Work in progress towards

# Key Update for OSCORE

draft-hoeglund-core-oscore-key-limits-02

**Rikard Höglund**, RISE
Marco Tiloca, RISE

CoRE WG interim meeting, October 13th, 2021

# Recap

- OSCORE (RFC8613) uses AEAD algorithms to provide security
  - Need to follow limits in key usage and number of failed decryptions, before rekeying
  - Otherwise, it is possible to break the security properties of the AEAD algorithm
  - Reference **draft-irtf-cfrg-aead-limits-03**

- (1) Study of AEAD limits and their impact on OSCORE
  - Defining appropriate limits for OSCORE
  - Defining counters for key usage; message processing details; steps when limits are reached
  - Taking into account John Mattsson's input at the April CoRE interim [1]

- (2) Defined a new method for rekeying OSCORE
  - Loosely inspired by Appendix B.2 of OSCORE
  - Goal: renew the Master Secret and Master Salt; derive new keys from those
  - Achieves Perfect Forward Secrecy

[1] https://datatracker.ietf.org/meeting/110/materials/slides-110-saag-analysis-of-usage-limits-of-aead-algorithms-00.pdf

# Key limits (1/3)

Integrity Advantage (IA): Probability of breaking integrity properties

Confidentiality Advantage (CA): Probability of breaking confidentiality properties

- Recap on AEAD limits
  - Discussed in **draft-irtf-cfrg-aead-limits-03**
  - Limits key encryption use (q) and invalid decryptions (v)
  - We have selected fixed values for ‘q’, ‘v’, and ‘l’ and from those calculated IA & CA probabilities
    - These probabilities must be acceptably low

- Explicitly limit the size of protected data to be sent in a new OSCORE message
  - The probabilities are influenced by ‘l’, i.e., maximum message size in cipher blocks
  - Implementations should not exceed 'l', and it has to be easy to do so
  - New text: *the total size of the COSE plaintext, authentication Tag, and possible cipher padding for a message may not exceed the block size for the selected algorithm multiplied with 'l‘*
  - Does this limitation, and worded in this way, make sense?

# Key limits (2/3)

- Increased value of ‘l’ (message size in blocks) for algos except AES_128_CCM_8
  - Increasing ‘l’ from 2^8 to 2^10 seems to maintain secure CA and IA probabilities
  - draft-irtf-cfrg-aead-limits mentions aiming for CA & IA lower than to 2^-50
    - They have added a table in that document with calculated ‘q’ and ‘v’ values

q = 2^20, v = 2^20, and l = 2^10

| Algorithm name | IA probability | CA probability |
|---|---|---|
| AEAD_AES_128_CCM | 2^-64 | 2^-66 |
| AEAD_AES_128_GCM | 2^-97 | 2^-89 |
| AEAD_AES_256_GCM | 2^-97 | 2^-89 |
| AEAD_CHACHA20_POLY1305 | 2^-73 | - |

- It there a possibility to increase 'q', 'v' and/or 'l' further?
  - Since we are well below 2^-50 for CA & IA currently

# Key limits (3/3)

- Updated table of ‘q’, ‘v’ and ‘l’ for AES_128_CCM_8
  - Added new value for ‘v’, still leaving CA and IA less than 2^-50
  - Ideal to stick to CA and IA as close to 2^-50 as possible?

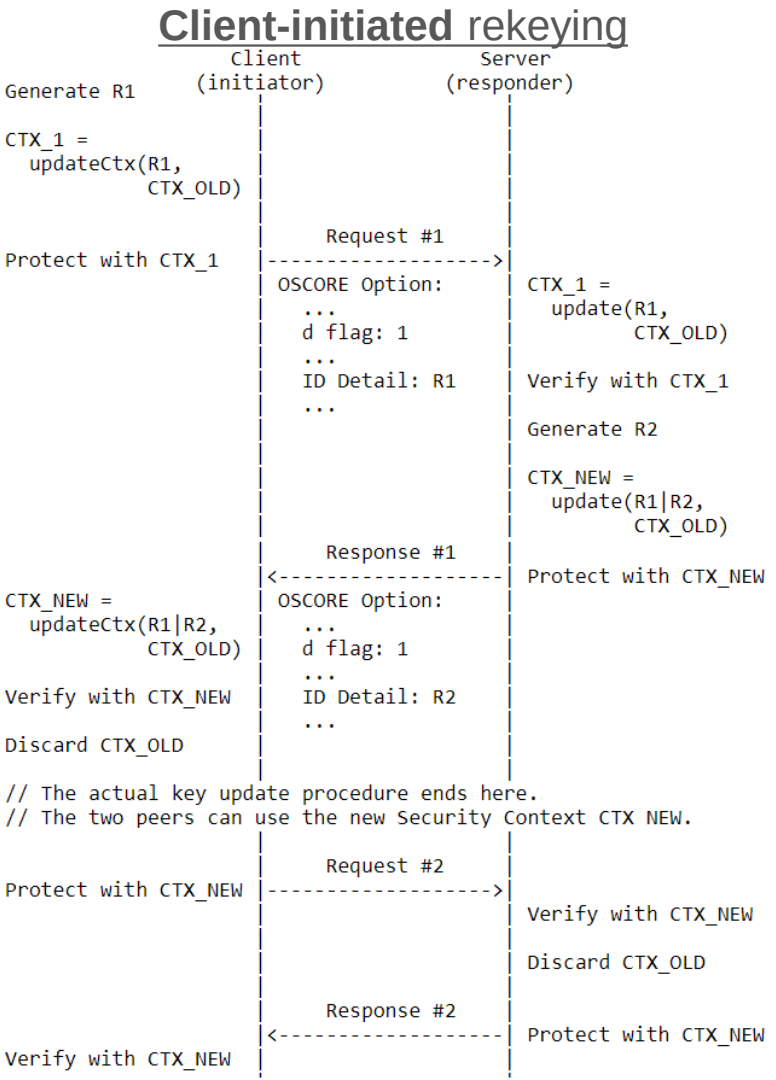
| 'q', 'v' and 'l' | IA probability | CA probability |
|---|---|---|
| q=2^20, v=2^20, l=2^8 | 2^-44 | 2^-70 |
| q=2^15, v=2^20, l=2^8 | 2^-44 | 2^-80 |
| q=2^10, v=2^20, l=2^8 | 2^-44 | 2^-90 |
| q=2^20, v=2^15, l=2^8 | 2^-49 | 2^-70 |
| q=2^15, v=2^15, l=2^8 | 2^-49 | 2^-80 |
| q=2^10, v=2^15, l=2^8 | 2^-49 | 2^-90 |
| q=2^20, v=2^14, l=2^8 | 2^-50 | 2^-70 |
| q=2^15, v=2^14, l=2^8 | 2^-50 | 2^-80 |
| q=2^10, v=2^14, l=2^8 | 2^-50 | 2^-90 |
| q=2^20, v=2^10, l=2^8 | 2^-54 | 2^-70 |
| q=2^15, v=2^10, l=2^8 | 2^-54 | 2^-80 |
| q=2^10, v=2^10, l=2^8 | 2^-54 | 2^-90 |

| 'q', 'v' and 'l' | IA probability | CA probability |
|---|---|---|
| q=2^20, v=2^20, l=2^6 | 2^-44 | 2^-74 |
| q=2^15, v=2^20, l=2^6 | 2^-44 | 2^-84 |
| q=2^10, v=2^20, l=2^6 | 2^-44 | 2^-94 |
| q=2^20, v=2^15, l=2^6 | 2^-49 | 2^-74 |
| q=2^15, v=2^15, l=2^6 | 2^-49 | 2^-84 |
| q=2^10, v=2^15, l=2^6 | 2^-49 | 2^-94 |
| q=2^20, v=2^14, l=2^6 | 2^-50 | 2^-74 |
| q=2^15, v=2^14, l=2^6 | 2^-50 | 2^-84 |
| q=2^10, v=2^14, l=2^6 | 2^-50 | 2^-94 |
| q=2^20, v=2^10, l=2^6 | 2^-54 | 2^-74 |
| q=2^15, v=2^10, l=2^6 | 2^-54 | 2^-84 |
| q=2^10, v=2^10, l=2^6 | 2^-54 | 2^-94 |

# Key update (1/6)

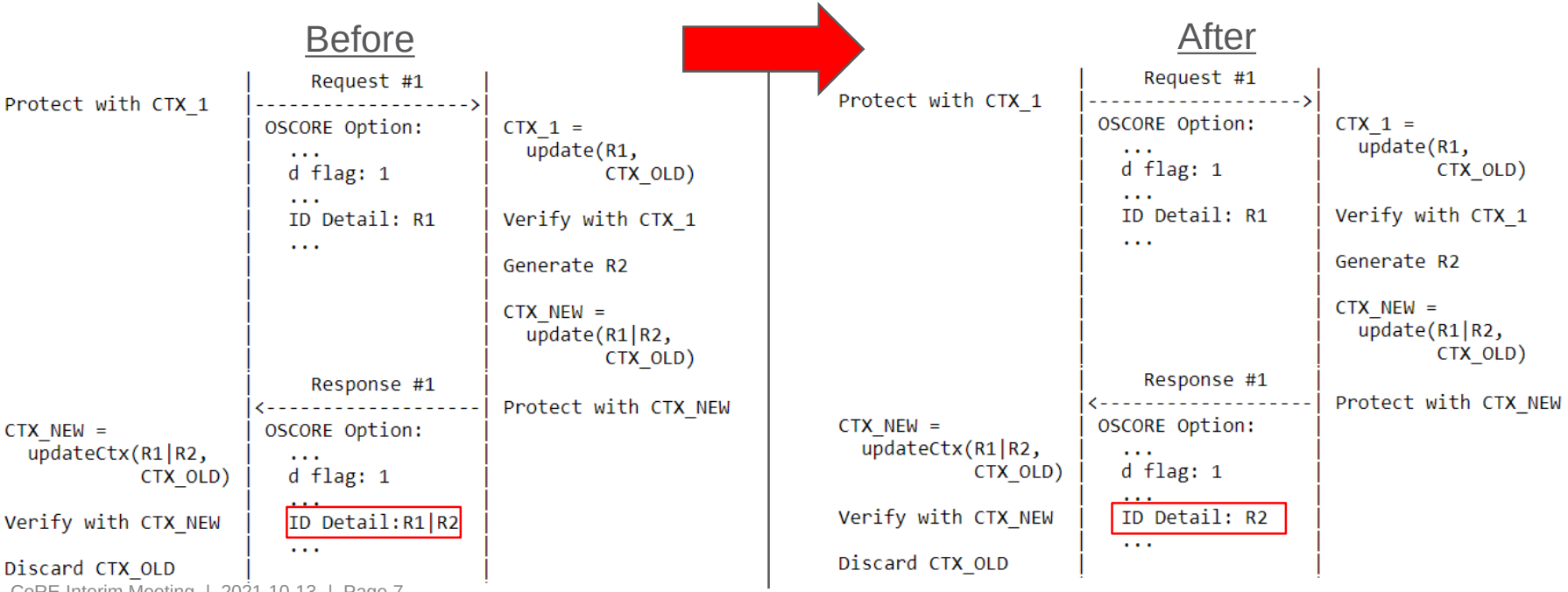
- Defined a new method for rekeying OSCORE
  - Client and server exchange two nonces R1 and R2
  - *UpdateCtx()* function for deriving new OSCORE Security Context using the nonces
  - Current Sec Ctx (to renew) => Intermediate Sec Ctx ==> **New Sec Ctx**

- Properties
  - Robust and secure against peer rebooting
  - Completes in one round-trip (after that, the new Security Context can be used)
  - Compatible with prior key establishment through the EDHOC protocol
  - Only one intermediate Security Context is derived
  - The ID Context does not change
  - Can be initiated by either the client or server



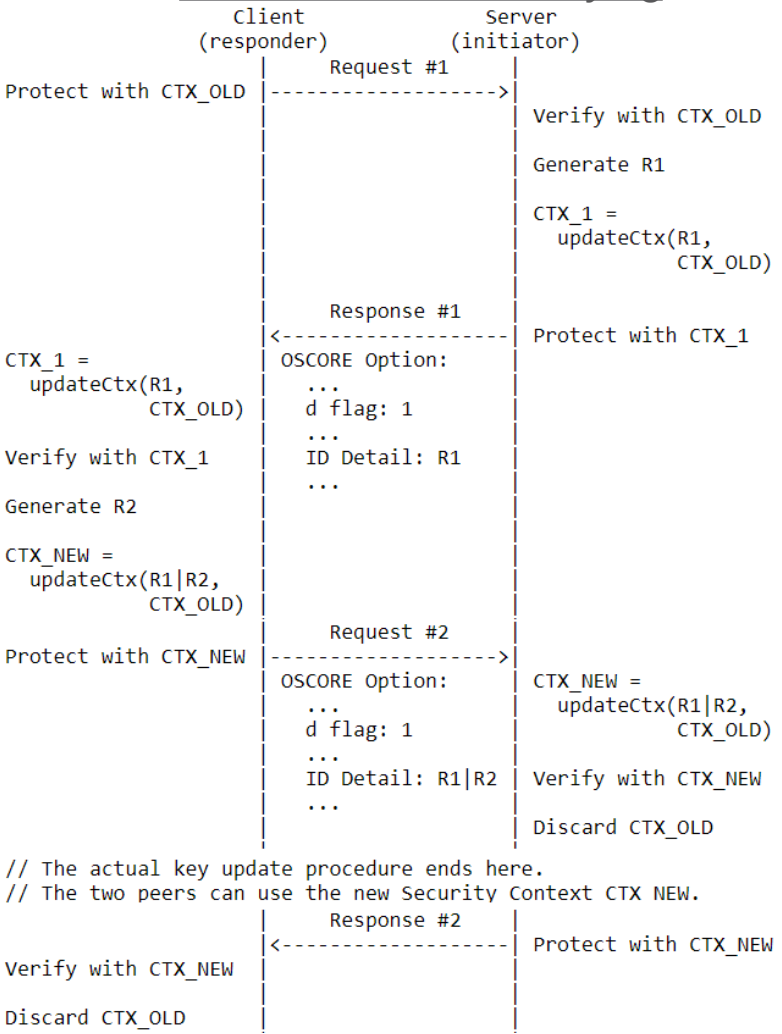
# Key update (2/6)

- No more R1 in the Response #1 for the **client-initiated** rekeying
  - Just like in OSCORE Appendix B.2
  - Simply not needed: Response #1 correlates to Request #1 through the CoAP Token



# Key update (3/6)

- Clarification on the Request #2 processing for the **server-initiated** rekeying
  - Just like in OSCORE Appendix B.2
  - Recognize R1 as sent in a previous Response #1
  - Recognize R1 | R2 as never received in a Request #2
  - Also need to add further text on generation/storage of R2 (similar to that in OSCORE Appendix B.2)



# Key update (4/6)

- Recommendations on minimum length of R1 and R2 values
  - R1 and R1 | R2 are used as nonces
  - Motivation is based on similar considerations for Appendix B.2 in RFC8613
  - We now recommend minimum 8 bytes, is this sufficient?
  - Further text needs to be added as in Appendix B.2. e.g. mentioning the birthday paradox

- Now MUST terminate ongoing observations after rekeying (derived CTX_NEW)
  - Possible to keep them ongoing by paying a price, i.e. admitting a sooner use of large Partial IVs
  - Possible solution: after a rekeying, the client considers PIV* as the highest req_piv among all the ongoing observations. Then, when the client starts the first new observation, the SSN jumps to PIV*+1, thus every observation request has a PIV greater than PIV*.
  - Drawback: Big jumps in PIV, i.e., faster consumption and larger communication overhead
  - Is it worth keeping observations ongoing across a rekeying?

# Key update (5/6)

- Align with EDHOC-Exporter interface, based on EDHOC v -11
  - Used correct labels as text strings
  - Empty CBOR byte string as context, i.e. h' ' (0x40)
  - New usage:

```
MSECRET_NEW = EDHOC-Exporter("OSCORE_Master_Secret", h'', key_length)
MSALT_NEW = EDHOC-Exporter("OSCORE_Master_Salt", h'', salt_length)
```

- A peer using EDHOC and using this OSCORE rekeying procedure …
  - … MUST support EDHOC-KeyUpdate() …
  - Which otherwise SHOULD support as per the EDHOC draft
  - OK with this?

# Key update (6/6)

- Added and discussed 6TiSCH as use case
  - 6TiSCH uses OSCORE Appendix B.2 to handle failure events
  - If the 6TiSCH JRC severely fails, it can use Appendix B.2 with the pledges (RECOMMENDED)
  - The new key update procedure is a good replacement, especially for 6TiSCH
  - Among its intrinsic advantages compared to Appendix B.2, it preserves the ID Context across rekeying
    - 6TiSCH uses ID Context as pledge identifier, meaning that:
    - → A key update would not change pledge identifier, which remains unchanged in the long run
    - → The JRC does not need anymore to do a remapping between new ID Context and pledge identifier
    - → ID Contexts and pledge identifiers can be used as intended at setup/deploy time

- The update to RFC8613 includes also “deprecating and replacing” its Appendix B.2
  - OK with superseding OSCORE Appendix B.2 per se?
  - OK with the wording “deprecating and replacing” ?

# More general updates

- Improved Table of Content structure
  - Key Limits
  - Current rekeying methods
  - New rekeying methods
    - Building blocks
    - Client-initiated procedure
    - Server initiated procedure
    - Policies
    - Discussion
- Editorial improvements
  - Terminology harmonization
  - Use of RFC8126 terminology in IANA considerations
- Should the rekeying procedure have an actual name for easier reference?

# Next steps

- Address open points
  - Continued work on open issues tracked on GitLab repo
  - Further refinement of limits

- Comments received during meeting or mailing list

- Submission of new draft version before the IETF 112 cut-off

# Thank you!

# Comments/questions?

https://gitlab.com/rikard-sics/draft-hoeglund-oscore-rekeying-limits/