Pre-Shared Key Authentication in Ephemeral Diffie-Hellman Over COSE

Elsa Lopez Perez, Thomas Watteyne, Mališa Vučinić

Pre-Shared Key Authentication in Ephemeral Diffie-Hellman Over COSE

Elsa Lopez Perez
Inria
Paris, France
elsa.lopez-perez@inria.fr

Thomas Watteyne
Inria
Paris, France
thomas.watteyne@inria.fr

Mališa Vučinić
Inria
Paris, France
malisa.vucinic@inria.fr

***Abstract*—This paper describes preliminary research on enhancing the Ephemeral Diffie-Hellman Over COSE (EDHOC) protocol with a new pre-shared key (PSK) authentication method. The research focuses on the design and the potential benefits of PSK for improving session key update efficiency, and reducing computational overhead compared to existing authentication methods in EDHOC. We outline the planned implementation and evaluation strategy that will include performance metrics such as latency, computational cost, memory usage, and energy consumption across various hardware and software configurations. This work aims to optimize EDHOC for secure communication in resource-constrained environments.**

***Index Terms*—Pre-Shared Key, Lightweight Security, Wireless Communication, Protocol Design and Analysis.**

INTRODUCTION

Motivation

The growth of the Internet of Things (IoT) has pushed organizations like the Internet Engineering Task Force (IETF) to create protocols suited for the needs of IoT devices and networks. Key challenges in these environments include limited bandwidth (often just bytes per second), limited memory (with hundreds of kilobytes for code and tens of kilobytes for RAM), intermittent communication with potential delays of several seconds, small maximum transmission units (around 50 bytes), and limited processing power (in the range of tens of Megahertz).

To tackle these challenges, the Internet community has come up with and standardized protocols designed for constrained environments, such as the Object Security for Constrained RESTful Environments (OSCORE). OSCORE is a security protocol that secures CoAP (Constrained Application Protocol) communication, offering end-to-end encryption and integrity, replay protection, and binding responses to requests across CoAP proxies.

However, OSCORE doesn't include a key establishment protocol. To fill this gap, the IETF's Lightweight Authenticated Key Exchange (LAKE) Working Group (WG) developed a key exchange protocol called Ephemeral Diffie-Hellman Over COSE (EDHOC).

EDHOC Outline

EDHOC is a compact handshake protocol that builds upon elements already used in OSCORE, including the "Concise Binary Object Representation" (CBOR) and COSE, the object security format CBOR Signature and Encryption (COSE). This leads to reduced message footprint.

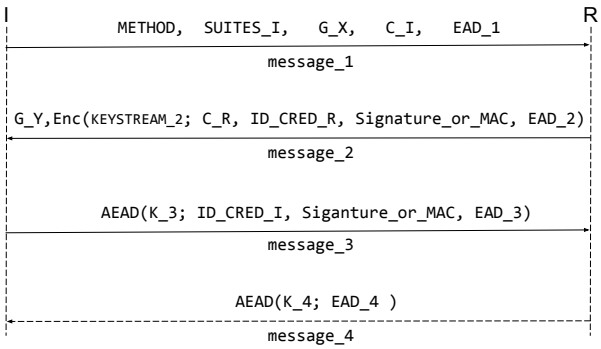


Fig. 1: The EDHOC message flow. G_X and G_Y represent the public ephemeral key of the Initiator and the Responder, respectively. Field ID_CRED_I (resp. ID_CRED_R) denotes the authentication credentials containing the public authentication keys of the Initiator (resp. Responder). Method is an integer (0-1-2-3) denoting the authentication method (see Table I). Cipher Suites is an ordered set of preferred algorithms (hash, encryption). If method is either 0 or 1 for the Initiator (resp. 0 or 2 for the Responder), then Sig or MAC equals Sig. The fourth message is optional (represented with a dashed line).

The protocol uses handshakes with messages around 100 bytes in size, requires only three mandatory flights, is transport-agnostic, and allows the code size to be low by reusing the same elements as OSCORE. Fedrecheski *et al.* [1] show that EDHOC achieves a 7.75× reduction in message footprint, 1.9× reduction in energy and time, and uses up to 4× less flash and RAM than DTLS 1.3.

The EDHOC cryptographic core is designed following the SIGMA-I protocol, and, more precisely, the MAC-then-Sign variant. In a nutshell, the SIGMA protocol is a family of key-exchange protocols that introduce a general approach to building authenticated Diffie-Hellman (DH) protocols using a combination of digital signatures and message authentication code (MAC) functions. The SIGMA-I variant provides identity protection against active attackers to the peer initiating the session, called Initiator, whereas it only provides identity

ID	Initiator	Responder
0	Signature	Signature
1	Signature	Static Diffie-Hellman
2	Static Diffie-Hellman	Signature
3	Static Diffie-Hellman	Static Diffie-Hellman

TABLE I: Current authentication methods registered by IANA.

protection against passive attackers to the peer engaging in an already initialized session, called Responder. The MAC-then-Sign variant includes the MAC under the signature in order to reduce the size of messages on the wire.

Apart from conventional signature keys used for authentication, EDHOC enables the use of static Diffie-Hellman keys. The EDHOC "authentication method", sent in the first message, defines which type of the authentication key the peers are using (see Table I).

Fig. 1 depicts the EDHOC message flow. The protocol consists of three mandatory messages, an optional fourth message, and an error message.

Pre-shared Key Authentication

There has been a recent interest in developing a pre-shared key (PSK) authentication method for EDHOC. This authentication method was proposed in the first drafts of EDHOC, and was ruled out to speed out the development process.

One use case of PSK authentication in EDHOC is the update of session keys. This method aims at reducing the computational cost that comes with re-running the protocol with public authentication keys. This efficiency is beneficial in scenarios where frequent key updates are needed, such in resource-constrained environments or applications requiring high-frequency secure communications. The use of PSK authentication in EDHOC ensures that session key can be refreshed without heavy computational overhead, typically associated with public key operations, thus optimizing both performance and security.

Another use case is the resumption capability in Extensible Authentication Protocol (EAP) leveraging EDHOC. EAP-EDHOC resumption aims at providing a streamlined process for re-establishing secure sessions, reducing latency and resource consumption. By employing PSK authentication for key updates, EAP-EDHOC resumption can achieve secure session resumption, enhancing overall efficiency and user experience.

EDHOC with PSK authentication is also beneficial for existing systems where two nodes have been provided with a PSK from other parties. This allows the nodes to perform ephemeral Diffie-Hellman to achieve Perfect Forward Secrecy (PFS), ensuring that past communications remain secure even if the PSK is compromised. The authentication provided by EDHOC prevents eavesdropping by on-path attackers, as they would need to be active participants in the communication to intercept and potentially tamper with the session. Examples include Generic Bootstrapping Architecture (GBA), Authenticated Key Management Architecture (AKMA) in mobile systems, or Peer and Authenticator in EAP.

Research Challenge of my PhD

The goal for my PhD research is to explore the implementation performance of the PSK authentication method within EDHOC, which will correspond to method 4 in Table I. This involves implementing PSK-based authentication and comparing its performance and security characteristics against other authentication methods. This comparison includes analyzing the computational overhead, latency, and security robustness of PSK authentication relative to signature and static Diffie-Hellman authentication modes. My ambition is to provide a comprehensive evaluation of PSK authentication in EDHOC, highlighting its advantages and potential drawbacks in various application contexts. This research contributes to a deeper understanding of how PSK can be effectively utilized to enhance secure communications, particularly in environments where efficiency and low computational cost are critical.

STATE-OF-THE-ART

The LAKE WG (https://datatracker.ietf.org/wg/lake/about/) was formed in 2019 and the first draft of EDHOC, version 00, was published in July 2020. Since then, 23 drafts have been published, until the publication of RFC9528 in March 2024.

Prior to TLS 1.3, the process of standardization and formalization took place sequentially. Formalization, which involves the rigorous mathematical analysis of the protocol's security properties, often occurred independently or as part of academic research efforts. Meanwhile, standardization efforts led by organizations such as the IETF focused on developing and documenting the protocol specification, considering practical deployment considerations, interoperability requirements, and feedback from implementers. As formal analysis in protocol design gained importance and recognition, formalization of protocols started to be integrated on the standardization process, so that both occurred simultaneously.

Formal analyses of protocols can be done using two approaches. First, a symbolic model, often called Dolev-Yao model, an adversary model that employs idealized cryptographic primitives and that allows the adversary to control the communication channel and interact with protocol sessions by dropping, injecting or modifying messages Second, a computational model, in which messages are modeled as bit strings, the cryptographic primitives are functions from bit strings to bit strings, and the adversary is any probabilistic Turing machine.

Table II summarizes the vulnerabilities that were found during the formalization process, to which draft they correspond, whether the analysis used a symbolic or computational model, what was the suggested improvement, and in what draft was it included.

Security goal	Vulnerability	Mitigation	Initial draft	Improved draft	Method	Proof type	Ref.
Confidentiality	Weak final key. Reuse of the last key-exchange internal key	Final key depending on PRK_{4x3m} and TH_4 (PRK_{out})	12	14	0-1-2-3	S	[2]
	Transcript collision attack	Reorder arguments in the hash function	12	14	0-1-2-3	S	[2]
	Duplicate Signature Key Selection (identity misbinding attack)	Include full/unique authentication credentials in the hash function. Build transcript hashes based on plaintext	14	17	0	C	[3]
	Key reuse	Not to reuse keys across calls of $EDHOC\_Extract$ and $EDCHO\_Expand$	14	17	0	C	[3]
	Salt Collision Attack	Use TH_2 as salt in the HKDF Extract function to derive PRK_{2e}	15	16	3	C	[4]
Mutual authentication	KCI	Modify the construction of message 3	15	-	3	C	[4]
	Leaking ephemeral secrets breaks authentication	Entity authentication should only rely on long-term authentication secrets	12	14	0-1-2-3	S	[2]
	Injective agreement	Add a fourth message as an option	00	Op.	0-1-2-3	S	[5]
Identity protection	Initiator impersonation	Include the Initiator identity in the list of trusted identities for the Initiator	12	14	0-1-2-3	S	[2]
	Partial privacy disclosure of the Responder's identity	Authenticate the first message and provide a way to validate the second message	07	-	0-1-2-3	S	[6]
Cryptographic strength	Attacks in 2^{64} operations for the Responder	Add a fourth message	15	Op.	3	C	[4]
Protection of external data	AEAD Key/IV reuse	Do not allow message recomputation from stored data	12	14	0-1-2-3	S	[2]
Non-repudiation	Unclear intended use	The Initiator should verify whether the identity of the Responder matches the intended one.	00	05	0-1-2-3	S	[5]
	Malleable signatures	Do not accept low-order points or the identity group element	12	-	0-1-2-3	S	[2]
Downgrade protection	Sessions sharing the same PRK_4e3m	Do not accept low-order points or the identity group element	12	-	0-1-2-3	S	[2]
	Unclear intended use	The Initiator should verify whether the identity of the Responder matches the intended one.	00	05	0-1-2-3	S	[5]

TABLE II: Different vulnerabilities found during the security analysis of EDHOC and their corresponding mitigation. "Op." denotes optional, and "-" denotes not included. "S" denotes symbolic analysis, "C" denotes computational analysis. This table is reproduced as-is from work currently under review [7].

Experimental Setup

We aim to develop a PSK authentication method in EDHOC. To this end, we have defined two different approaches, which offer different security and privacy considerations.

- Approach A follows the structure of TLS 1.3, sending the ID_CRED_PSK in message_1 in the clear, with the Responder authenticating first (see Fig. 2a),
- Approach B deviates from TLS 1.3 by sending the ID_CRED_PSK in message_3 encrypted using a key derived from the ephemeral shared secret G_XY. In this case, the Initiator authenticates first (see Fig. 2b).

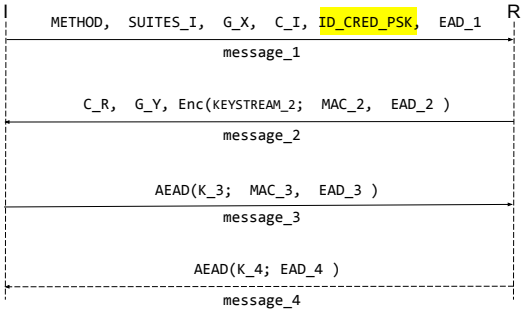
In order to evaluate the performance of the different approaches, as well as comparing them with the already existing authentication methods, we are in the process of integrating them in the current Rust implementation of EDHOC called *lakers* (https://github.com/openwsn-berkeley/lakers)

Evaluation

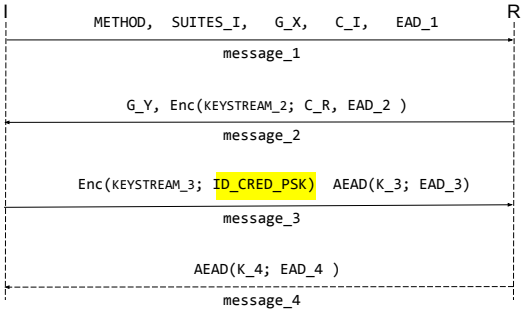
To evaluate the performance and security of the PSK authentication method in EDHOC, we intend to measure various metrics across different experimental setups, utilizing both hardware and software platforms. These metrics will include handshake latency, energy consumption, payload, memory consumption, and overall throughput. We will use the nRF52840-DK and nRF5340-DK development kits to run our code. These feature the state-of-the-art nRF52840 and nRF5340 chips, respectively.

We will be measuring the following performance indicators:

- **Energy Consumption.** This can be measured using the OTII, a power measurement tool designed to measure the power consumption of embedded devices accurately, to which we connect the nRF. During the execution of the code, the OTII software will record the power



(a) Message flow of EDHOC with symmetric keys. Approach A.



(b) Message flow of EDHOC with symmetric keys. Approach B.

Fig. 2: Message flow of EDHOC with PSK.

consumption data and we will be able to detect peaks and average consumption.

- **Latency.** We will need to use timestamps at critical points, such as the start or end of the handshake, that will be later on be analyzed using the OTII software.
- **Memory consumption.** We can distinguish two types of memory: flash memory and RAM memory. Flash memory can be measured using the "size" command of the GNU toolchain. This command provides information about different sections of the compiled binary, such as text, data and `bss` (uninitialized data). The `bss` column serves as well to partially quantify RAM memory, in addition to the stack and heap memory. A common technique to measure RAM memory consists of memory painting (also known as watermarking). The idea is to fill the RAM with a known pattern before running the application. After the application has executed, we can check the memory to see which parts have been overwritten, indicating actual RAM usage.
- **Payload size.** Logging can be used to record the size of the payload, allowing to capture the exact data being transmitted. Another common technique consists in using a serial monitor or a network packet analyzer to capture the transmitted messages. Tools like Wireshark can be used to capture network traffic.
- **Security.** Security can be quantify evaluating potential threats, cryptographic strength, protocol analysis, implementation testing, and resistance to known attacks. Formal verification techniques and implementation testing ensure protocol correctness and resilience against various attack vectors. Additionally, side-channel analysis helps identify and mitigate vulnerabilities related to timing and power consumption.
- **Privacy.** Measuring and checking privacy involves assessing data handling practices, access controls, user consent mechanisms, and transparency to ensure compliance with privacy principles and regulatory requirements. Privacy impact assessments, third-party audits, and user feedback mechanisms serve as essential tools to systematically evaluate and improve privacy practices.

Conclusion

The EDHOC protocol with PSK authentication is a robust solution for secure key exchange in constrained environments. By leveraging a pre-shared key, it ensures strong mutual authentication and forward secrecy, while remaining simple to implement and scalable. The adoption of EDHOC with PSK can enhance the security and reliability of IoT networks, promote interoperability, and drive standardization across industries, making it a valuable tool for securing communications in resource-limited settings.

Acknowledgments

This document is issued within the frame and for the purpose of the OpenSwarm project. This project has received funding from the European Union's Horizon Europe Framework Programme under Grant Agreement No. 101093046. Views and opinions expressed are however those of the author(s) only and the European Commission is not responsible for any use that may be made of the information it contains.

References

[1] Geovane Fedrecheski, Mališa Vučinić, and Thomas Watteyne. Performance Comparison of EDHOC and DTLS 1.3 in Internet-of-Things Environments. In *IEEE Wireless Communications and Networking Conference (WCNC)*, Dubai, United Arab Emirates, 21–24 April 2024.

[2] Charlie Jacomme, Elise Klein, Steve Kremer, and Maiwenn Racouchot. A Comprehensive, Formal and Automated Analysis of the EDHOC Protocol. In *USENIX Security Symposium (USENIX)*, Anaheim, CA, USA, August 2023.

[3] Felix Günther and Marc Ilunga Tshibumbu Mukendi. Careful with MAc-then-SIGn: A Computational Analysis of the EDHOC Lightweight Authenticated Key Exchange Protocol. In *IEEE European Symposium on Security and Privacy (EuroS&P)*, Delft, The Netherlands, July 2023.

[4] Baptiste Cottier and David Pointcheval. Security Analysis of Improved EDHOC Protocol. In *International Symposium on Foundations and Practice of Security (FPS)*, Ottawa, ON, Canada, 2022.

[5] Karl Norrman, Vaishnavi Sundararajan, and Alessandro Bruni. Formal Analysis of EDHOC Key Establishment for Constrained IoT Devices. *arXiv*, 2020.

[6] Jiyoon Kim, Daniel Gerbi Duguma, Sangmin Lee, Bonam Kim, JaeDeok Lim, and Ilsun You. Scrutinizing the Vulnerability of Ephemeral Diffie–Hellman over COSE (EDHOC) for IoT Environment Using Formal Approaches. *Mobile Information Systems*, 2021.

[7] Elsa Lopez Perez, Göran Selander, John Preuß Mattsson, Thomas Watteyne, and Mališa Vučinić. EDHOC is a New Security Handshake Standard: Overview of Security Analysis. *IEEE Computer Magazine*, 2024. [Manuscirpt submitted for publication].