

Implementation Considerations for Ephemeral Diffie-Hellman Over COSE (EDHOC)

draft-ietf-lake-edhoc-impl-cons-01

Marco Tiloca, RISE

IETF 120 Meeting – Vancouver – July 23rd, 2024

Recap

- › **Adopted as a LAKE WG document in June 2024**

- › **Scope: considerations on side-topics related to the implementation of EDHOC [1]**
 - Those topics are out of scope for EDHOC itself, and [1] focuses on the actual protocol

- › **Topics covered in version -00**
 - Handling of EDHOC sessions and derived applications keys, if become invalid
 - Trust models for learning peers' public authentication credentials on-the-fly
 - Branched, side-processing of incoming EDHOC messages. This includes:
 - › Fetching and validation of authentication credentials
 - › Processing of EAD items, which may play a role in validating authentication credentials

Updates since version -00

- › **New Section 5** – “Using EDHOC over CoAP with Block-wise”
- › **Build on Appendix A.2 of RFC 9528**
 - Transfer of EDHOC with CoAP (RFC 7252); forward or reverse message flow
- › **Build on *draft-ietf-core-oscore-edhoc***
 - More details on EDHOC over CoAP to key OSCORE (RFC 8613)
 - Optimized combination of EDHOC forward message flow and first protected data exchange
 - › Only 2 RTTs, by using an “EDHOC + OSCORE” CoAP request
- › **Considerations on using EDHOC with CoAP and Block-wise transfer (RFC 7959)**
 - The full data to send (body) is split into smaller chunks, each sent as a message payload
 - This topic was covered in *draft-ietf-core-oscore-edhoc*, but deemed too implementation-related

Updates since version -00

- › **Pre-requirements for using the optimized EDHOC workflow**
 - When using Block-wise or not
- › **Number of RTTs to complete EDHOC and a first protected data exchange, based on:**
 - Small or large bodies; using Block-wise or not; using the optimized EDHOC workflow or not
- › **Number of RTTs when using Block-wise:**
 - If the use of Block-wise is not specifically due to using the optimized EDHOC workflow ...
 - › Then the optimized EDHOC workflow always performs better than the original one
 - If the use of Block-wise is specifically due to using the optimized EDHOC workflow ...
 - › Then the optimized EDHOC workflow never performs better than the original one
 - It might actually perform worse than the original EDHOC workflow
 - › The client should resort to using the original EDHOC workflow instead

Updates since version -00

- › **Trust models for learning authentication credentials of other peers**
 - Section 3.0 defines possible trust policies NO-LEARNING and LEARNING
- › **New Section 3.1 – “Enforcement in the EDHOC and OSCORE Profile of ACE”**
 - Profile defined in *draft-ietf-ace-edhoc-oscore-profile*
 - An EDHOC peer acts as Resource Server (RS), another as ACE Client (C)
 - The Authorization Server issuing Access Tokens facilitates C and RS in running EDHOC
- › **The Access Token specifies the public authentication credential of C (AUTH_CRED_C)**
 - C can optionally upload the Access Token to RS within an EDHOC EAD item
 - The first Access Token issued to C for RS likely includes AUTH_CRED_C by value ...
 - ... and RS does not store AUTH_CRED_C yet, but can learn it from the Access Token
- › **For supporting Access Tokens in EAD items, RS has to enforce the policy LEARNING**

Updates since version -00

- › **Side processing of incoming EDHOC messages**
 - Main content already in Sections 4.0-4.3
- › **New Section 4.4 – “Side Processing in Particular Situations”**
 - Intended for special message handling, beyond the common case of Sections 4.0-4.3
- › **Section 4.4.1 – “EDHOC and OSCORE profile of ACE” (now mostly Editor’s notes)**
 - How to consistently enforce NO-LEARNING if an EAD item conveys an Access Token?
 - When to perform a consistency check of ID_CRED_X with the credential in the Access Token?
 - Some of this might be better fitting in *draft-ietf-ace-edhoc-oscore-profile*
- › **Input is welcome on more “particular situations” to cover**
 - *draft-ietf-lake-authz*?
 - *draft-song-lake-ra*?

Next steps

› Content to include in the next versions

- More on side-processing of incoming EDHOC messages in special situations
- Appendix with example certificates to plug-in for testing
- Security considerations

› Process comments and reviews as they come – Please do chime in!

- Feedback and input from authors/implementors of *-lake-authz* and *-lake-ra* are welcome

Thank you!

<https://github.com/lake-wg/edhoc-impl-cons>