EDHOC

draft-ietf-lake-edhoc-17

https://github.com/lake-wg/edhoc

IETF 115, LAKE WG, November 08, 2022

Since IETF 114

- edhoc-16
  - updated following security analysis
  - verified after update
  - wire format change
- edhoc-17
  - minor update for WGLC

- traces-03
  - matching edhoc-16/17

As always, details in https://github.com/lake-wg/edhoc

edhoc-15 → edhoc-16

Summary: edhoc-15 → edhoc-16

- Main changes:
  - TH_2 used as salt in the derivation of PRK_2e
  - CRED_R/CRED_I included in TH_3/TH_4

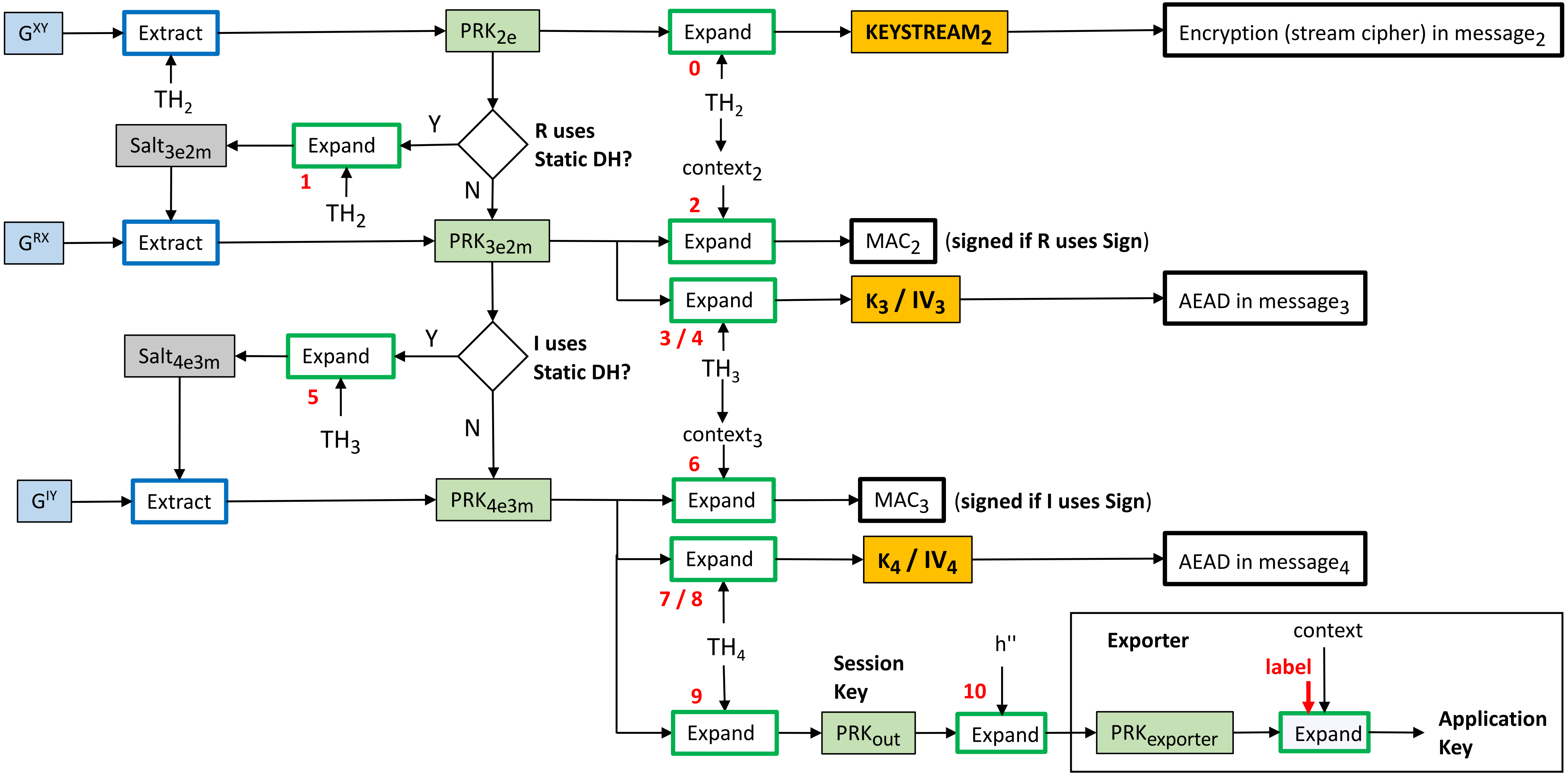
- Minor changes
  - Distinguish label used in info, exporter or elsewhere
    - label → info_label
    - label → exporter_label
  - New Appendix for optional handling arbitrarily large message_2
    - info_label type changed to int to support this
  - Implementation note about identifiers which are bstr/int
  - Clarifications, in particular compact EC representation
  - Type bug fix in CDDL section
  - Updated security considerations
  - Updated references

edhoc-16 → edhoc-17

Summary: edhoc-16 → edhoc-17

- Changes:
  - Security consideration about 128 bit security against online attacks
    - verifying multiple MACs
    - proposed by ENS
  - Updated text on peer awareness
  - EDHOC-KeyUpdate is made OPTIONAL (was RECOMMENDED)
    - EDHOC-KeyUpdate moved to Appendix
  - Clarifications in Appendix on large message_2 (PLAINTEXT_2)

EDHOC-17 Key Schedule



WGLC Comments

WGLC comments

- Marco Tiloca (#347)
- Charlie Jacomme (#344, #351)
- Felix Günther (#350)
- Rafa Marin-Lopez (#352)
- Christian Amsüss (#353)
- Mališa Vučinić (#354)

Thanks!

- From author
  - AEAD with zero plaintext as KEYSTREAM_2 (#355)

Specific issues

- Session key terminology (#344, #354)
- Detecting changes in message_1/2 (#351)
- No exchange defined for KeyUpdate (#352)
- Encoding of TH_2 (#347, #354)
- Protocol state machine (#354)
- EAD error processing (#347)
- Clarify byte string representation (#347)
- Informative references to security analyses (#343, #350)
- AEAD with zero plaintext as KEYSTREAM_2 (#355)

Session key terminology (#344, #354)

- Clarifying session key = PRK_out

Proposal: PR #345

Detecting changes in message_1/2 (#351)

- "Changes in message_1 and message_2 (except PAD_2) are detected when verifying Signature_or_MAC_2. ”

- Correct for strongly unforgeable signature schemes, but not in general
  - EUF-CMA, a signature authenticates only the underlying message
  - SUF-CMA, a signature authenticates both the underlying message and the signature itself

Charlie et al.:

- None of the concrete signature scheme currently standardize appears to be malleable under xor.
- We report it for thoroughness, but are uncertain whether the sentence should be changed or not.

Proposal: PR #356

No exchange defined for KeyUpdate (#352)

- EDHOC-KeyUpdate defined in Appendix J
- Should we define a protocol using it?

- EDHOC-KeyUpdate was defined as a method for forward secrecy
- Overwrites PRK_out
- Requires state

- draft-ietf-core-oscore-key-update defines similar method + protocol
- Independent of EDHOC
- Part of reason why EDHOC-KeyUpdate moved to Appendix J
- No additional use case identified

Proposal: Don’t define the protocol in this draft

Encoding of TH_2 (#347, #354)

- Definition
  - TH_2 = H(G_Y, C_R, H(message_1))
  - "The transcript hash TH_2 is a CBOR encoded bstr ..."   (*)

- Used in various CBOR objects:
  - context_2 = << ID_CRED_R, **TH_2**, CRED_R, ? EAD_2 >>
  - external_aad = << **TH_2**, CRED_R, ? EAD_2 >>
  - TH_3 = H(**TH_2**, PLAINTEXT_2, CRED_R)
  - KEYSTREAM_2 = EDHOC-KDF(PRK_2e, 0, **TH_2**, plaintext_length)
  - SALT_3e2m = EDHOC-KDF(PRK_2e, 1, **TH_2**, hash_length)

- As of -16, also used as salt
  - PRK_2e = HMAC-SHA-256(**TH_2**, G_XY)
- **In traces-03, TH_2 is here the raw byte string output of H(), i.e, not a CBOR item.**
- **Either: keep that and remove (*), or keep (*) and use CBOR encoded TH_2 in PRK_2e**

- (TH_3 only used in CBOR objects, but has similar formulation)

Protocol state machine (#354)

- Mališa:
  - valid states summarized and illustrated through a figure
  - very useful from the implementor’s point of view
  - similar to Appendix A of RFC 8446
- John:
  - discussed before
  - EDHOC does not really have the kind of states that TLS 1.3 does
  - not against having a figure

Proposal: Sketch an appendix

EAD error processing (#347)

- General rule:
  - "If any processing step fails, the Responder MUST send an EDHOC error message back ..."

Section 3.8

- "If an endpoint receives a critical EAD item it does not recognize or a critical EAD item that contains information that it cannot process, the EDHOC protocol MUST be discontinued."
    - Must an EDHOC error message also be sent before discontinuing the protocol?
    - Is it something that must be specified by the application/specification that defines the EAD item and its processing when used as critical?

- Does "processing" cover also the actual EAD processing, or only the act of making EAD_x available to the application?

Proposal: Apply “MUST send”, considering DoS reasons for not sending (Section 8.7). Clarify that the EAD specification defines when and what to send.

Clarify byte string representation (#347)

- "Connection identifiers in EDHOC are intrinsically byte strings."
- "The byte strings which coincide with a one-byte CBOR encoding of an integer MUST be represented by the CBOR encoding of that integer."
- Other byte strings are encoded as CBOR byte strings.

OLD (edhoc-17):

h'21' is represented by 0x21 (CBOR encoding of the integer -2), not by 0x4121

NEW (proposed change):

0x21 is represented by 0x21 (CBOR encoding of the integer -2), not by 0x4121

OLD (edhoc-17):

h'18' is represented by 0x4118

NEW (proposed change):

0x18 is represented by 0x4118 **(CBOR encoding of the byte string 0x18)**

Proposal: Do this

Informative references to security analyses (#343, #350)

- “Two earlier versions of EDHOC have been formally analyzed [Norrman20] [Bruni18] and the specification has been updated based on the analysis.”

- Incomplete list of references. Some analyses are not yet available or preprints. Some pointers:
  - Jacomme, C., Klein, E., Kremer, S., Racouchot, M., "A comprehensive, formal and automated analysis of the EDHOC protocol", October 2023 (to appear at USENIX Security, January 2023) https://hal.inria.fr/hal-03810102/
  - Cottier, B., Pointcheval, D., "Security Analysis of the EDHOC protocol", September 2022, https://arxiv.org/pdf/2209.03599.pdf
  - Ilunga, M., Günther, F., "Analysis of the EDHOC Lightweight Authenticated Key Exchange Protocol", August 2022, https://www.research-collection.ethz.ch/handle/20.500.11850/576036

- Proposal: At least update the list. Annotated with insights from the analysis?

AEAD with zero plaintext as KEYSTREAM_2 (#355)

- EDHOC-17 uses HMAC and KMAC as stream ciphers for encryption of message_2
  - KEYSTREAM_2 = EDHOC-KDF(PRK_2e, 0, TH_2, plaintext_length)

- COSE does not have IND-CPA encryption algorithms like AES-CTR and ChaCha20
- Hard to remove the tag from an AEAD call such as AES-CCM(K_2, P_2, A, N)

- Missed in the discussion: AEAD with a plaintext consisting of zeroes
- For example, implementing AES-CTR with AES-CCM:
  - KEYSTREAM_2 = AES-CCM(K_2, 0000000……, A, N)
  - CIPHERTEXT_2 = PLAINTEXT_2 XOR (beginning of KEYSTREAM_2)

Next steps

- Address WGLC comments
- Submit updated version of –edhoc (and, if necessary, –traces)