Key Update for OSCORE (KUDOS)

draft-ietf-core-oscore-key-update-09

Rikard Höglund, RISE
Marco Tiloca, RISE

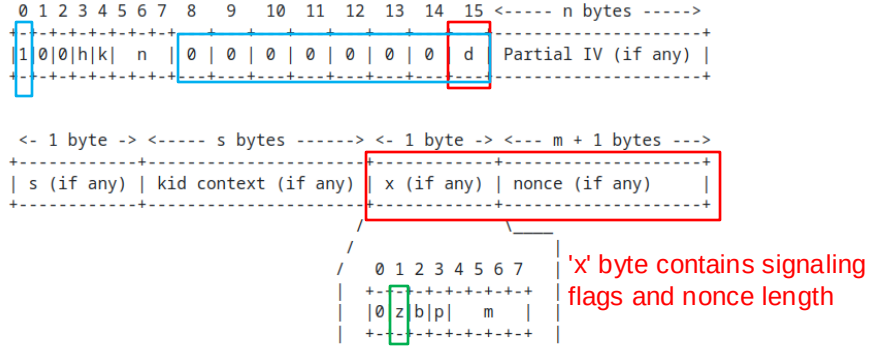
IETF CoRE WG meeting – IETF 121 – November 5th, 2024

Recap

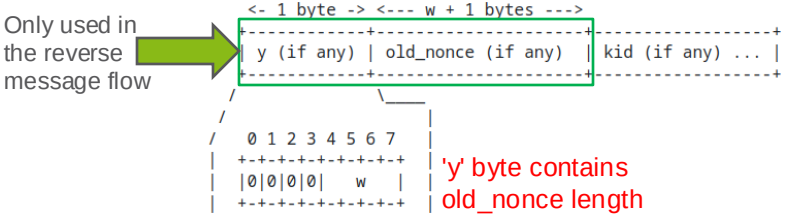
- (1) Key Update for OSCORE (KUDOS)
  - Renew the Master Secret and Master Salt; derive new Sender/Recipient keys
  - No change to the ID Context; can achieve Perfect Forward Secrecy
  - Agnostic of the key establishment method originally used
  - Loosely inspired by Appendix B.2 of OSCORE
- (2) AEAD 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
  - Was split out as a separate draft as of March 2024: *draft-ietf-core-oscore-id-update*

Rekeying Procedure

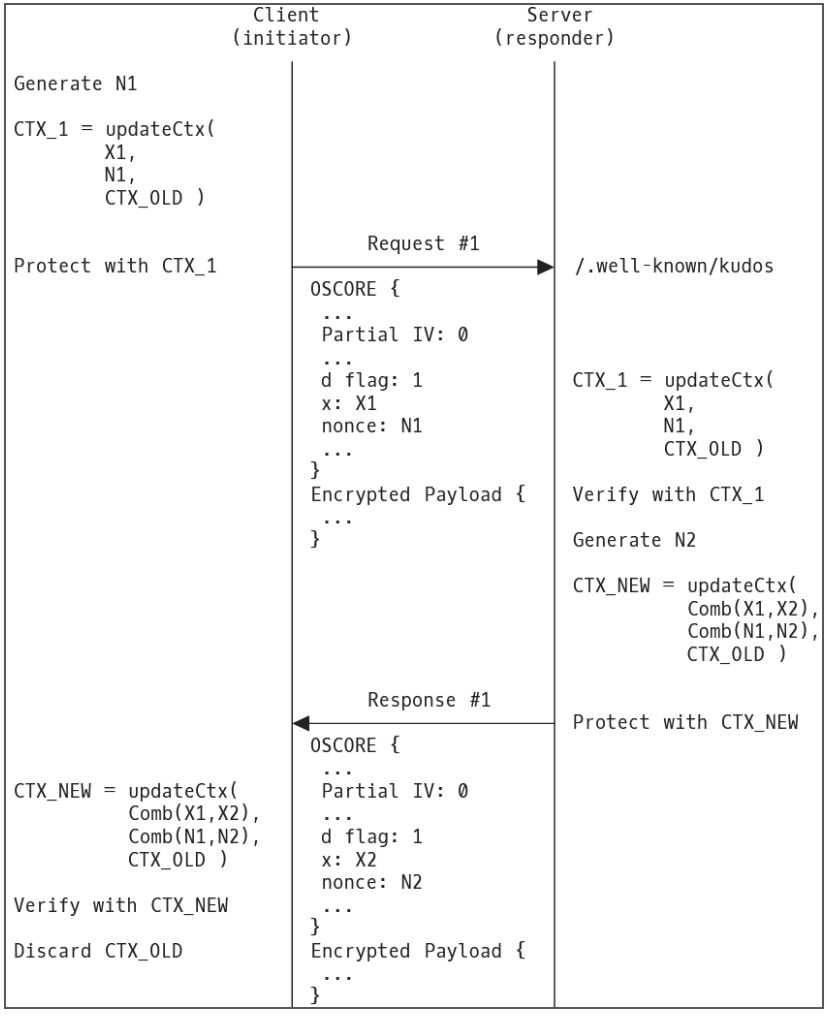
- **Key Update for OSCORE (KUDOS)**
  - Message exchange to share two nonces N1 and N2
  - Nonces are placed in new fields in OSCORE CoAP option
  - *UpdateCtx()* function for deriving new OSCORE Security Context using the two nonces and two ‘x’ bytes
  - Extended OSCORE Option



Only used in the reverse message flow



KUDOS forward message flow



Updates to v-09 (1/5)

- **Updated KUDOS communication overhead values and description**
  - Clarified starting assumptions
    - Nonces in the 2 KUDOS message are of same size, 1 byte PIV, requests have 1 byte KID, responses have no KID
  - Take into account possible presence of the 1-byte Option Length (extended) field

Nonce size	First KUDOS message	Second KUDOS message	Total
1	3	5	8
8	11	12	23
16	19	21	40

Table 1: Communication overhead (forward message flow)

Nonce size	First KUDOS message	Second KUDOS message	Total
1	5	5	10
8	12	20	32
16	21	36	57

Table 2: Communication overhead (reverse message flow)

- **Considered adding new parameter to signal KUDOS support when using the ACE OSCORE profile**
  - However, we concluded that usage of KUDOS is not compatible with the OSCORE ACE profile since the Master Secret (PoP key in the Access Token) is changed
  - Will add text covering usage of EDHOC OSCORE ACE profile

Updates to v-09 (2/5)

- **Introduce special case when non-CAPABLE devices may operate in FS Mode**
  - Devices may not be pre-provisioned with Bootstrap material
    - E.g. due to storage limitations of persistent memory or lack thereof
  - Normally, a non-CAPABLE device always uses KUDOS in no-FS mode
    - **An exception is introduced**: If Bootstrap material is dynamically installed at the device through an in-band process (e.g. EDHOC ACE profile)
  - Description of workflow for a peer A running KUDOS with peer B
    - Peer A is not provisioned with Bootstrap material associated with peer B at the time of commissioning
    - Peer A establishes OSCORE keying material associated with peer B through an in-band procedure, storing that keying material in volatile memory as the Latest material
    - Peer A runs KUDOS in FS mode with peer B and will store the newly derived Security Context in volatile memory
    - As long as peer A does not reboot, executions of KUDOS use the Latest material from volatile memory
    - If peer A reboots, no OSCORE keying material associated with the peer B will be retained, and the in-band procedure needs to be repeated

Updates to v-09 (3/5)

- **Clarifications about achieving key confirmation and deletion of old contexts**
  - Improved clarity of language by more clearly describing the relations between contexts
  - Clarified when key confirmation is reached when running KUDOS twice in sequence before getting key confirmation for CTX_NEW
    - ...the server will achieve key confirmation of CTX_NEW, upon successfully verifying the first KUDOS message as that CTX_NEW is used to derive CTX_1 in this new execution of KUDOS

- **Warn of consequences of running KUDOS with insufficient margin**
  - Example events that may trigger the need to do key update: The OSCORE Security Context expiration or the exhaustion of the Sender Sequence Number space
  - If the rekeying is not initiated ahead of such events, it may become practically impossible to perform a key update **without aborting ongoing message exchanges**
  - For use cases with very long-lived exchanges this is an important point

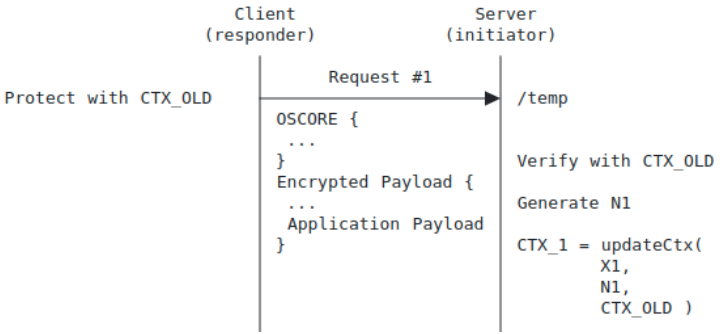
Updates to v-09 (4/5)

- **Usefulness of core.kudos resource type for KUDOS requests without side effects**
  - A resource of this type does not pertain to any real application
  - No application-level actions are triggered as a result of the KUDOS request
  - Allows clients to run KUDOS being fully certain that no application-layer processing is intended to occur on the server

- **Editorial improvements and clarifications**

- **Content restructuring**
  - Merged separate sections about avoiding in-transit requests during KUDOS execution
  - Clarify that Section 4 describes the main functionality of KUDOS itself, although in FS mode
    - Section 5 is a delta, and focuses on using the no-FS mode

Updates to v-09 (5/5)



- **Enable pure CoAP servers to use KUDOS in reverse flow in corner case**
  - Consider peers that are only capable of acting as CoAP servers
  - If such a peer reaches key usage limits for its Recipient Key
    - ...it cannot decrypt Request #1 to be able to initiate the KUDOS reverse message flow
  - An option is for the client to run KUDOS in the forward flow
    - … but there is no means for client to know this
  - Solution: Allow the server to send KUDOS Response #1 (to initiate KUDOS in reverse flow), without decrypting Request #1
  - Optional modified steps for OSCORE processing of incoming requests
    - *If the retrieved Recipient Context is invalid, the server MAY respond with a 4.01 (Unauthorized) error message …. If the error message is a KUDOS Response #1, then it is protected with the OSCORE Security Context CTX_1 derived from the Security Context CTX_OLD.*

Key Usage Limits & ID Update

- **ID Update (draft-ietf-core-oscore-id-update)**
  - **Recap**: Method for updating peers' OSCORE Sender/Recipient IDs. Can be embedded in a KUDOS execution or run standalone, and be initiated by a client or by a server
  - Next steps
    - Add more examples including failure cases
    - Restructure text to split long sections for readability

- **Key Usage Limits (draft-ietf-core-oscore-key-limits)**
  - **Recap**: OSCORE-specific safe limits for Sender/Recipient Key usage
    - Safe number of encryptions using the Sender Key
    - Safe number of failed decryptions using the Recipient Key
  - Next steps: Following developments in the related document from CFRG
    - https://datatracker.ietf.org/doc/draft-irtf-cfrg-aead-limits/

Summary and next steps

- **Process remaining points from KUDOS review by Christian Amsüss - Thanks!**
  - See mail: https://mailarchive.ietf.org/arch/msg/core/QGS8QfeySlrTKYRvFnEH7laxBDk/
  - Possibility of generalizations relying on bormann-core-responses (non-traditional responses)
  - Different mechanism for preserving observations
  - Possible simplification of forward flow and reverse flow concepts, and avoiding deadlocks in a more simple manner

- **KUDOS implementations (tested with each other)**
  - Implementation in Java supporting the forward message and reverse message flows [1]
  - Implementation in C for Contiki-NG fully supporting the forward message flow

- **Comments and reviews are welcome!**

[1] https://github.com/rikard-sics/californium/blob/oscore_cmd/cf-oscore/

Thank you!

Comments/questions?

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

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

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

Backup

Key Usage 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*

Update of Sender/Recipient IDs

- **Recap: Method for updating peers' OSCORE Sender/Recipient IDs**
  - This procedure can be embedded in a KUDOS execution or run standalone
  - This procedure can be initiated by a client or by a server

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

Table 1: The Recipient-ID Option.
C=Critical, U=Unsafe, N=NoCacheKey, R=Repeatable

- **Properties**
  - The message 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 not be used yet under the same (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**