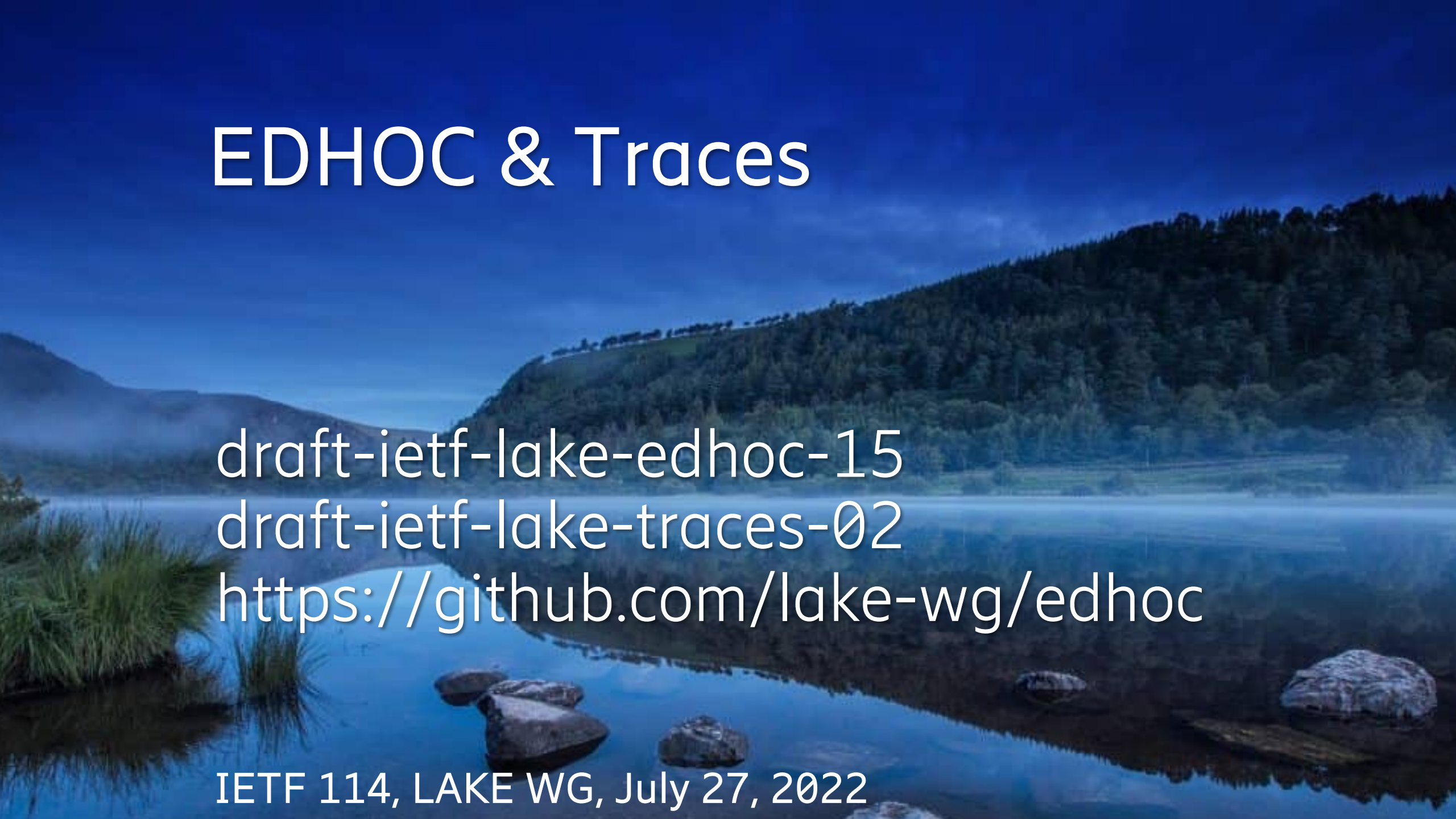


EDHOC & Traces

draft-ietf-lake-edhoc-15
draft-ietf-lake-traces-02
https://github.com/lake-wg/edhoc

IETF 114, LAKE WG, July 27, 2022

Since IETF 113

— edhoc-13 == edhoc12
— edhoc-14, major update, first since Nov. 2021
— edhoc-15, mainly clarifications, same wire format as -14
(overview of changes in the different versions is in the last appendix)

— traces-01, update matching edhoc-14/15
— traces-02, recent bug fix

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

edhoc-13 → edhoc-14

edhoc-13 → edhoc-14

- Merge of section 1.1 and 1.2
- Connection and key identifiers are byte strings (next slide)
- Rewrite of 3.5
  - Clarification of authentication related operations
  - Authentication related verifications, including old section 3.5.1, moved to new appendix D
- Rewrite of 3.8
  - Move content about use of EAD to new appendix E
  - ead_value changed to bstr
- EDHOC-KDF updated
  - transcript_hash argument removed
  - TH included in ‘context’ argument
  - all text string labels are replaced with uints
- Key schedule updated (later slide)
  - New salts derived to avoid reuse of same key with Expand and Extract
  - PRK_4x3m renamed PRK_4e3m (to indicate its use; does not include export anymore)
  - K_4 and IV_4 derived from PRK_4e3m
  - New PRK: PRK_out derived from PRK_4e3m and TH_4
    - Main output of EDHOC
  - New PRK: PRK_exporter derived from PRK_out
  - Exporter defined by EDHOC-KDF and PRK_exporter
  - Key update defined by Expand instead of Extract
- All applications of EDHOC-KDF collected in one table

Identifier Encoding

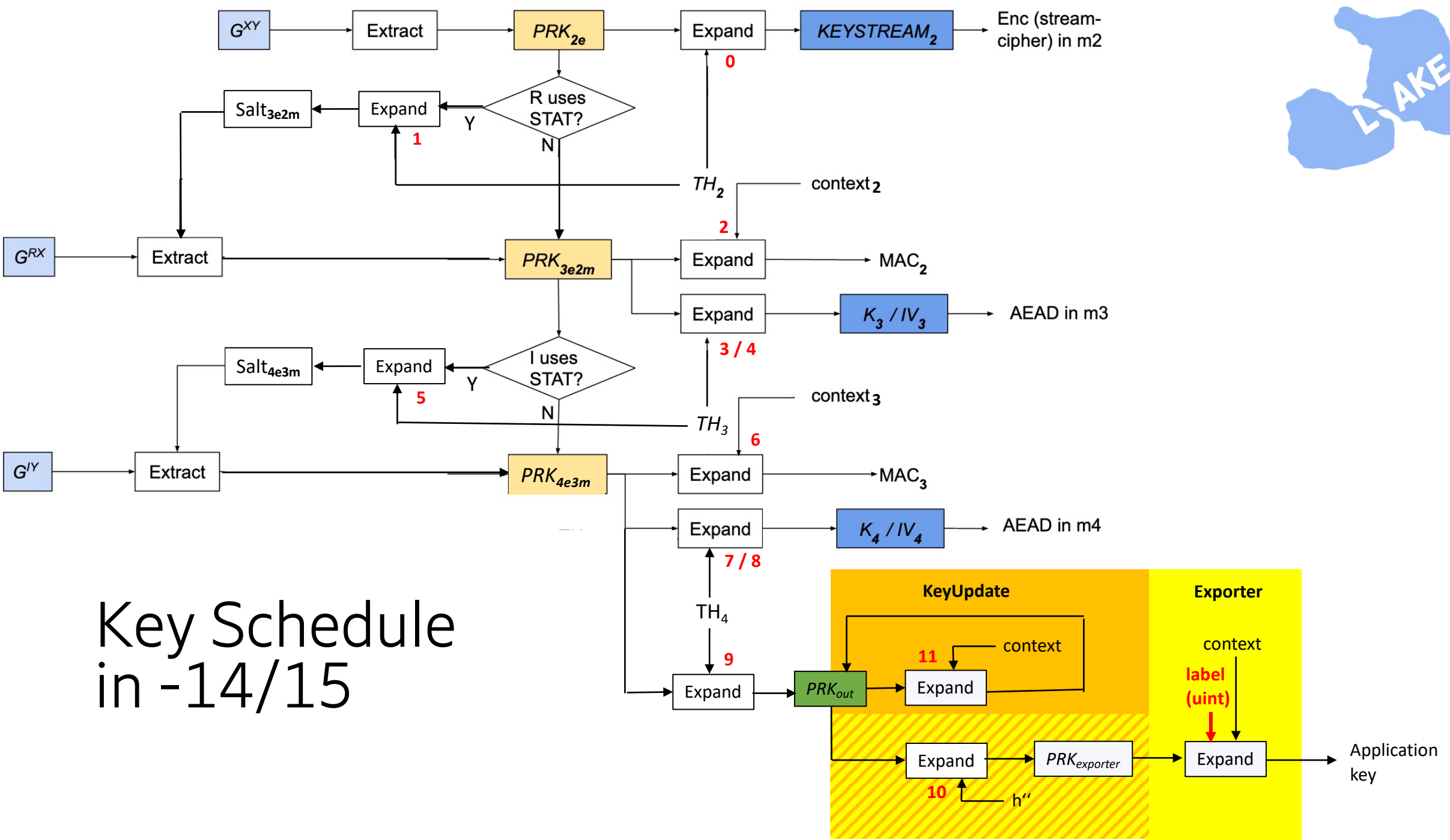
Revisit of old problem:
- OSCORE identifiers and COSE key identifiers (kid) are byte strings
- CBOR byte strings typically at least two bytes long, i.e., not optimal

- Previous solution attempts in LAKE (bstr_identifier) and COSE (int kid)

New solution:
- Connection and key identifiers are intrinsically byte strings
  - Represented as CBOR bstr in the EDHOC message
    - Unless the byte string happen to encode a one-byte CBOR int (-24..23)
      - In which case they are encoded as that CBOR int (i.e. unchanged)

Examples:
- h'0E' is represented by 0x0E (CBOR encoding of the integer 14)
  - not by 0x410E (CBOR encoding of the byte 0x0E)
- h'FF' is represented by 0x41FF
  - since it is not the CBOR encoding of an integer in (-24..23)

- Simplifies mapping between EDHOC and OSCORE identifiers (essentially identity mapping)
- No need for int kids to be defined in COSE



Key Schedule in -14/15

edhoc-13 → edhoc-14

- Update of processing
  - EAD and ID_CRED passed to application when available
  - identity verification and credential retrieval omitted in protocol description
  - Transcript hash defined by plaintext messages instead of ciphertext
  - Changed order of input to TH_2
  - Removed general G_X checking against selfie-attacks
- Support for padding of plaintext
- Updated compliance requirements
- Updated security considerations
  - Updated and more clear requirements on MAC length
  - Clarification of key confirmation
  - Forbid use of same key for signature and static DH
- Updated appendix on message deduplication
- Clarifications of
  - connection identifiers
  - cipher suites, including negotiation
  - EAD
  - Error messages
- Updated media types
- ”applicability template” renamed “application profile”
- Editorials

edhoc-14 → edhoc-15

edhoc-14 → edhoc-15

- EAD update (next slide)
- New section in Appendix D: Unauthenticated Operation
- Clarifications
  - Lengths used in EDHOC-KDF
  - Key derivation from PRK_out
    - EDHOC-KeyUpdate and EDHOC-Exporter
  - Padding
- Security considerations
  - When a change in a message is detected
  - Confidentiality in case of active attacks
  - Connection identifiers should be unpredictable
  - Maximum length of message_2 (later slide)
- Minor bugs

EAD update proposal

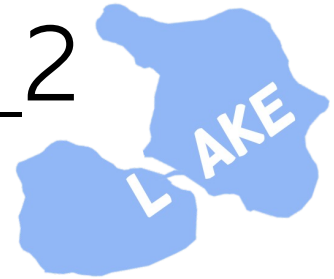
- Defined **EAD item** = (ead_label, ead_value)
  - Each EAD field (EAD_1 .. EAD_4) may contain multiple EAD items
- ead_label > 0 is registered with a specification containing
  - formatting details of ead_value
  - processing
  - security considerations
- An EAD item may be **critical** or **non-critical,** specified by the processing
  - Using the registered positive value indicates that the EAD item is non-critical.
    - The corresponding negative value indicates that the EAD item is critical.
  - If an endpoint receives
    a critical EAD item it does not recognize, or
    a critical EAD item that contains information that it cannot process,
    then the EDHOC protocol MUST be discontinued.
    - A non-critical EAD item can be ignored.

Open Issues and PRs

- Proposal from ETH to include authentication credential in transcript hash (#317, PR #318)
- Proposal from ENS to include TH_2 as salt in PRK_2e derivation (#299, #323)
- Proposal from ENS to derive K_3 from PRK_4e3m (#324)

- Support for size of message_2 > 8160 bytes with SHA-256 (#303, PR #304)

Using SHA-256, do we need larger message_2 than 8160 bytes? (#303)

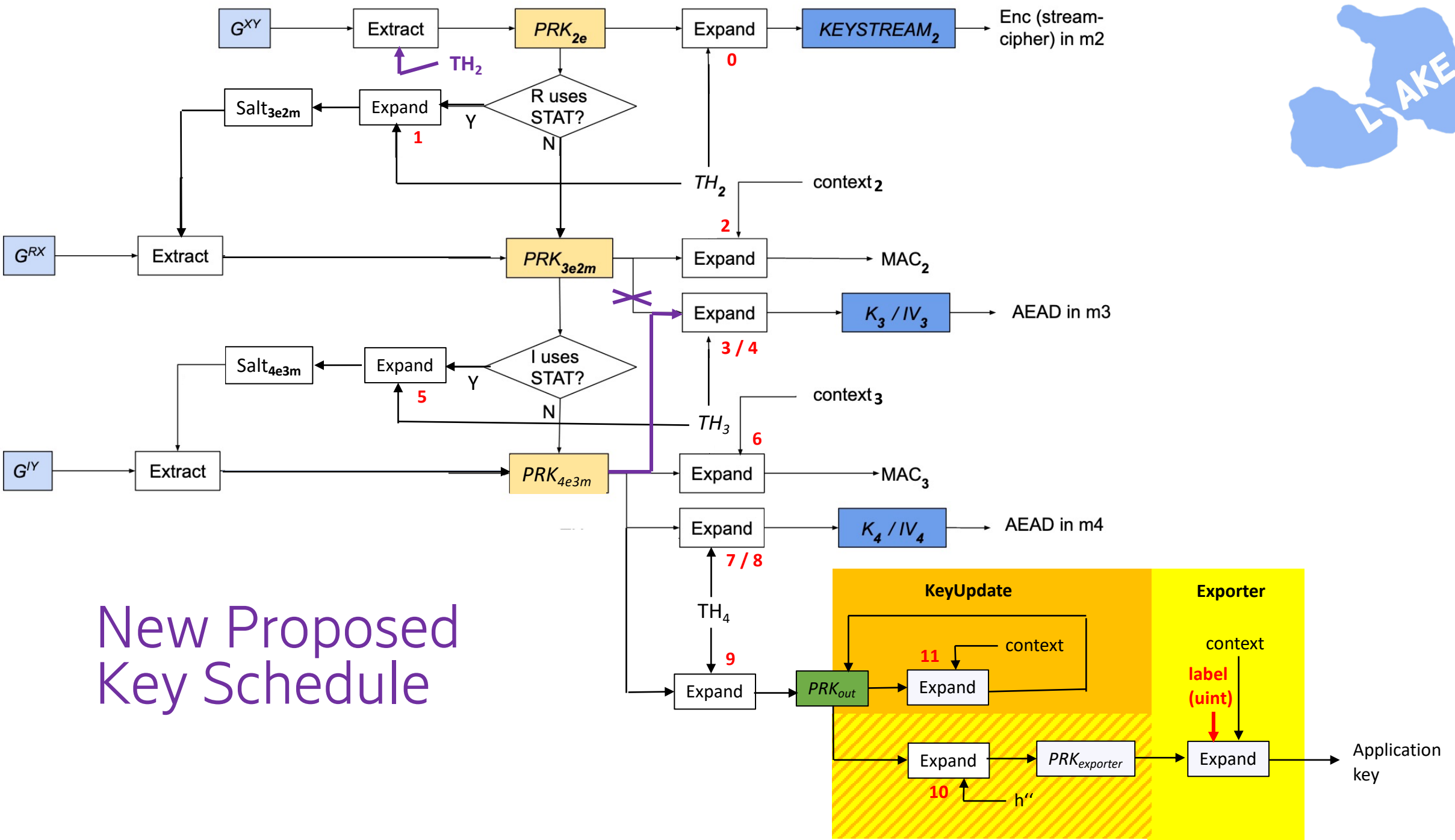


- HKDF paper states that the counter is fixed length.
- RFC 5869 chose 1 byte.
- message_2 is encrypted with $KEYSTREAM_2$ generated with Expand, which with SHA-256 leads to HKDF with max output 255 * 32 = 8160 bytes for the keystream.
- Is this a problem we should fix?
- Even if this is not a problem for typical applications, we may want to define a way to handle larger message_2. Candidates:
  1. Replace HKDF-Expand with HKDF-Expand', which allows larger length of output
  2. Use HKDF-Expand for message_2 size < 8160, HKDF-Expand' for larger message_2
  3. Replace message_2 encryption with AES-CTR / ChaCha20
  4. Use KMAC instead of HKDF
  5. Multiple invocations of HKDF, to produce sufficiently long keystream

5. Multiple invocations of HKDF

- Divide PLAINTEXT_2 in fixed size chunks (of 8160 bytes or similar) + last chunk.
- Introduce dependency on chunk number n = 0, 1, 2, 3, ...
  - one of the first three arguments of the keystream derivation should depend on n
- KEYSTREAM_2 = EDHOC-KDF(PRK_2e, 0, TH_2, plaintext_length)
- Examples
  - PRK_2e(n) = Extract (salt, IKM) = HMAC-SHA-256(n, G_XY)
  - replace second argument with non-positive labels: -n for chunk n = 0, 1, 2, 3, ... (PR #304)
  - replace TH_2 with context = << n, TH_2 >>

New Proposed Key Schedule



-traces

-traces-01/02

- Same two traces as in -00:
  - Method 0 (signature), cipher suite 0 (EdDSA), X.509 certificate identified by ‘x5t’ (hash of cert)
  - Method 3 (static DH), cipher suite 2 (P-256), RPK encoded as CCS identified by ‘kid’ (key id)
    - Cipher suite negotiation (error with SUITES_R)
    - Explicit ‘y’ coordinate of public keys
- New printouts matching the new key schedule and other changes in edhoc-14
- Marco provided first instance of values for -01, added as author
- Stefan verified the trace and found a few bugs
- All known bugs fixed in -02

Next steps

— Address review comments
— Submit updated versions of –edhoc and –traces
— WGLC?