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

draft-ietf-lake-edhoc-impl-cons-02

Marco Tiloca, RISE

IETF 121 Meeting – Dublin – November 4th, 2024

Recap

- **Scope: considerations on side-topics related to the implementation of EDHOC [1]**
  - Those topics are out of scope for EDHOC itself

- **Topics that are currently covered**
  - 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
  - Guidelines/advice on using EDHOC over CoAP with Block-wise transfers

- **Explored cases in point, in addition to plain EDHOC**
  - EDHOC and OSCORE profile of ACE --- *draft-ietf-ace-edhoc-oscore-profile*
  - Lightweight authorization using EDHOC --- *draft-ietf-lake-authz*

[1] https://www.rfc-editor.org/rfc/rfc9528.html

Updates since version -01

Aligned content on the EDHOC and OSCORE profile of ACE

- **In general terms, an access token is bound to AUTH_CRED_C**
  - AUTH_CRED_C is the public authentication credential of the ACE client (C), i.e.:
    - CRED_I, if the EDHOC forward message flow is used
    - CRED_R, if the EDHOC reverse message flow is used

- **Uploading of the access token to the ACE resource server (RS)**
  - C uses a dedicated EAD item defined in the EDHOC and OSCORE profile
    - The exact EDHOC message to use depends on the EDHOC message flow used
  - Alternatively, the ACE Authorization Server can upload the access token [2]

- **Clarified associations and binding**
  - (Access token) ←→ (EDHOC Session, OSCORE Security Context)
  - Re-running EDHOC binds the same access token with the new session and context

[2] https://datatracker.ietf.org/doc/draft-ietf-ace-workflow-and-params/

Updates since version -01

Improved trust model for learning authentication credentials of other peers in EDHOC

- Section 3.0 defines the possible trust policies LEARNING and NO-LEARNING

- **NO-LEARNING "as-is" would defeat some use cases altogether**
  - The ACE profile defined in *draft-ietf-ace-edhoc-oscore-profile*
    - The ACE RS is meant to learn AUTH_CRED_C from the access token within an EAD item
  - The lightweight authorization method defined in *draft-ietf-lake-authz*
    - The device U is meant to learn the credential of V (CRED_R) when confirmed by a voucher

- **Admitted use-case specific exceptions for the NO-LEARNING policy**
  - *NO-LEARNING: according to this policy, the peer P trusts CRED only if P is already storing CRED at message reception time,* ***unless in cases where situation-specific exceptions apply and are deliberately enforced (see below)****.*
  - If an acceptable exception applies, the outcome is like if the LEARNING policy was used
  - The steps on message side-processing and related diagrams were updated accordingly

Updates since version -01

Practical exceptions to NO-LEARNING: presence of specific EAD items

- **In the EDHOC and OSCORE profile**
  - The EAD item transporting the access token that includes AUTH_CRED_C
  - Already explained in Section 3.1
- **In the method of *draft-ietf-lake-authz***
  - The EAD item of EDHOC message_2 transporting the voucher
  - TODO: add a short Section 3.2 explaining this
- **Note: the current Section 4.4.1 is moot and can be safely removed**
  - Message handling for the EDHOC and OSCORE profile with NO-LEARNING w/o exceptions
  - The text covers a non-problem – Thanks Göran and Rikard for the discussion!
  - No applicable exceptions ➔ Only ID_CRED_X is left to consider
  - This is sufficient to consistently fail or continue the message processing

Updates since version -01

Guidelines on using EDHOC with CoAP and Block-wise transfers

- **Revised Section 5 to be more general and consistent**
  - The original text came from a removed appendix of *draft-ietf-core-oscore-edhoc*
  - It was written to cover mainly the EDHOC + OSCORE combined request
- **Generalization**
  - Avoid the impression that Block-wise cannot be used in general
  - Block-wise can be used on a data body consisting of (connection ID and) an EDHOC message
- **Related fixes**
  - Renamed items used when evaluating conditions, e.g., s/SIZE_APP/SIZE_BODY
  - Corrected sentence on what can be fragmented when using the EDHOC + OSCORE request
    - Not the combined request, but rather the plain application data that it transports

Mail from Geovane

Sent to the LAKE mailing list on 2024-08-05, 09:36 UTC

- **“Advertising lake-authz support”**
  - *double processing of message_2: it has been argued that EDHOC clients might receive and process message_2 more than once. On the other hand, correct implementations should maintain a state machine and avoid processing messages in unexpected states (note: maybe this point could be added as an implementation consideration in Marco’s draft).*

- **From Section 5.1 of RFC 9528:**
  - *EDHOC messages SHALL be processed according to the current protocol state.*
  - *Different instances of the same message MUST NOT be processed in one EDHOC session. Note that processing will fail if the same message appears a second time for EDHOC processing in the same EDHOC session because the state of the protocol has moved on and now expects something else.*

- **No addition should be needed in this document**

Next steps

- **On learning authentication credentials**
  - New section 3.2 to clarify specific exception to NO-LEARNING policy for *draft-ietf-lake-authz*
  - Remove the current Section 4.4.1 as moot
  - Consistency checks between authentication credential in ID_CRED_X and in an EAD item
    - Placeholder notes in Section 4.4.2

- **Security considerations**

- **Appendix with example certificates to plug-in for testing**

- **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