Key Update for OSCORE (KUDOS)

draft-ietf-core-oscore-key-update-07

Rikard Höglund, RISE
Marco Tiloca, RISE

IETF CoRE WG meeting – IETF 119 – March 20th, 2024

Recap

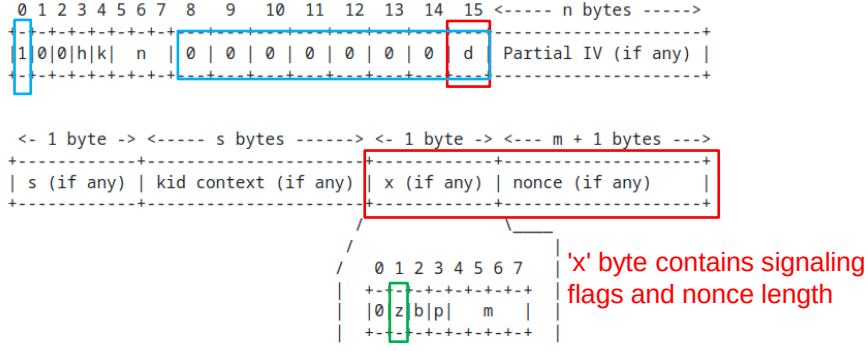
- (1) Key Update for OSCORE (KUDOS)
  - Renew the Master Secret and Master Salt; derive new Sender/Recipient keys
  - No change to the ID Context; can achieve Perfect Forward Secrecy
  - Agnostic of the key establishment method originally used
  - Loosely inspired by Appendix B.2 of OSCORE

- (2) AEAD Key Usage Limits in OSCORE
  - Was split out as a separate draft as of March 2023: *draft-ietf-core-oscore-key-limits*

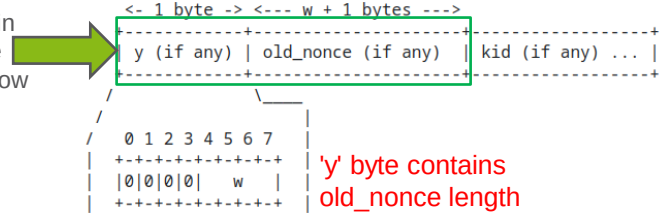
- (3) Procedure for updating OSCORE Sender/Recipient IDs
  - Agreed during IETF 118 to split out as a separate draft
  - Has now been split out and submitted as a separate draft: *draft-ietf-core-oscore-id-update*

Rekeying procedure

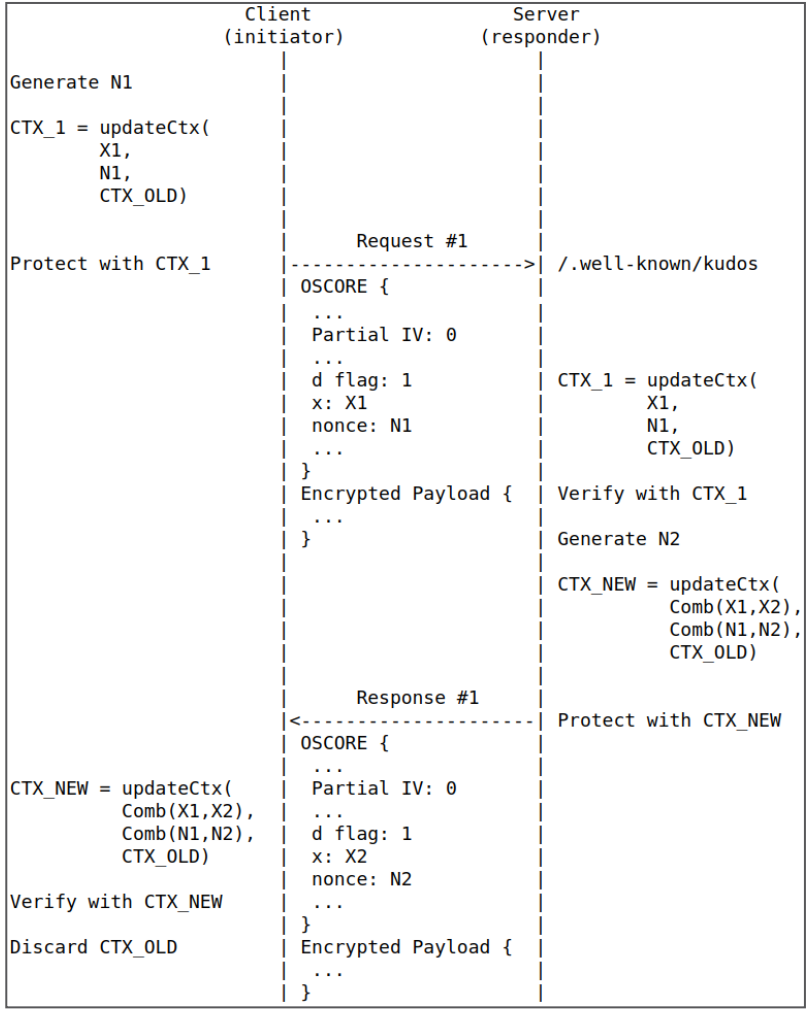
- **Key Update for OSCORE (KUDOS)**
  - Message exchange to share two nonces N1 and N2
  - Nonces are placed in new fields in OSCORE CoAP option
  - *UpdateCtx()* function for deriving new OSCORE Security Context using the two nonces and two ‘x’ bytes
  - Extended OSCORE Option



Only used in the reverse message flow



KUDOS forward message flow



Updates Since IETF 118 Overview

- **Thanks to Christian Amsüss for multiple points of feedback**
- **Submitted version -07 before the cut-off for IETF 119**

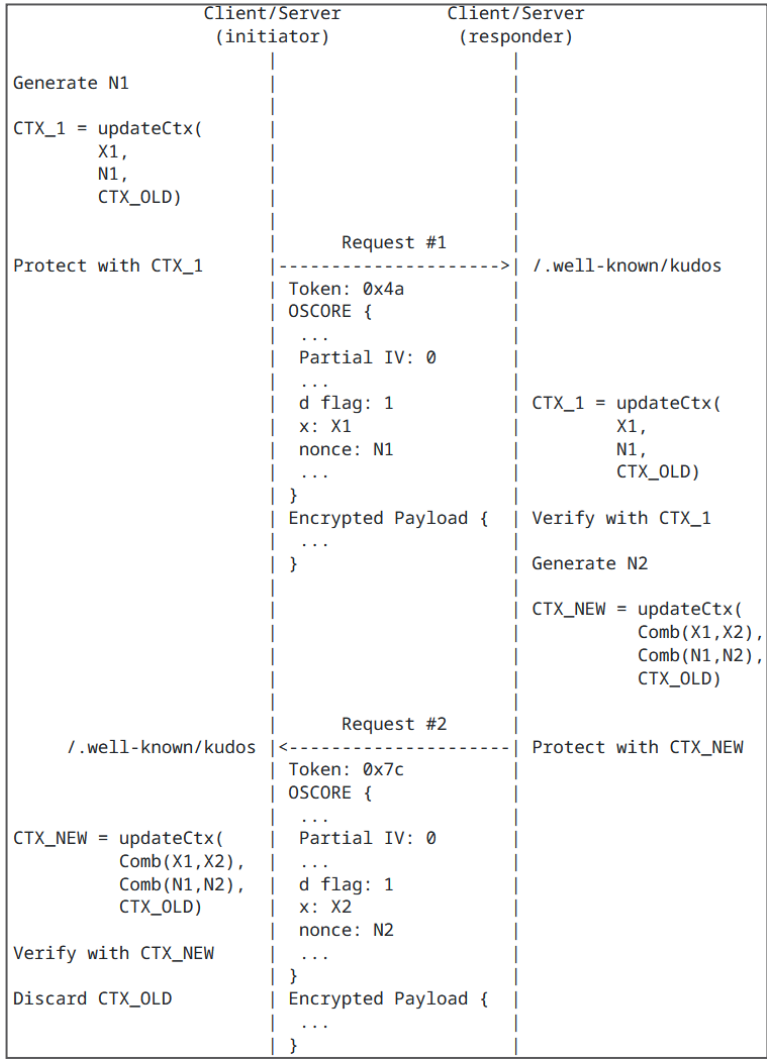
- **Summary of updates**
  - Allow non-random nonces for CAPABLE devices
  - Permit flexible message pair with KUDOS messages as any request/response
  - Enable sending KUDOS messages as regular application messages
  - Removed material about the ID update procedure
    - Has been split out into a separate draft.
  - Editorial improvements

Allowing use of Non-random Nonces

- **Previously we always referred to the nonces as random values**
  - However, in some scenarios using counters makes sense
  - CAPABLE devices can persist the context over a reboot, hence they may use a nonce counter while ensuring to not reuse them
- **The draft now explicitly allows usage of counters as nonces**
  - Recommending the same size as the nonces, that is the use of 8 byte nonce values is RECOMMENDED
- **Solution**
  - Non-CAPABLE devices: MUST use random values
  - CAPABLE devices: MAY use a value obtained from a monotonically incremented counter
    - As long as it is ensured that counter values are not reused, e.g. as is done for the Sender Sequence Number in RFC8613 Appendix B.1.1
- **Privacy Considerations**
  - Random nonces is preferable for maintaining privacy
  - Using counters will reveal the frequency of rekeying procedures performed

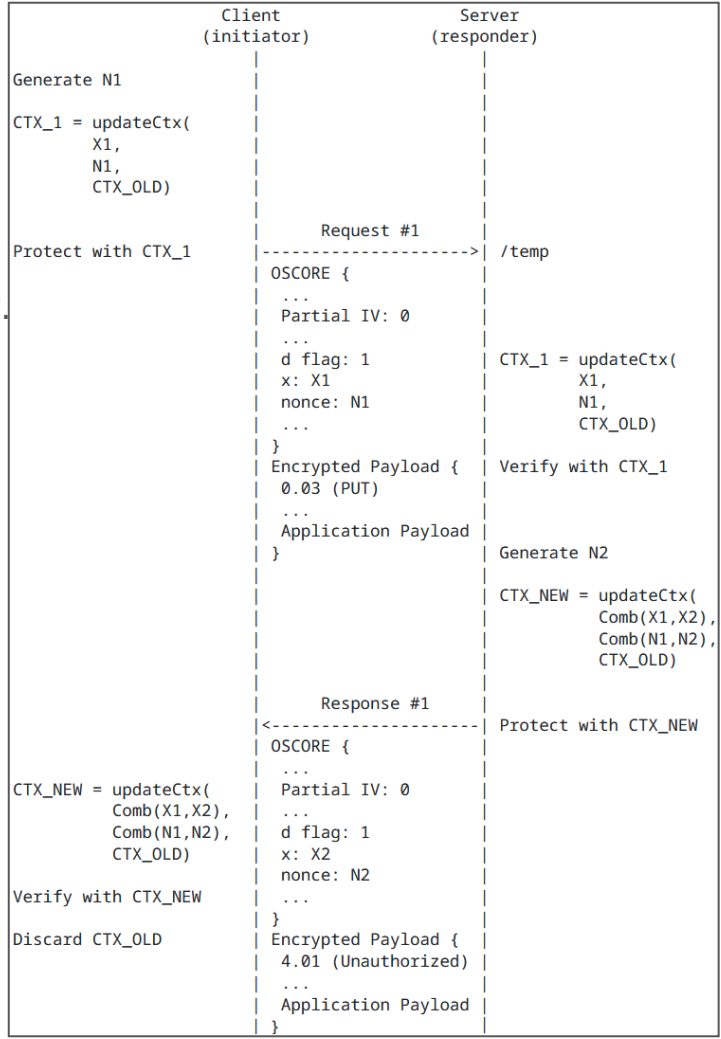
Flexible Message Pair

- **An execution of KUDOS does not need to be a request/response message pair**
  - Instead, more flexible messages flows can be allowed (e.g., two CoAP requests)
- **E.g., a scenario using the Resource Directory where both KUDOS messages are requests**
- **Other alternatives are also possible**
  - Second KUDOS message is a response to a different request than the first KUDOS message
  - Could be the case where there are ongoing observations between the peers



KUDOS Messages as Regular Application Messages

- **Allow the client to initiate KUDOS with a 'normal' application message**
  - The client wants to send an application request to the server. Thus, this message also serves as a KUDOS message.
  - Practically KUDOS request messages can target any resource at the server:
    - In the forward message flow, the client sends the application message that it currently wants to send as a KUDOS message
    - The server cannot be sure the request is fresh, thus if it requires freshness it MUST respond with a protected 4.01 response.
    - Then the client re-sends a non-KUDOS request protected with CTX_NEW, typically with the same content as the first request.
- **The *./.well-known/kudos* resource can still be used**
  - For instance, if the client does not want to send any application request currently
  - In this case, the CoAP request composed before OSCORE protection should not include an application payload



Update of Sender/Recipient IDs

- Method for updating peers' OSCORE Sender/Recipient IDs
  - Based on earlier discussions on the mailing list [1][2] and on [3]
  - This procedure can be embedded in a KUDOS execution or run standalone
  - This procedure can be initiated by a client or by a server
  - As agreed during IETF 118 content has been split out into *draft-ietf-core-oscore-id-update*

No.	C	U	N	R	Name	Format	Length	Default
TBD24					Recipient-ID	opaque	any	(none)

C=Critical, U=Unsafe, N=NoCacheKey, R=Repeatable

- Properties
  - The sender indicates its new wished Recipient ID in the new Recipient-ID Option (class E)
  - Both peers have to opt-in and agree in order for the IDs to be updated
  - Changing IDs practically triggers derivation of new OSCORE Security Context
  - Must not be done immediately following a reboot if run standalone (e.g., KUDOS must be run first)
  - Offered Recipient ID must be not used yet under (Master Secret, Master Salt, ID Context)
  - Received Recipient ID must not be used yet as own Sender ID under the same triple
- Examples are provided in Sections 2.1.1 and 2.1.2

[1] https://mailarchive.ietf.org/arch/msg/core/GXsKO4wKdt3RTZnQZxOzRdIG9QI/
[2] https://mailarchive.ietf.org/arch/msg/core/ClwcSF0BUVxDas8BpgT0WY1yQrY/
[3] https://github.com/core-wg/oscore/issues/263#issue-946989659

Summary and next steps

- **Related point on OSCORE limits document**
  - Submitted new version –02 in January
  - Waiting for updates to *cfrg-aead-limits* and possible feedback

- **Proceed with work on open issues**
  - All are documented on the draft Github repository
  - https://github.com/core-wg/oscore-key-update/issues

- **Implementation**
  - Continue with implementation work in Java and C

- **Comments and reviews are welcome!**

Thank you!

Comments/questions?

https://github.com/core-wg/oscore-key-update

https://github.com/core-wg/oscore-id-update

Backup

Key Limits Overview

- **Working group document**
  - Content split out from *Key Update for OSCORE (KUDOS)* (draft-ietf-core-oscore-key-update)
  - Discussed during previous core interim on 2022-09-28 [1]
  - Also discussed and confirmed during IETF 115 [2]

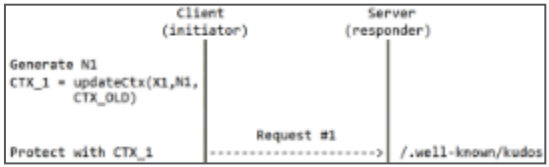
- **Content of the draft: AEAD Key Usage Limits in OSCORE**
  - Excessive use of the same key can enable breaking security properties of the AEAD algorithm*
  - Defining appropriate limits for OSCORE, for a variety of algorithms
  - Defining counters for key usage; message processing details; steps when limits are reached

[1] https://datatracker.ietf.org/meeting/interim-2022-core-13/session/core
[2] https://datatracker.ietf.org/meeting/115/session/core

*See also *draft-irtf-cfrg-aead-limits*

KUDOS Request target

- **What resource should KUDOS Requests target?**
  - Should the client target a KUDOS resource, or just any resource?



Forward message flow

- **Option 1**
  - The client must send Requests to a dedicated KUDOS resource (that doesn't produce a payload or act on requests).
  - Downside: This may require that the KUDOS resource interacts with methods within the OSCORE-related code. Alternatively, the OSCORE-related code can be aware of which resources are "KUDOS resources".

- **Option 2 (**like in OSCORE Appendix B.2**)**
  - The client's Requests can target any resource (existing or not)
  - The server cannot act on this request (in the forward flow)
  - The client must ignore any payload in KUDOS Responses.
  - Downside: Likely requires modifications to the OSCORE library itself, not sufficient to just implement a new standalone resource