Re-keying Mechanisms for Symmetric Keys

draft-irtf-cfrg-re-keying

Stanislav V. Smyshlyaev, Ph.D.
Head of Information Security Department, CryptoPro LLC

draft-irtf-cfrg-re-keying-08

- Re-keying Mechanisms for Symmetric Keys, S. Smyshlyaev, Ed.
- Main contributors:
  - Evgeny Alekseev
  - Russ Housley
  - Daniel Fox Franke
  - Ekaterina Smyshlyaeva
  - Shay Gueron
- Many thanks for comments and considerations to:
  - Mihir Bellare
  - Scott Fluhrer
  - Dorothy Cooley
  - Yoav Nir
  - Maksim Kollegin
  - Jim Schaad
  - Paul Hoffman
  - Dmitry Belyavsky
  - Dmitry Pichulin
  - Yaron Sheffer

Motivation

Re-keying is needed to increase the lifetime of session keys (when new negotiation is too heavy/undesirable), limited by bounds coming from:

- general combinatorial properties of cipher modes of operation; recent example (3DES, limit = 8 MB) — Sweet32, NIST reaction:

> In response, NIST plans to reduce the maximum amount of plaintext allowed to be encrypted under a single TDEA 3-key bundle from 2^{32} to 2^{20} (64-bit) blocks. This will be announced in the upcoming draft of SP 800-67 Revision 2, and NIST will seek comments on this reduction in the public review of that document.

- estimations of material needed for success of various cryptanalysis methods for a used cipher (linear, algebraic, differential etc.);
- side-channel cryptanalysis methods of block ciphers; recent example (AES, limit $\lesssim$ 160 MB) — "TEMPEST attacks against AES" paper, https://www.fox-it.com:

> Fox-IT and Riscure show how to covertly recover the encryption key from two realistic AES-256 implementations while:
>
> 1. Attacking at a distance of up to 1 *m* (30 *cm* in realistic conditions; "TEMPEST"),
> 2. Using minimal equipment (fits in a jacket pocket, costs less than €200) and
> 3. Needing only a few minutes (5 minutes for 1 *m* and 50 seconds for 30 *cm*).

Example: re-keying in TLS 1.3 — KeyUpdate

Recommended to do KeyUpdate after $\approx 2^{24.5}$ full-size records (AES-GCM).

$$\text{traffic\_secret\_N} = \text{HKDF}_{\text{Expand}}(\text{traffic\_secret\_N} - 1, [\ldots])$$
$$\text{write\_key} = \text{HKDF}_{\text{Expand}}(\text{traffic\_secret\_N}, \text{“key“}, [\ldots])$$

Main objective

To prepare a document with „a menu of choices for developers“ for re-keying mechanisms.

- Secure and efficient procedures, solving the re-keying task in the majority of common cases.
- Rather small redundancy of mechanism set.
- General recommendations and choice principles — when to choose which mechanism.

Scope

The document is about:

- How to deal with limits on key usage against side channel attacks.
- Adding security to the usage of LW ciphers with 32/64-bit blocks.
- Gaining basic PFS regarding segments of encryption process.
- Safety margin against possible future attacks on the used ciphers.

Out of scope

The document is NOT about:

- Solving any post-quantum issues (cf. draft-ietf-ipsecme-qr-ikev2).
- Defining any schemes of key (re)negotiation with exchanging fresh nonces, DH key shares (cf. IKEv2 child-SA/IKE SA rekeying).
- Rules of choosing specific limitations to resist side-channel attacks.
- Methods to prolong life of ciphers already known to be vulnerable.

IETF 97, Seoul, November 2016

- A proposal from the CFRG chairs to create a document with a framework for re-keying.
- CFRG meeting: talk on re-keying, discussion.

IETF 98, Chicago, March 2017

- A CFRG side meeting on re-keying
- One-hour wide discussion of the document, a number of important considerations.

IETF 99, Prague, July 2017

- CFRG meeting: document status update.

draft-irtf-cfrg-re-keying, "Re-keying Mechanisms for Symmetric Keys"

- February 27, 2017 — the -00 version.
- March 7, 2017 — the -01 version: usage recommendations and principles of choice added.
- June 5, 2017 — the -02 version: major revision — most of concerns from Chicago (IETF 98) meeting addressed.
- June 20, 2017 — the -03 version: major revision based on a list of considerations by Russ Housley.
- June 30, 2017 — the -04 version: major revision based on considerations by Shay Gueron and Dmitry Belyavsky.
- July 3, 2017 — the -05 version: minor revision.
- September, 8, 2017 — the -06 version: removed OFB-ACPKM, major revision based on considerations by Dmitry Pichulin.
- October, 6, 2017 — the -07 version: removed both CCM-ACPKM modes; added Key Hierarchy section and test vectors.
- October, 9, 2017 — the -08 version: minor revision.

Summary

- Contains external/internal, parallel/serial, hash-based/cipher-based constructions, with/without master key.
- Security (according to the models of «Increasing the Lifetime of a Key: A Comparative Analysis of the Security of Re-Keying Techniques», M. Abdalla, M. Bellare) — ≈ quadratical increase of key lifetime.

Recent discussions

- The document was sent to Crypto Review Panel in October.
- A review from (Crypto Review Panel member) Yaron Sheffer.
- Discussions about connection of the I-D with related documents in the mailing list and private communication.

Considerations to be resolved

- Defining the cases where the serial mechanisms may be preferred.
  - External serial mechanisms provide additional forward secrecy regarding segments of processed text: in terms of TLS 1.3, compromise of application_traffic_secret_N does not compromise all previous application_traffic_secret_i, i < N.
- Adding a section with similar or related constructions.
- It's worth also mentioning applicability for data-at-rest.
- Relationship with mechanisms involving fresh mixed-in entropy for each re-keying.

Comparison with solutions exploiting additional entropy on re-keying

- A good PRF does not require additional entropy for the quality of keys.
- Mixed-in entropy can increase security: security recovery after a time-limited but complete breach of the system.
  - requires storing master keys in an HSM;
  - additional exchanges with mixed-in entropy must be made;
  - can't be done „on the fly“

  — important, but different task.
- Relying on the additional entropy must be done with deep studying security in various adversary models — in many cases it won't increase security, but may give a false sense of that.

Current state and plans

draft-irtf-cfrg-re-keying, "Re-keying Mechanisms for Symmetric Keys"

The structure, principles, major recommendations and all mechanisms (with security bounds) seem to be negotiated and do not tend to be changed.

- Adding comments about the data-at-rest, about control of re-keying process.
- Comparison of described approaches with mechanisms exploiting additional entropy.
- More references to related work.
- New test vectors.
- Resolving editorial comments given in the reviews.

Plan: to get a version addressing these issues (and the concerns given in the Crypto Panel reviews) by the end of January, 2018.

Thank you for your attention!

Questions?

- Materials, questions, comments:
  - svs@cryptopro.ru

Main decisions about the I-D (IETF 98 side meeting)

Considerations on the scope and aims of re-keying

- Consider the following reasons to use re-keying:
  - additional side channel resistance (against DPA or EMI style attacks);
  - PFS security regarding segments of encryption process;
  - lightweight cryptography, usage of ciphers with 32-bit and 64-bit blocks;
  - additional security against possible future attacks on the used ciphers — as a safety margin. Important notice: This MUST NOT be used as a method to prolong life of ciphers that are already known to be vulnerable.
- To add words that no post-quantum issues can be solved by re-keying.
- To add words about reasons remaining for ciphers with large block sizes (e.g. ChaCha20) — side-channel resistance, PFS, safety margin.

Considerations on the recommendations and guidelines

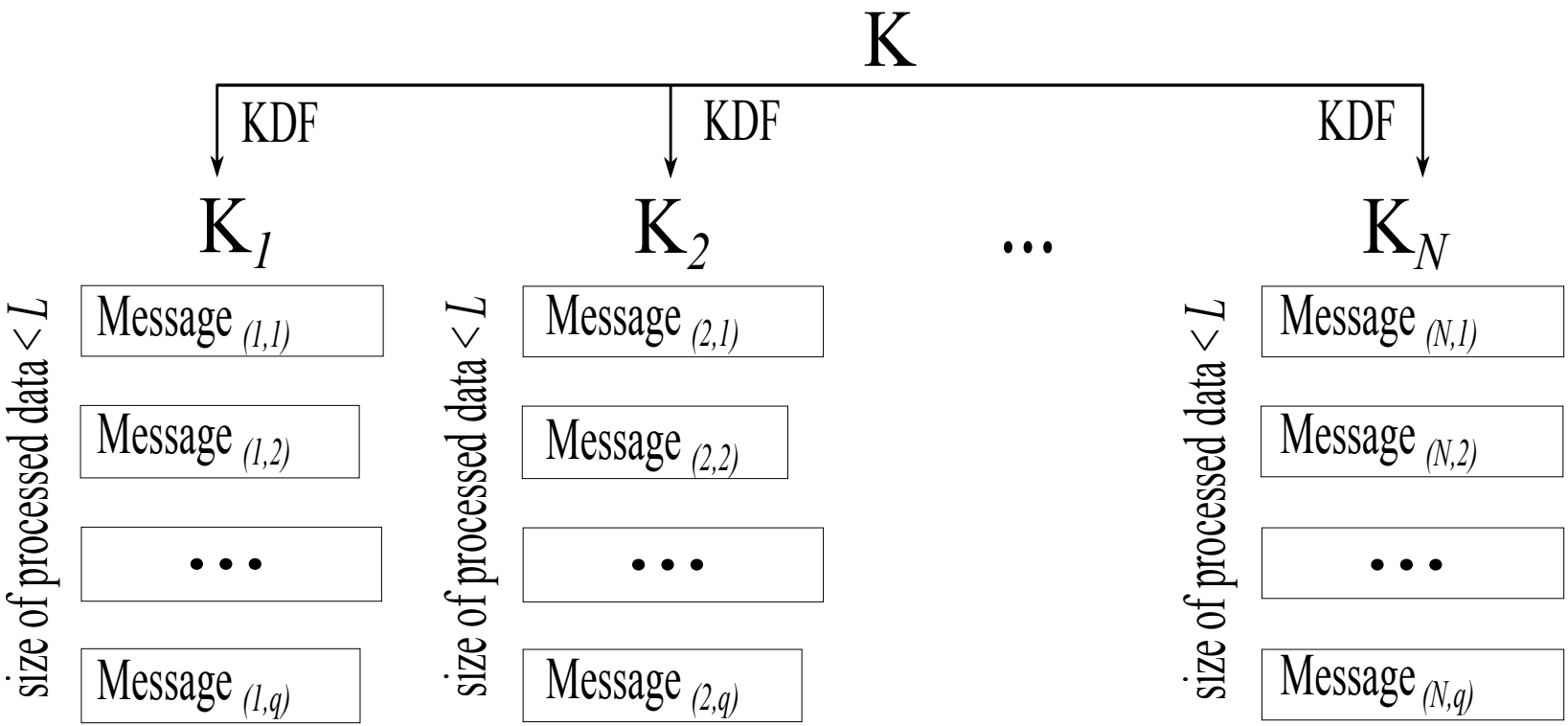
- To base on the following frame: external re-keying is chosen on a protocol level (independently of a block cipher and a block cipher mode), while an internal re-keying is chosen linked to a block size (of a used cipher) and block cipher mode of operation.
- To add a text about advantages and disadvantages of various types of re-keying based on Seoul CFRG (IETF 97) slides.
- To provide sample cases (working examples or toy examples) for choosing one or another type of re-keying for the protocols.
- To change the order of chapters such that the main part of recommendations would be given before the description of specific mechanisms.
- Not to consider related questions for stream ciphers.

Considerations on the mechanisms themselves

- To add explicit text about the principles of choice of constants for internal re-keying CTR mode.
- For the modes: to consider primarily CTR and GCM — but also add CCM and CBC with corresponding comments.
- To add clarifications about advantages and disadvantages of usage of the same primitives for re-keying.
- When choosing constants for internal re-keying, consider only lengths that are multiples of 8.

Key diversification (external re-keying)

- Uses an initial (negotiated) key as a master key, which is never used directly for the encryption but is used for session key derivation.
- A new derived session key (and IV) for each section (of size $\leqslant$ L).



NIST Special Publication 800-108

KDF in Counter Mode

$$K_i = PRF(K, [i]_2|label|0x00|Context|[L]_2)$$

KDF in Feedback Mode

$$K_i = PRF(K, K_{i-1}|[i]_2|label|0x00|Context|[L]_2)$$

KDF in Double-Pipeline Iteration Mode

$$A_i = PRF(K, A_{i-1});$$
$$K_i = PRF(K, A_i|[i]_2|label|0x00|Context|[L]_2)$$

Security analysis

«Increasing the Lifetime of a Key: A Comparative Analysis of the Security of Re-Keying Techniques», M. Abdalla, M. Bellare.

- Parallel variant: $K_i = PRF(K, i)$.
- Serial variant: $K_i = PRF(StateKey_i, 0)$, $StateKey_{i+1} = PRF(StateKey_i, 1)$, $StateKey_1 = K$.

The lifetime of keys drastically increases (independently of the encryption mode) — complete and correct security proofs exist.

Parallel variant: the capacity of the initial key

If we have really hard restrictions on key capacity (e.g., an adversary with powerful side-channel analysis equipment), it can exceed even for the initial („master“) key.

NIST Special Publication 800-108: Key Hierarchy (Key Tree)

An example:

$$\begin{aligned}
\text{Key[i]} = \text{KDF}(& \text{KDF}(& \text{KDF}(& \text{RootKey} & , \text{i\&Mask1}) & , \text{i\&Mask2}) & , \text{i\&Mask3}) \\
& \text{KDF}(& \text{KDF}(& \text{RootKey} & , \text{i\&Mask1}) & , \text{i\&Mask2}) & \\
& & \text{KDF}(& \text{RootKey} & , \text{i\&Mask1}) & & \\
& & & \text{RootKey} & & &
\end{aligned}$$

- Parallel variant: $K_i = PRF(K, i)$.
- Serial variant: $K_i = PRF(StateKey_i, 0)$, $StateKey_{i+1} = PRF(StateKey_i, 1)$, $StateKey_1 = K$.

Advantages

- The material encrypted on each derived key K_i can be strictly limited by L without strict limits on the lifetime of the original key K.
- The adversary cannot combine the information (input-output behaviour, side-channel information...) obtained when observing work on several derived keys.
- The leakage of one derived key K_i does not have any impact on other derived keys.
- The mechanism can be chosen independently of a mode of operation.

- Parallel variant: $K_i = PRF(K, i)$.
- Serial variant: $K_i = PRF(StateKey_i, 0)$, $StateKey_{i+1} = PRF(StateKey_i, 1)$, $StateKey_1 = K$.

Disadvantages

- In both variants $K_1 \neq K$ — thus, we always have to make at least one PRF calculation, even for extremely short plaintexts (the proofs significantly depend on keeping some state (K and $StateKey_i$) unused with the cipher itself).
- An external mechanism: if L is restrictive, inconvenient restrictions on the size of an individual message (with its own header, IV, MAC etc.) appear.

And what if we have chosen some certain mode of operation and want to

1) keep the properties of

- having the strong limits of the section that is explicitly encrypted with any symmetric key;
- impossibility of an adversary to combine the information obtained from different sections

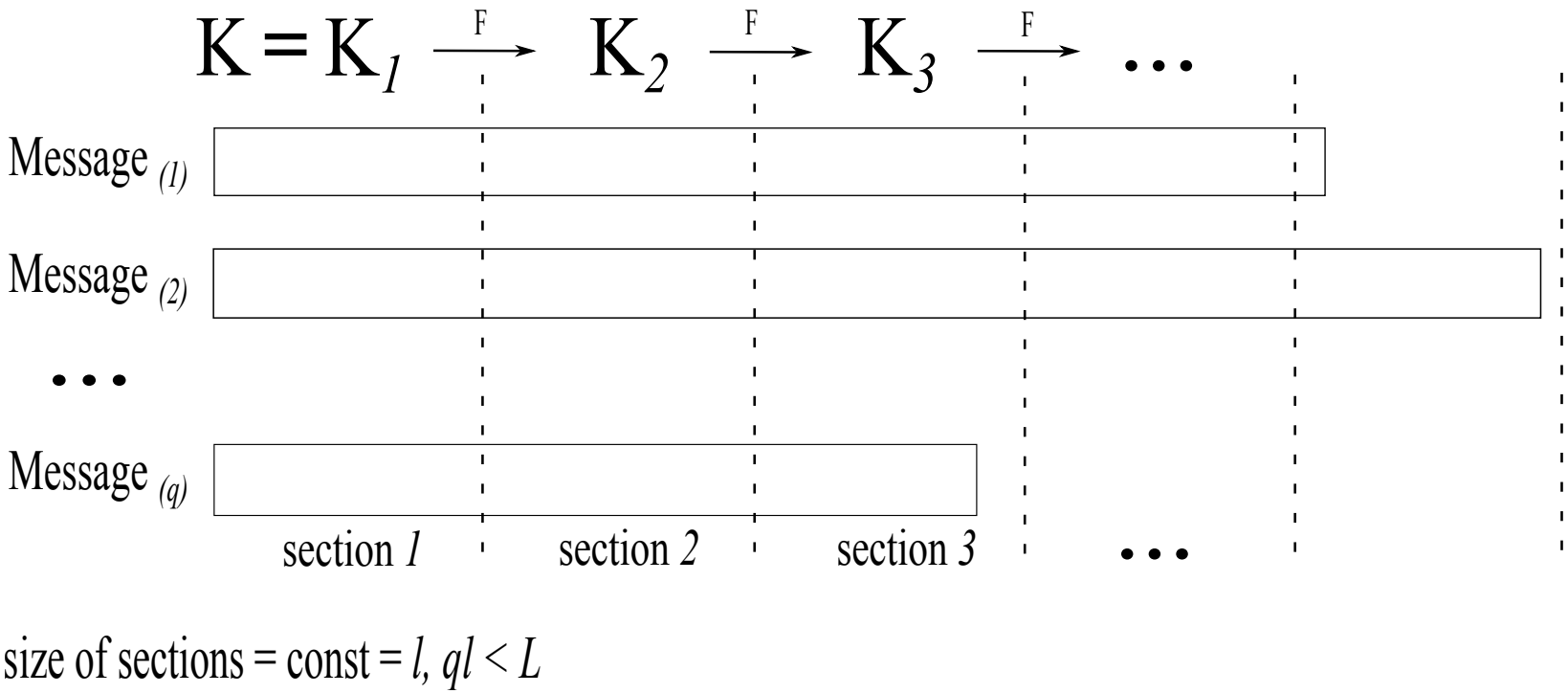
— to limit the possibility of an adversary to study anything about any encryption key by one section;

2) obtain also the properties of

- being efficient on short plaintexts (without any additional operations on such);
- not restarting encryption with new IV's (and MAC calculation) frequently.

Internal re-keying („key meshing“)

- $K_1 = K$, $IV_1 = IV$;
- $(K_{i+1}, IV_{i+1}) = F(K_i, IV_i, i + 1)$.



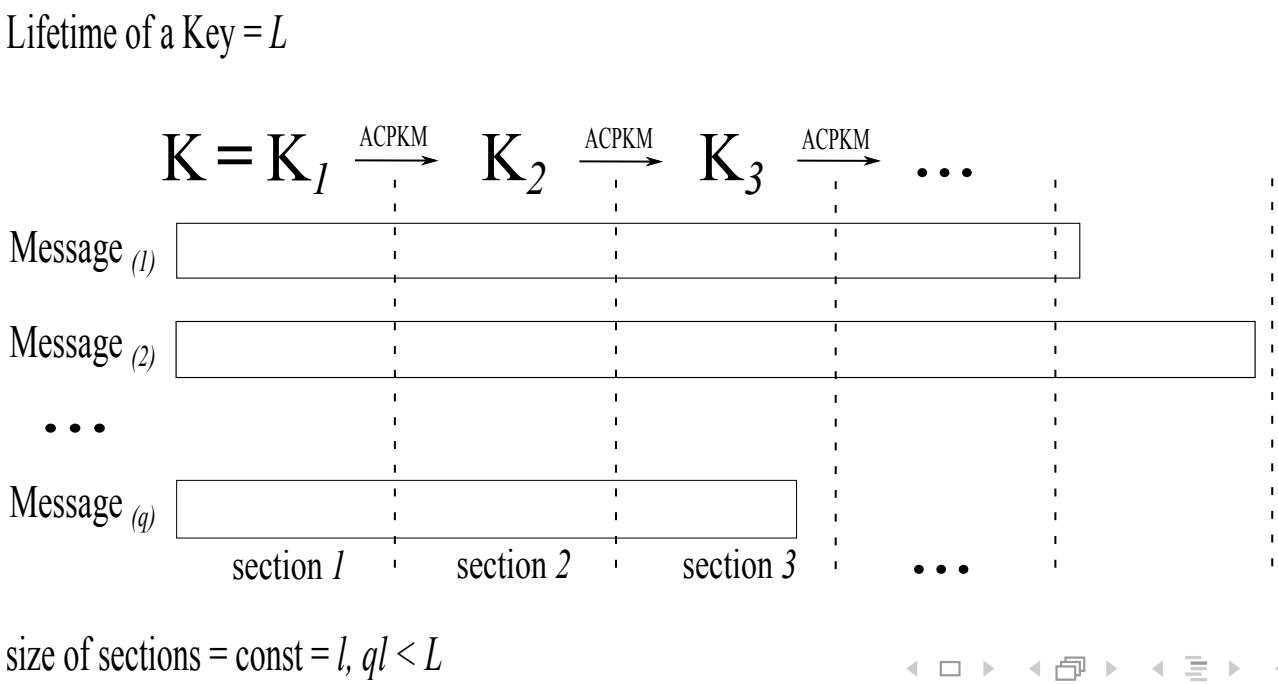
size of sections = const = l, $ql < L$

The proposed method of re-keying („key meshing“)

draft-irtf-cfrg-re-keying

For CTR/GCM with n-bit block, $CTR_i = (ICN \mid counter)$, with c-bit counter value and $(n - c)$-bit ICN.

- $K_{i+1} = ACPKM\text{-}CTR(K_i) = MSB_k(E_{K_i}(W_1)|...|E_{K_i}(W_J))$,
- The section size MUST be less than $2^{c/2-1}$ blocks.



draft-irtf-cfrg-re-keying

For CTR/GCM with n-bit block, $CTR_i = (ICN \mid counter)$, with c-bit counter value and $(n - c)$-bit ICN.

- $K_{i+1} = ACPKM\text{-}CTR(K_i) = MSB_k(E_{K_i}(W_1)|...|E_{K_i}(W_J))$,
- The section size MUST be less than $2^{c/2-1}$ blocks;
- $(n - c + 1)$-th bit of each W_j is 1;
- W_j are defined for any
  - block size n of $64 \leqslant n \leqslant 512$,
  - counter size c of $32 \leqslant c \leqslant \frac{3}{4}n$,
  - key size k of $128 \leqslant k \leqslant 512$.
- W_j are pairwise different fixed constants for all allowed n, c, k.

draft-irtf-cfrg-re-keying

$K_{i+1} = \text{ACPKM-CTR}(K_i) = MSB_k(E_{K_i}(W_1)|...|E_{K_i}(W_J))$.

Disandvantages

- The leakage of one section key K_i will lead to a leakage of all following keys.
- The mechanism must not be chosen independently of a mode of operation.

draft-irtf-cfrg-re-keying

$$K_{i+1} = \text{ACPKM-CTR}(K_i) = \text{MSB}_k(E_{K_i}(W_1)|...|E_{K_i}(W_J)).$$

Advantages (Performance)

- There are no additional unnecessary operations for short plaintexts — if the plaintext length is shorter than the section size, the initial key will be used;
- The plaintext size is not needed to be known in advance — key transformations are made when and only when needed;
- Transparency: no need to restart the encryption process with new IV's (and GHASH calculation).

draft-irtf-cfrg-re-keying

$K_{i+1} = \text{ACPKM-CTR}(K_i) = MSB_k(E_{K_i}(W_1)|...|E_{K_i}(W_J))$.

Advantages (Security)

- The material encrypted on each section key K_i can be strictly limited by L without strict limits on the lifetime of the original key K;
- The adversary cannot combine the information (input-output behaviour, side-channel information...) obtained when observing work on several section keys;
- The total lifetime of an initial key drastically increases.

Really?

draft-irtf-cfrg-re-keying

$K_{i+1} = \text{ACPKM-CTR}(K_i) = MSB_k(E_{K_i}(W_1)|...|E_{K_i}(W_J))$.

Advantages (Security)

- The material encrypted on each section key K_i can be strictly limited by L without strict limits on the lifetime of the original key K;
- The adversary cannot combine the information (input-output behaviour, side-channel information...) obtained when observing work on several section keys;
- The total lifetime of an initial key drastically increases.

Really?

Security model

Cipher mode with internal re-keying is considered as an extension of a base cipher mode of operation, since it affects the process of processing of every single message.

Internal re-keying method must not be considered without specifying cipher mode of operation.

Security models

Security models for block ciphers

- PRF — «Pseudorandom function»;
- PRP-CPA — «Pseudorandom permutation in chosen plaintext attack»;
- PRP-CCA — «Pseudorandom permutation in chosen ciphertext attack».

Security models for cipher modes

- LOR-CPA — «Left Or Right in Chosen Plaintext Attack» (Bellare M., Desai A., Jokipii E., Rogaway P. A Concrete Security Treatment of Symmetric Encryption, 2000).

A security model for the cipher mode (for encryption) — LOR-CPA

An adversary A has access to an oracle $\mathcal{O}^{\mathrm{LOR}}$. Before starting the work the oracle $\mathcal{O}^{\mathrm{LOR}}$ chooses $\mathrm{b} \in_{\mathcal{U}} \{0,1\}$. The adversary A can make requests to the oracle $\mathcal{O}^{\mathrm{LOR}}$. Each of these requests is a pair of strings $(\mathrm{M}^0, \mathrm{M}^1)$, where $|\mathrm{M}^0| = |\mathrm{M}^1|$. In response to the request $(\mathrm{M}^0, \mathrm{M}^1)$ the oracle returns a string C that is a result of the processing of the string M^{b} according to the $\mathcal{SE}$ cipher mode.

Known for CTR

$$\mathrm{Adv}^{\text{LOR-CPA}}_{\mathrm{CTR}}(t, q, m) \leqslant 2 \cdot \mathrm{Adv}^{\mathrm{PRF}}_{\mathrm{E}}(t + q + nqm, qm).$$

Main result for CTR-ACPKM

$$\mathrm{Adv}^{\text{LOR-CPA}}_{\text{CTR-ACPKM}_{\mathrm{E,l}}}(t, q, ml) \leqslant 6m \cdot \mathrm{Adv}^{\text{PRP-CPA}}_{\mathrm{E}}(t + mlqn, ql + s) + \\ + m \cdot \frac{(ql)^2}{2^n} + m \cdot \frac{2sql + s^2 - s}{2^n},$$

where $s = k/n$, l is a section size.

Comparison with CTR

Base assumptions

In case of the block cipher that has no specific methods to decrease the security, the values of adversary's advantages are bounded in the following way:

$$\mathrm{Adv}_{\mathrm{E}}^{\mathrm{PRF}}(t, q) \approx \frac{t}{2^k} + \frac{q^2}{2^n},$$

$$\mathrm{Adv}_{\mathrm{E}}^{\mathrm{PRP\text{-}CPA}}(t, q) \approx \frac{t}{2^k},$$

$$\mathrm{Adv}_{\mathrm{E}}^{\mathrm{PRP\text{-}CCA}}(t, q) \approx \frac{t}{2^k}.$$

Comparison

$$\mathrm{Adv}^{\text{LOR-CPA}}_{\text{CTR}}(\mathrm{t}, \mathrm{q}, \mathrm{m}\ell) \sim \mathrm{m}^2 \cdot \frac{2\mathrm{q}^2\ell^2}{2^{\mathrm{n}}},$$

$$\mathrm{Adv}^{\text{LOR-CPA}}_{\text{CTR-ACPKM}_\ell}(\mathrm{t}, \mathrm{q}, \mathrm{m}\ell) \sim \mathrm{m} \cdot \frac{2\mathrm{q}^2\ell^2}{2^{\mathrm{n}}}.$$

Comparison

$$\mathrm{Adv}^{\text{LOR-CPA}}_{\text{CTR}}(t, q, m\ell) \sim m^2 \cdot \frac{2q^2\ell^2}{2^n},$$

$$\mathrm{Adv}^{\text{LOR-CPA}}_{\text{CTR-ACPKM}_\ell}(t, q, m\ell) \sim m \cdot \frac{2q^2\ell^2}{2^n}.$$

Performance for AES

Does not it reduce speed?

Machine characteristics

Intel Core i5-6500 CPU 3.20GHz, L1 D-Cache 32 KB x 4, L1 I-Cache 32 KB x 4, L2 Cache 256 KB x 4.

Speed of the encryption (OpenSSL) process in the base CTR mode with the hardware supported AES-256 was: 3800 MB/s.

KB	64	128	256	512	1024	2048	4096
MB/s	3700.4	3722.0	3753.7	3765.3	3770.0	3786.5	3795.2
%	2.6	2.1	1.2	0.9	0.8	0.4	0.2

The CTR-ACPKM mode with the AES-256 cipher (hardware support).

Performance for AES

Does not it reduce speed?

Machine characteristics

Intel Core i5-6500 CPU 3.20GHz, L1 D-Cache 32 KB x 4, L1 I-Cache 32 KB x 4, L2 Cache 256 KB x 4.

Speed of the encryption (OpenSSL) process in the base CTR mode with the hardware supported AES-256 was: 3800 MB/s.

KB	64	128	256	512	1024	2048	4096
MB/s	3700.4	3722.0	3753.7	3765.3	3770.0	3786.5	3795.2
%	2.6	2.1	1.2	0.9	0.8	0.4	0.2

The CTR-ACPKM mode with the AES-256 cipher (hardware support).

Speed of the encryption process in the base CTR mode with the hardware supported AES-128 cipher is 5160 MB/s.

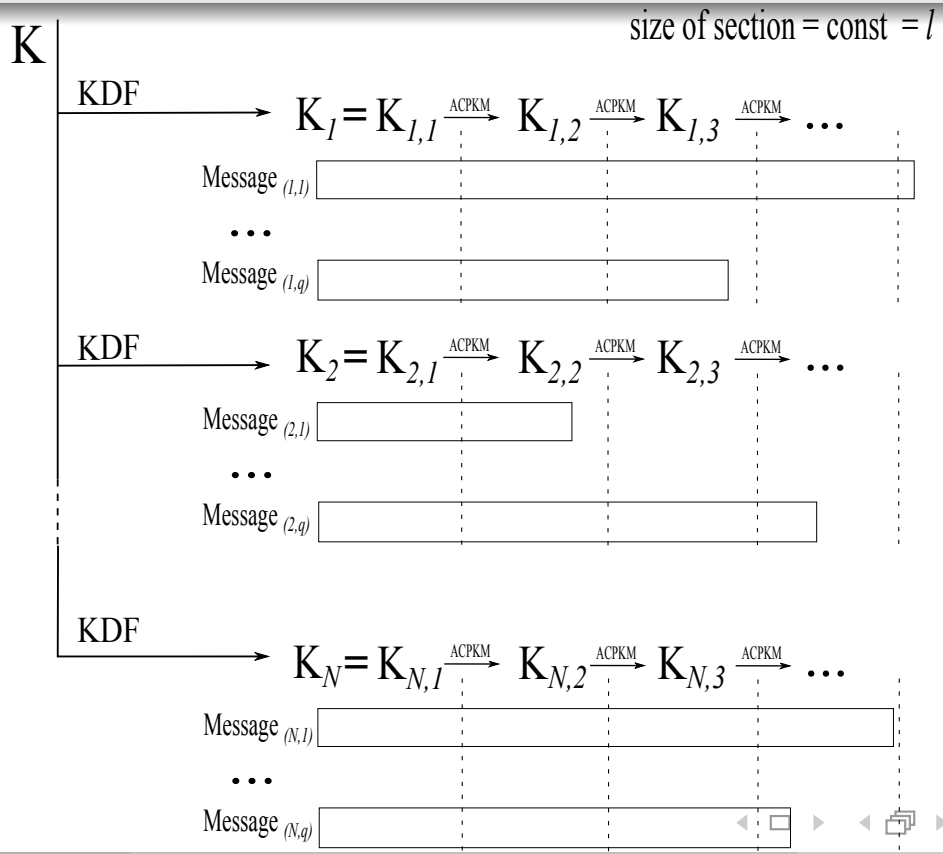
KB	64	128	256	512	1024	2048	4096
MB/s	5040.4	5061.3	5080.6	5105.0	5120.1	5139.4	5150.2
%	2.3	1.9	1.0	0.9	0.7	0.4	0.2

The CTR-ACPKM mode with the AES-128 cipher (hardware support).

External and internal re-keying: allies or rivals?

Two disandvantages to be eliminated by combination

- External: if L is restrictive, inconvenient restrictions on the size of an individual message (with its own header, IV, MAC etc.) appear.
- Internal: section key compromise ⇒ compromise of all next ones.



Summary

- External re-keying (defined independently of a mode):
  - drastically increases the lifetime of keys (considering general bounds, classical and side-channel attacks on a used cipher);
  - almost does not affect performance for long messages;
  - provides forward and backward secrecy of section keys;
  - requires additional operations (KDFs) even for very short plaintexts;
  - procedures (IVs,...) must be handled separately — not transparent;
  - in case of restrictive L: 1) the message sizes can become inconvenient; 2) the key tree should be used — it becomes less effective, if we do not use some additional techniques.
- Internal re-keying approach (defined for a particular mode):
  - drastically increases the lifetime of keys (considering general bounds, classical and side-channel attacks on a used cipher);
  - almost does not affect performance for long messages;
  - does not affect short messages transformation at all;
  - transparent (works like any encryption mode): does not require changes of IV's and restarting MACing;
  - but does not provide backward security of section keys — if needed, should be combined with ext. re-keying (for much larger sections).