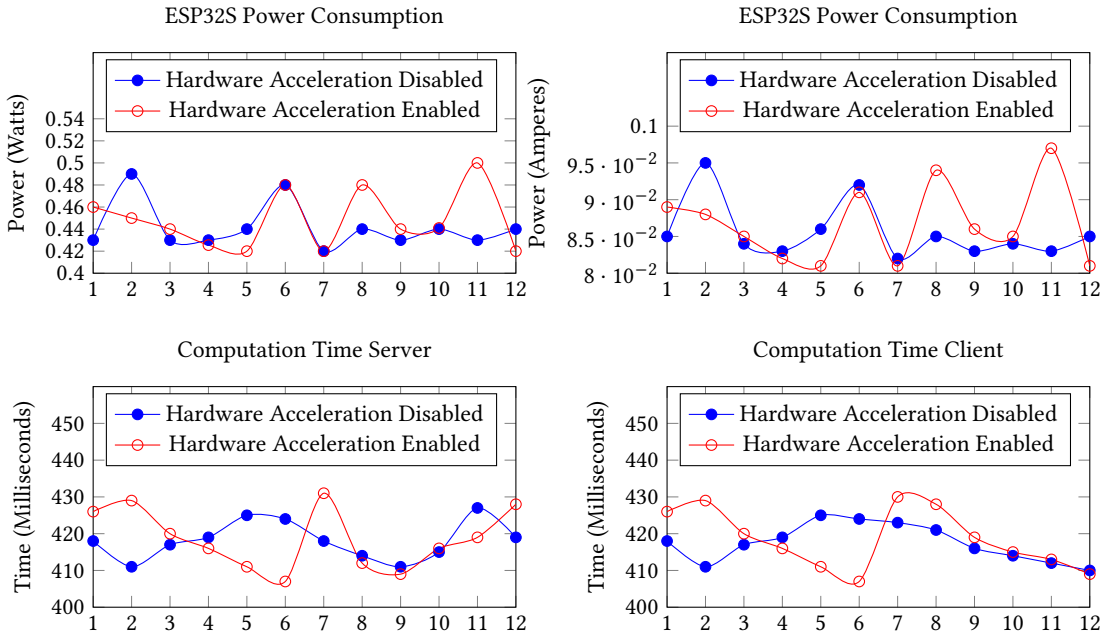

Fig. 4. ESP32S Power Consumption and Total Computation Time with and without Hardware Acceleration

4.3.1 Power Consumption: The maximum power consumed by the libcoap client side was 0.395 Amperes or 2.235 Watts. The power consumed by the other library's client was less, 0.301 Amperes or 1.536 Watts.

4.3.2 Computation time: After using timing operations within the two systems, the measurements revealed it takes 2.04 seconds to establish secure context through libcoap, while with uOSCORE-eEDHOC, the operations and message exchange across the two devices takes 46 milliseconds.

4.3.3 Memory Usage: The DRAM and SRAM usage by the libcoap client measured a total of 7049 KB, while that of the uSOCORE-uEDHOC client was significantly less, 2732 KB, representing a reduction of approximately 61.24%.

4.3.4 Confidentiality: DTLS uses encryption of the entire communication channel, ensuring no unauthorized access is made. EDHOC ensures the confidentiality of individual messages by offering end-to-end encryption [21].

4.3.5 Integrity: In DTLS, integrity is achieved by a Hash-based Message Authentication Code (HMAC) on the receiver side, allowing the detection of any altercation to the records. [14]. Conversely, EDHOC makes use of CBOR Object Signing and Encryption (COSE) for cryptography and identification of credentials [21], ensuring message integrity.

4.3.6 Authentication: Authentication in DTLS is certificate-based, while EDHOC has more options, such as RPK or public-key certificates. In DTLS, there are three types of handshake: unauthenticated, server-authenticated, and fully authenticated. As the handshake phase includes multiple packet exchanges, it introduces significant communication and energy consumption overhead [15]. In contrast, EDHOC's handshake requires only three messages, making it a more compact alternative for constrained environments.

5 DISCUSSION

The results from the previous Section 4 related to performance and security are analyzed for the purpose of providing a better understanding of the trade-offs and implications for enhancing cybersecurity in resource-constrained IoT environments.

5.1 Answering RQ1

The three authentication methods have certain advantages and possible disadvantages. While EDHOC ensures PFS through ephemeral keys, signature keys reused across sessions do not offer forward secrecy. This emphasizes the necessity for frequent key updates to ensure the system's robust security. DH static keys provide benefits such as forward secrecy and small message sizes. Given that the DH authentication mechanism is weaker and can be vulnerable to MiTM attacks, robust authentication should be ensured.

Authentication can be strengthened through digital signatures by utilizing signature keys. CBOR certificates reduce the risk of devices being impersonated in the network; however, as seen from the experimental data, the latency is significantly higher than that of the other methods. Moreover, the message overhead can increase, posing a burden for some resource-constrained devices.

Considering these factors, authentication mechanisms should be chosen according to the system's requirements. Hybrid approaches may be suitable in order to leverage the strengths and mitigate any potential weaknesses.

5.2 Answering RQ2

These results from the performed experiments suggest that the EDHOC protocol's design, coupled with the efficient implementation of the libcoap library, ensures high performance even without hardware acceleration. The similarity in performance metrics is proof of the protocol's suitability for constrained environments, providing robust security without imposing significant computational or energy overheads. While there is a slight difference between the memory usage between the two implementations, this can be explained by the fact that hardware acceleration offloads cryptographic computations to dedicated hardware, reducing the CPU and software load and the memory requirements.

By incorporating these findings, the study highlights that while hardware acceleration can enhance performance, the efficient design of the EDHOC protocol and its implementation ensures that IoT devices can maintain secure and efficient communication without supporting it. This is particularly beneficial for resource-constrained IoT environments where hardware acceleration may not always be feasible.

To achieve more conclusive results, it would be beneficial to perform more thorough tests on different microcontrollers with different types of hardware acceleration, as ESP32S is only one example.

5.3 Answering RQ3

The DTLS and EDHOC protocols offer a similar degree of security and available features. As the protocols are suitable for both IPv6 and IPv4, they are accessible for a variety of devices. Based on the measured results in the previous section, using uOSCORE-uEDHOC demonstrated a significant improvement from libcoap together with OSCORE with DTLS secure context establishment. It is important to keep in mind that the latter implementation also contained an improved version of OSCORE and EDHOC, which was designed specifically for constrained devices. The experiments highlight that the computational overhead was much smaller, which could be explained by the fact that securing CoAP messages through DTLS introduces significant delay as it requires decrypting the messages at proxies and a significant number of roundtrips. Additionally, the difference of 31.3% between power consumption in Watts and 23.8% in electrical current further proves OSCORE and EDHOC's efficiency over DTLS.

When it comes to the security offered by both protocols, although achieved by different means, both ensure the communication between IoT devices is tamper-proof. Consequently, both protocols have been optimized for different usages and have considerable strengths. While DTLS may not be as tailored to constrained devices as EDHOC, its benefits are notable as its standardization is wider, meaning integration into existing technologies can be simpler. Furthermore, it is highly suitable for the continuous transfer of large volumes of data where the robust protection of an entire communication session is crucial. Nevertheless, in relation to resource-limited systems, EDHOC remains more lightweight for its efficiency and low overhead.

6 CONCLUSIONS

This research explored the performance of the OSCORE and EDHOC protocols for resource-constrained IoT devices. Different key management strategies, including Diffie-Hellman static keys, signature keys, and CBOR certificates, offer their respective strengths and weaknesses. Using DH static keys allows a balance between security and performance to be met; however, the system becomes vulnerable without proper authentication. Although there is no guaranteed forward secrecy without frequent key updates, signature keys offer strong encryption and integrity, and CBOR certificates, despite being highly secure, can introduce a significant delay in the program as well as an increase in flash memory usage. These findings emphasize selecting authentication mechanisms based on specific application requirements is of key importance.

The impact of hardware acceleration on the EDHOC protocol revealed minimal impact in terms of computation time, memory usage, and power consumption. This efficiency of the protocol makes it suitable for environments where hardware acceleration is not possible, demonstrating that secure communication can be feasible with limited resources.

A comparison between two implementations, one using OSCORE and EDHOC and the other using only OSCORE with DTLS security context establishment, revealed higher power consumption and computational overhead when the latter was used. Consequently, OSCORE and EDHOC protocols have proven to be better suited for devices with limited resources, although DTLS can have considerable strengths in continuous data transfer scenarios.

Overall, the research provides insights into the trade-offs between security, latency, and resource utilization in IoT networks using OSCORE and EDHOC protocols, highlighting the need for additional investigation into possible optimizations.

7 FUTURE WORK

This demonstration indicates it is crucial to explore the possibilities of OSCORE and EDHOC as it would be beneficial to have wider standardization and use of lightweight protocols in IoT networks. Evaluating the protocols for multi-hop scenarios would provide insight into possible packet loss. Future improvements to the EDHOC and OSCORE implementation may include adding a caching mechanism similar to NDN's for better performance in multi-hop scenarios. Moreover, it can be beneficial to see what impact crypto-accelerators can make when different microcontrollers are used instead.

ACKNOWLEDGMENTS

Gratitude is extended to this research's supervisor, Dr. Mohammed Elhajj, whose invaluable guidance helped shape the outcome and direction of the conducted study.

ADDITIONAL TOOLS

During the preparation of this work, the author used Grammarly to correct writing errors and enhance readability. After using this tool, the author reviewed and edited the content as needed and takes full responsibility for the work.

REFERENCES

[1] Dan-Radu Berte. 2018. Defining the IoT. *Proceedings of the International Conference on Business Excellence* 12, 1 (2018), 118–128. https://doi.org/doi:10.2478/picbe-2018-0013

[2] Alessandro Bruni, Thorvald Sahl Jørgensen, Theis Grønbech Petersen, and Carsten Schürmann. 2018. Formal Verification of Ephemeral Diffie-Hellman Over COSE (EDHOC). In *Security Standardisation Research*, Cas Cremers and Anja Lehmann (Eds.). Springer International Publishing, Cham, 21–36.

[3] Baptiste Cottier and David Pointcheval. 2022. Security Analysis of the EDHOC protocol. arXiv:2209.03599 [cs.CR] https://arxiv.org/abs/2209.03599

[4] Geovane Fedrecheski, Mališa Vučinić, and Thomas Watteyne. 2024. Performance Comparison of EDHOC and DTLS 1.3 in Internet-of-Things Environments. In *IEEE Wireless Communications and Networking Conference*. Dubai, United Arab Emirates. https://hal.science/hal-04382397

[5] Cenk Gündogan, Christian Amsüss, Thomas C. Schmidt, and Matthias Wählisch. 2020. IoT Content Object Security with OSCORE and NDN: A First Experimental Comparison. In *Proc. of 19th IFIP Networking Conference* (Paris, France). IEEE Press, Piscataway, NJ, USA, 19–27. https://ieeexplore.ieee.org/document/9142731

[6] Cenk Gündoğan, Peter Kietzmann, Martine Lenders, Hauke Petersen, Thomas C. Schmidt, and Matthias Wählisch. 2018. NDN, CoAP, and MQTT: a comparative measurement study in the IoT. In *Proceedings of the 5th ACM Conference on Information-Centric Networking* (Boston, Massachusetts) *(ICN '18)*. Association for Computing Machinery, New York, NY, USA, 159–171. https://doi.org/10.1145/3267955.3267967

[7] Martin Gunnarsson, Joakim Brorsson, Francesca Palombini, Ludwig Seitz, and Marco Tiloca. 2021. Evaluating the performance of the OSCORE security protocol in constrained IoT environments. *Internet of Things* 13 (2021), 100333. https://doi.org/10.1016/j.iot.2020.100333

[8] David Harris-Birtill and Rose Harris-Birtill. 2021. *Chapter 12 Understanding Computation Time: A Critical Discussion of Time as a Computational Performance Metric*. Brill, Leiden, The Netherlands, 220 – 248. https://doi.org/10.1163/9789004470170_014

[9] Stefan Hristozov, Manuel Huber, Lei Xu, Jaro Fietz, Marco Liess, and Georg Sigl. 2021. The Cost of OSCORE and EDHOC for Constrained Devices. In *Proceedings of the Eleventh ACM Conference on Data and Application Security and Privacy* (Virtual Event, USA) *(CODASPY '21)*. Association for Computing Machinery, New York, NY, USA, 245–250. https://doi.org/10.1145/3422337.3447834

[10] Rikard Höglund, Marco Tiloca, Göran Selander, John Preuß Mattsson, Mališa Vučinić, and Thomas Watteyne. 2024. Secure Communication for the IoT: EDHOC and (Group) OSCORE Protocols. *IEEE Access* 12 (2024), 49865–49877. https://doi.org/10.1109/ACCESS.2024.3384095

[11] IETF. 2019. *Object Security for Constrained RESTful Environments (OSCORE)*. https://datatracker.ietf.org/doc/html/rfc8613

[12] S. L. Keoh, S. S. Kumar, and H. Tschofenig. 2014. Securing the Internet of Things: A Standardization Perspective. IEEE Internet of Things Journal. , 43-49 pages. https://doi.org/10.1109/JIOT.2014.2312291

[13] Jiyoon Kim, Daniel Gerbi Duguma, Sangmin Lee, Bonam Kim, JaeDeok Lim, and Ilsun You. 2021. Scrutinizing the Vulnerability of Ephemeral Diffie–Hellman over COSE (EDHOC) for IoT Environment Using Formal Approaches. *Mobile Information Systems* 2021 (2021), 1–18. https://doi.org/10.1155/2021/7314508

[14] Thomas Kothmayr, Corinna Schmitt, Wen Hu, Michael Brünig, and Georg Carle. 2013. DTLS based security and two-way authentication for the Internet of Things. *Ad Hoc Networks* 11, 8 (2013), 2710–2723. https://doi.org/10.1016/j.adhoc.2013.05.003

[15] Hyeokjin Kwon, Jiye Park, and Namhi Kang. 2016. Challenges in Deploying CoAP Over DTLS in Resource Constrained Environments. In *Information Security Applications*, Ho-won Kim and Dooho Choi (Eds.). Springer International Publishing, Cham, 269–280.

[16] Tapio Levä, Oleksiy Mazhelis, and Henna Suomi. 2014. Comparing the cost-efficiency of CoAP and HTTP in Web of Things applications. *Decision Support Systems* 63 (2014), 23–38. https://doi.org/10.1016/j.dss.2013.09.009 1. Business Applications of Web of Things 2. Social Media Use in Decision Making.

[17] Malwarebytes. 2023. What was the Mirai Botnet? Malwarebytes. https://www.malwarebytes.com/what-was-the-mirai-botnet

[18] Francesca Palombini, Marco Tiloca, Rikard Höglund, Stefan Hristozov, and Göran Selander. 2024. *Using Ephemeral Diffie-Hellman Over COSE (EDHOC) with the Constrained Application Protocol (CoAP) and Object Security for Constrained RESTful Environments (OSCORE)*. Internet-Draft draft-ietf-core-oscore-edhoc-11. Internet Engineering Task Force. https://datatracker.ietf.org/doc/draft-ietf-core-oscore-edhoc/11/ Work in Progress.

[19] Biagio Peccerillo, Mirco Mannino, Andrea Mondelli, and Sandro Bartolini. 2022. A survey on hardware accelerators: Taxonomy, trends, challenges, and perspectives. *Journal of Systems Architecture* 129 (2022), 102561. https://doi.org/10.1016/j.sysarc.2022.102561

[20] T. P. Santos, R. Chaves, E. Homsirikamol, M. Rogawski, and O. Adamo. 2021. Efficient Implementation of Lightweight Cryptographic Algorithms for Securing IoT Devices. Cryptology ePrint Archive, Report 2021/058. https://eprint.iacr.org/2021/058.pdf

[21] G. Selander, J. Preuß Mattsson, and F. Palombini. 2024. Ephemeral Diffie-Hellman Over COSE (EDHOC). Internet Requests for Comments. https://doi.org/10.17487/RFC9528

[22] Sensative. 2020. *End-to-End Security for Constrained IoT Environments Based on OSCORE*. https://sensative.com/iot_use_cases/end-to-end-security-for-constrained-iot-environments-based-on-oscore/

[23] Zach Shelby, Klaus Hartke, and Carsten Bormann. 2014. The Constrained Application Protocol (CoAP). Internet Engineering Task Force (IETF). https://www.rfc-editor.org/rfc/rfc7252

[24] S. Singh, P.K. Sharma, S.Y. Moon, et al. 2024. Advanced lightweight encryption algorithms for IoT devices: survey, challenges and solutions. *Journal of Ambient Intelligence and Humanized Computing* 15 (2024), 1625–1642. https://doi.org/10.1007/s12652-017-0494-4

[25] William Stallings and Lawrie Brown. 2007. *Computer Security: Principles and Practice*.

[26] Zhiyi Zhang, Yingdi Yu, Haitao Zhang, Eric Newberry, Spyridon Mastorakis, Yanbiao Li, Alexander Afanasyev, and Lixia Zhang. 2018. An Overview of Security Support in Named Data Networking. *IEEE Communications Magazine* 56, 11 (2018), 62–68. https://doi.org/10.1109/MCOM.2018.1701147