

IQRA: Incremental Quadratic Re-keying friendly Authentication scheme

Liliya Akhmetzyanova, Evgeny Alekseev,
Alexandra Babueva, Lidiia Nikiforova and Stanislav Smyshlyaev

CryptoPro LLC



CTCrypt'2021

Outline

1. Motivation
2. IQRA scheme
3. Security notions
4. Security bounds

Outline

1. Motivation
2. IQRA scheme
3. Security notions
4. Security bounds

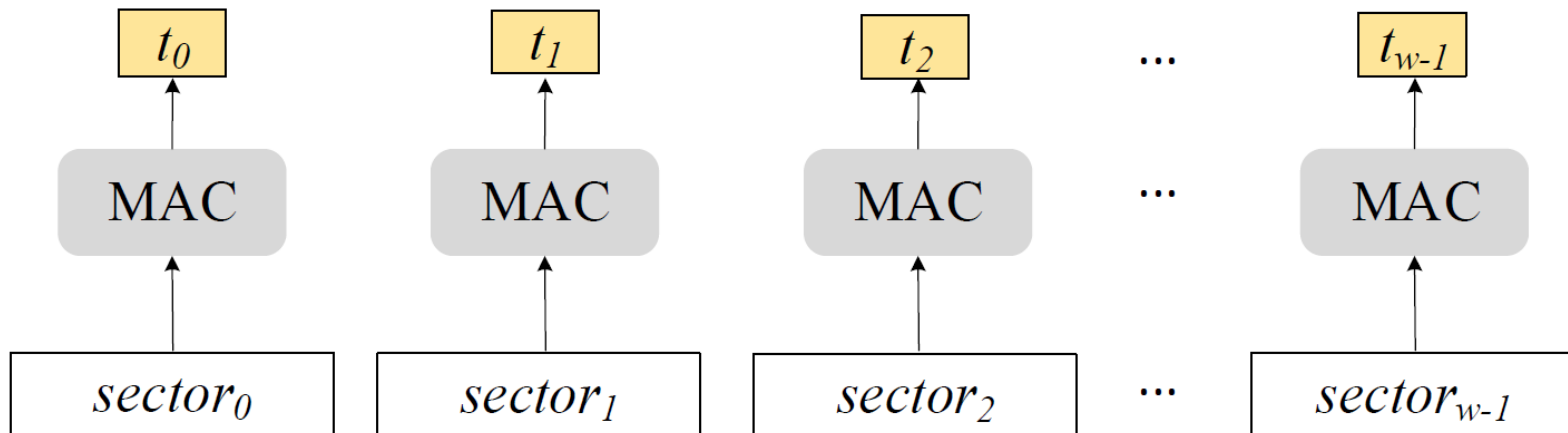
Full disk encryption

2019-2021

- ❖ TC26 «Block ciphers mode of operation designed to protect data storage medium with a block-oriented structure (draft)”, 2019
- ❖ E. Alekseev «Data storage security and full disk encryption», Report on workshop «Mathematical Methods in Cryptanalysis», 2020
- ❖ E. Alekseev, L. Akhmetzyanova, A. Babueva, S. Smyshlyaev «Data storage security and full disk encryption», Applied Mathematics, 2020, №49, pp 78-97
- ❖ D. Bogdanov and V. Nozdrunov, «Data storage encryption. DEC mode», RusCrypto’2021
- ❖ L. Akhmetzyanova, L. Nikiforova «FDE scheme Matryoshka», Report on workshop «Mathematical Methods in Cryptanalysis», 2021
- ❖ D. Bogdanov and V. Nozdrunov «Some Properties of One Mode of Operation of Block Ciphers», CTCrypt’2021

Full disk encryption

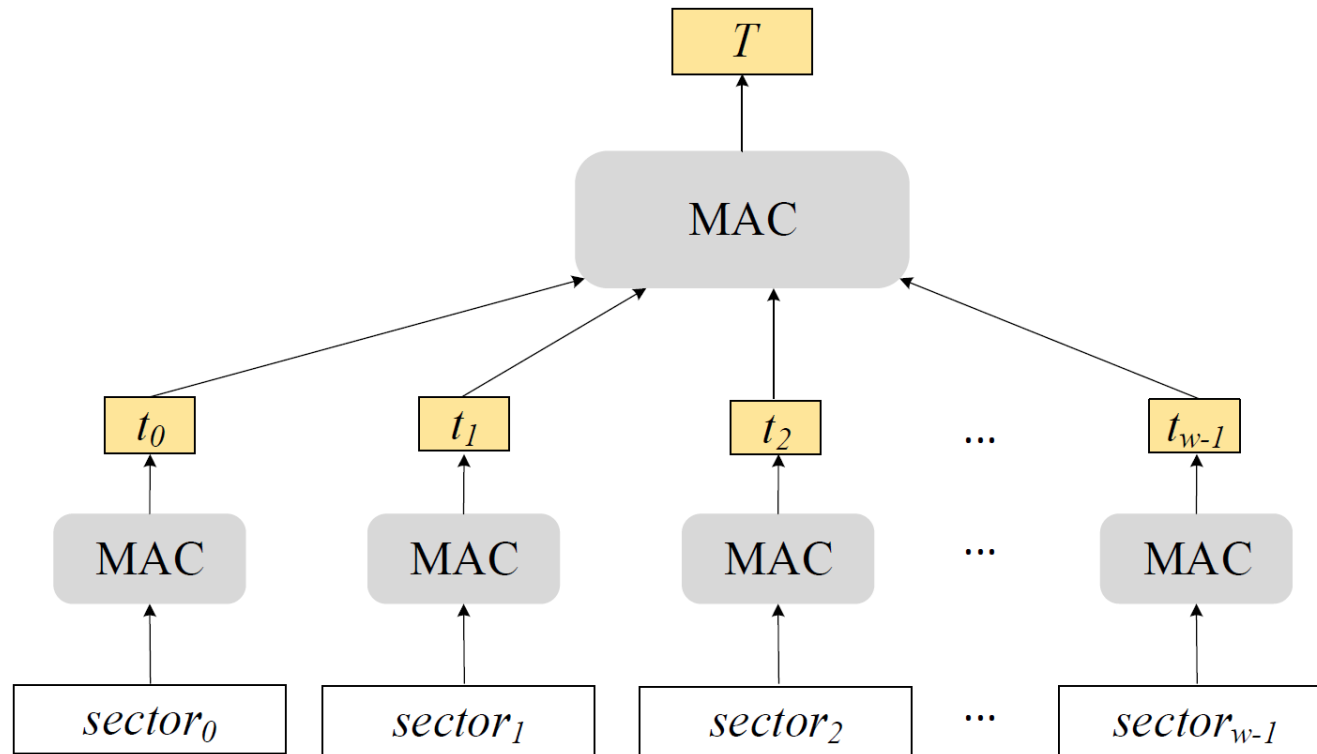
Task: to provide disk integrity (disk = w sectors)



Problem: sector-level integrity, but no disk-level integrity

Full disk encryption

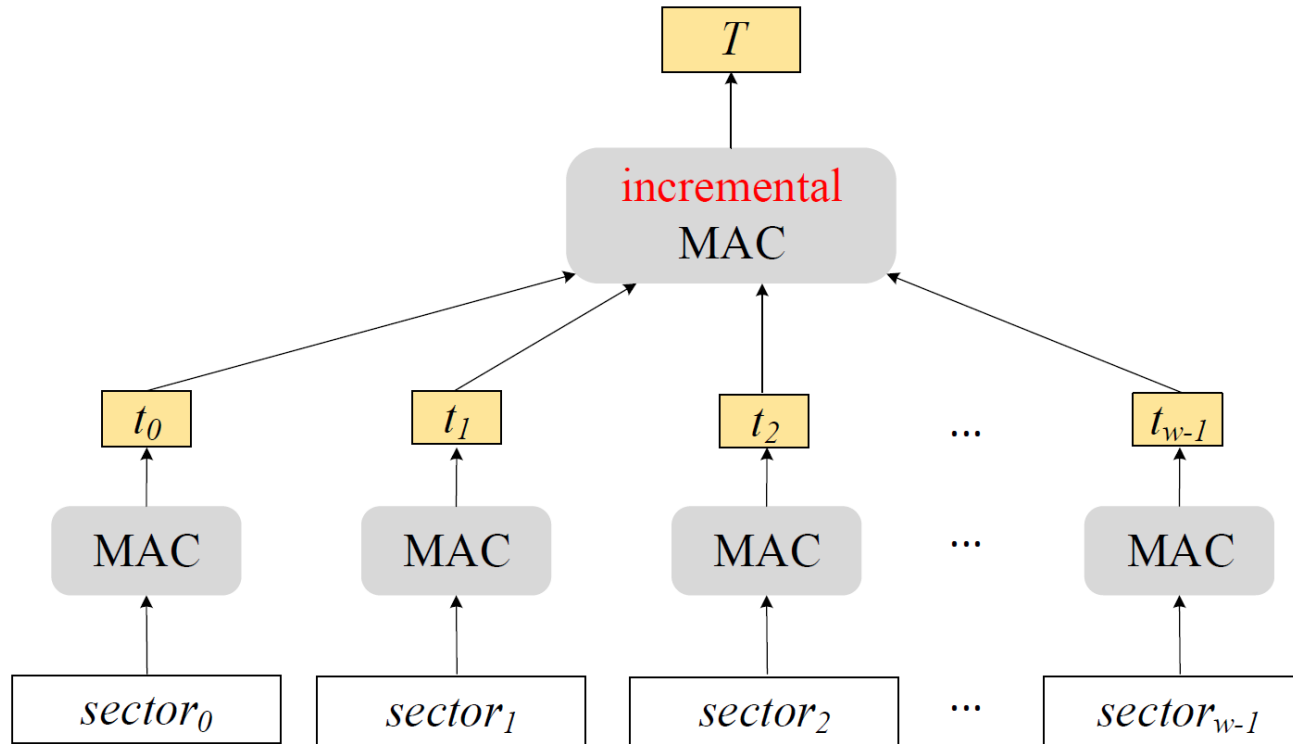
Task: to provide disk integrity (disk = w sectors)



Problem: re-computing global MAC is too slow

Full disk encryption

Task: to provide disk integrity (disk = w sectors)

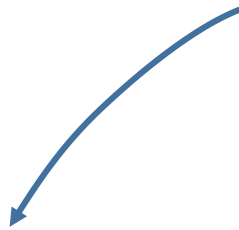


Solution: use incremental MAC

Incremental MAC

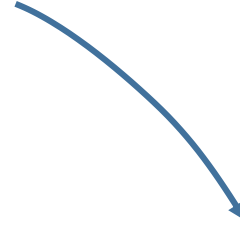
M. Bellare, O. Goldreich, S. Goldwasser. 1994
Incremental Cryptography: The Case of Hashing and Signing

after each message modification



Standard MAC

re-compute MAC from scratch



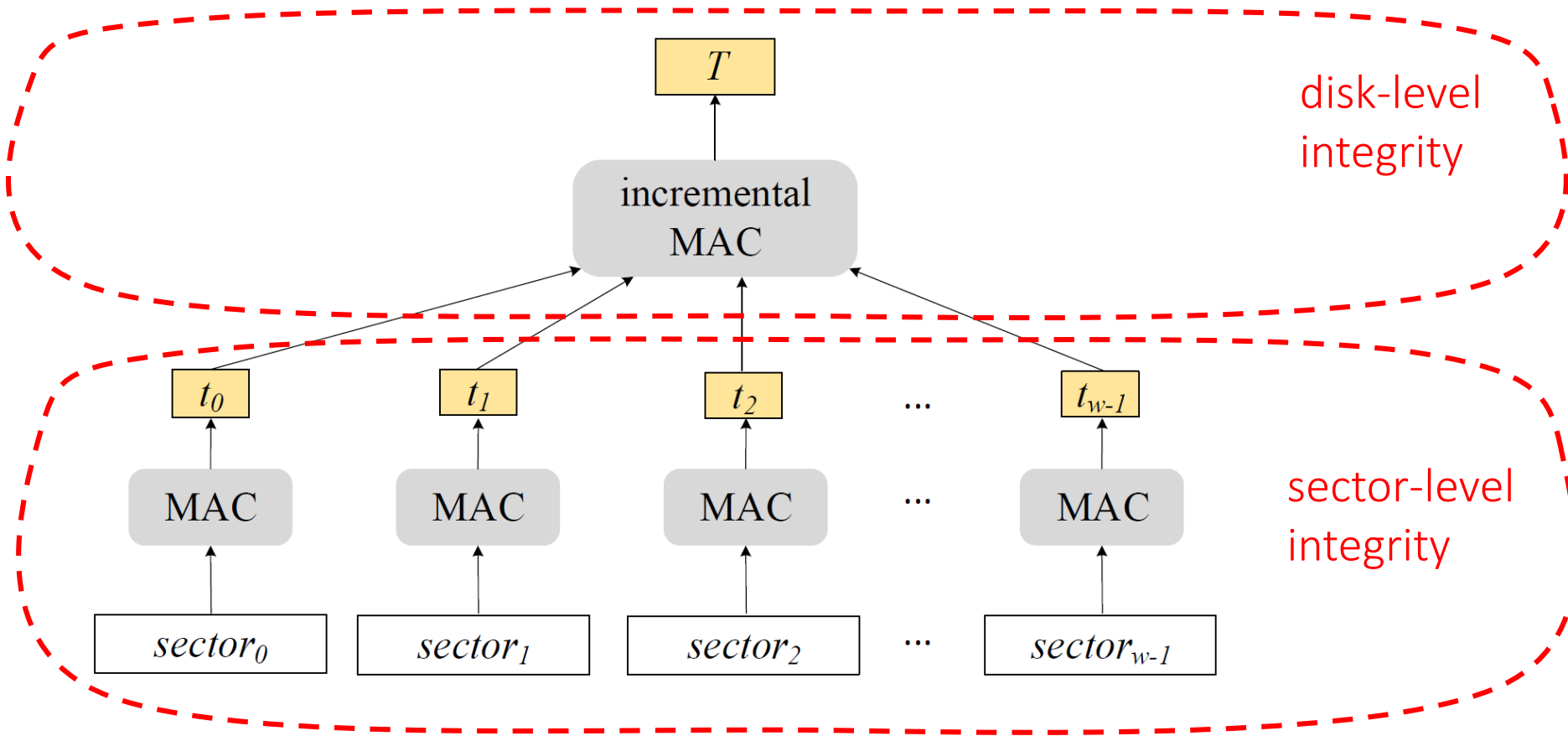
Incremental MAC

update MAC in a time proportional
to the number of modified blocks

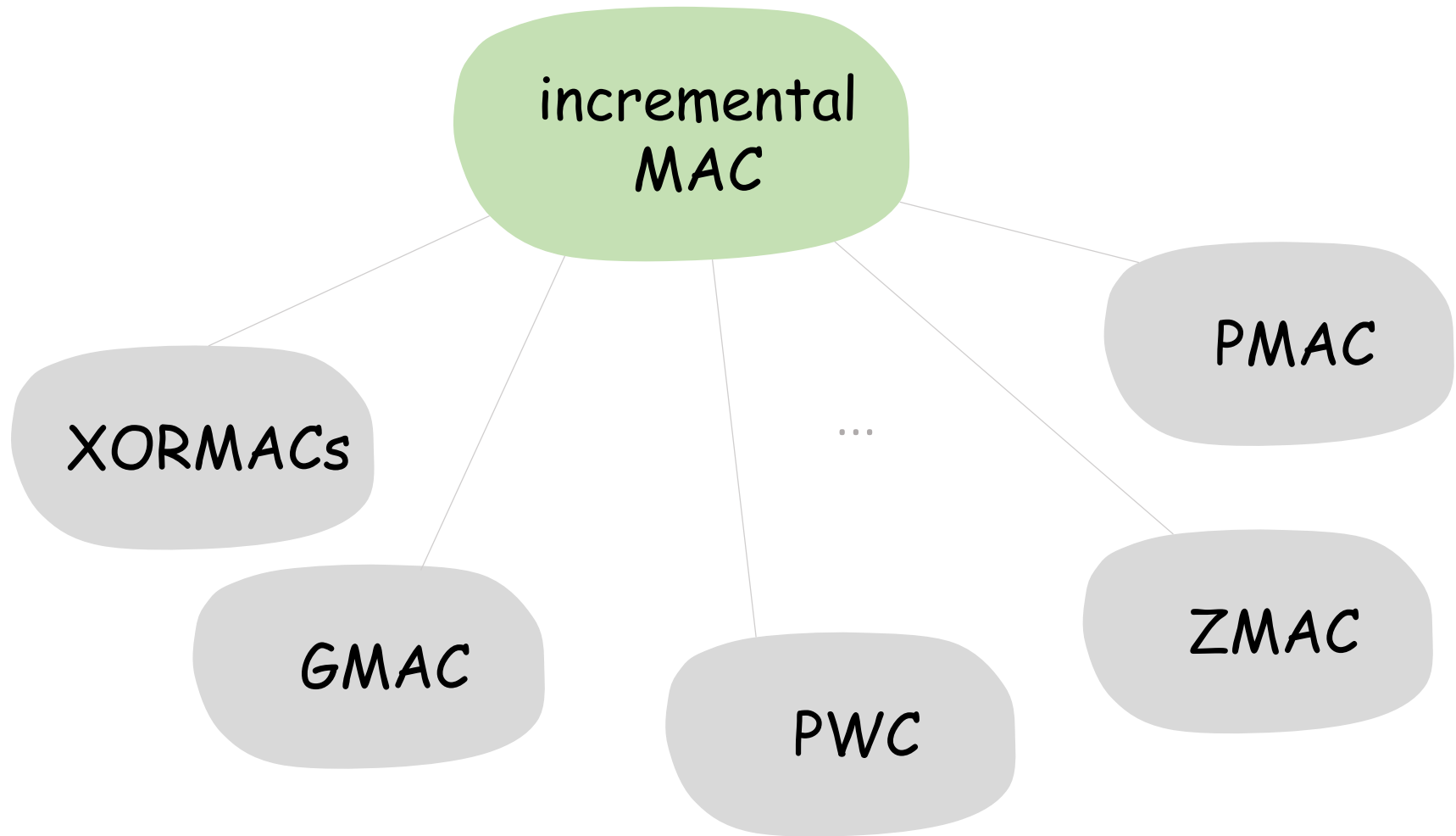
Supported operations:

- ❖ Replace
- ❖ Insert
- ❖ Delete

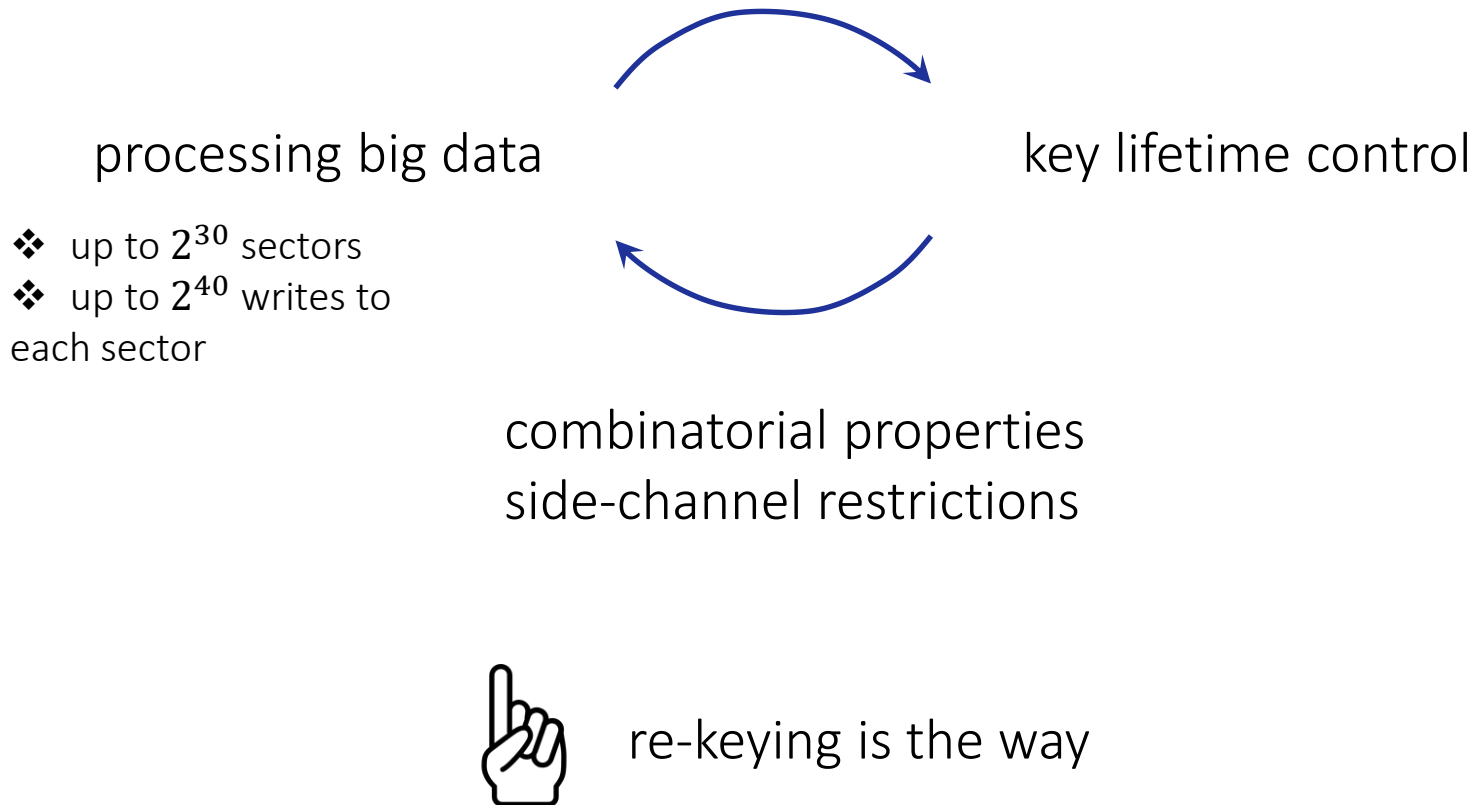
Full disk encryption



Existing schemes

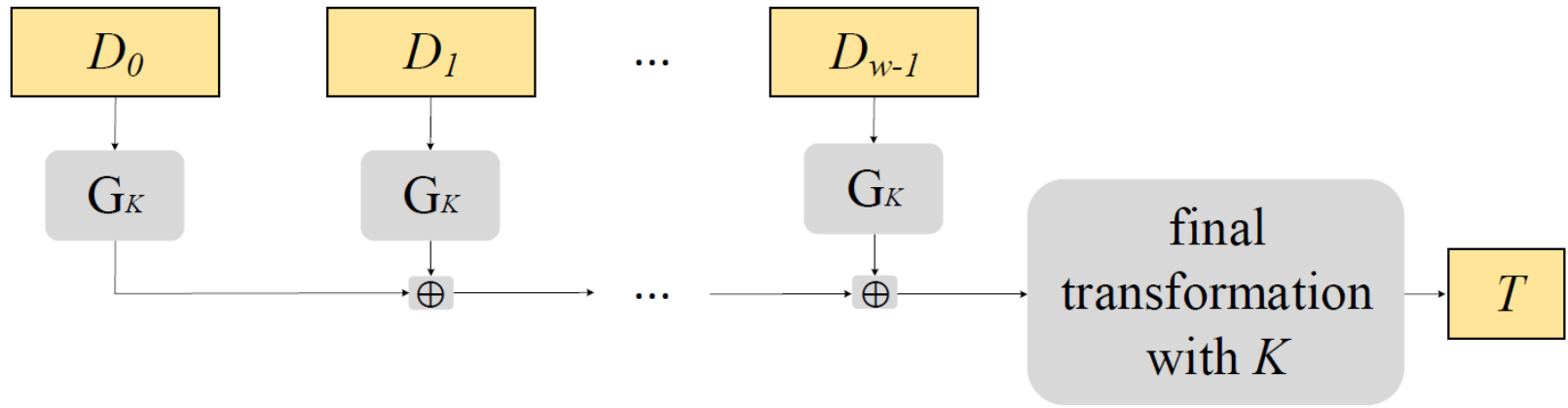


Full disk encryption



In detail:
E. Alekseev, L. Akhmetzyanova, I. Oshkin, S. Smyshlyaev «On Internal Re-keying» //International Conference on Research in Security Standardisation. – Springer, Cham, 2020. – C. 23-45.

Existing schemes



one key for processing each block and final transformation



no incrementality in case of key update

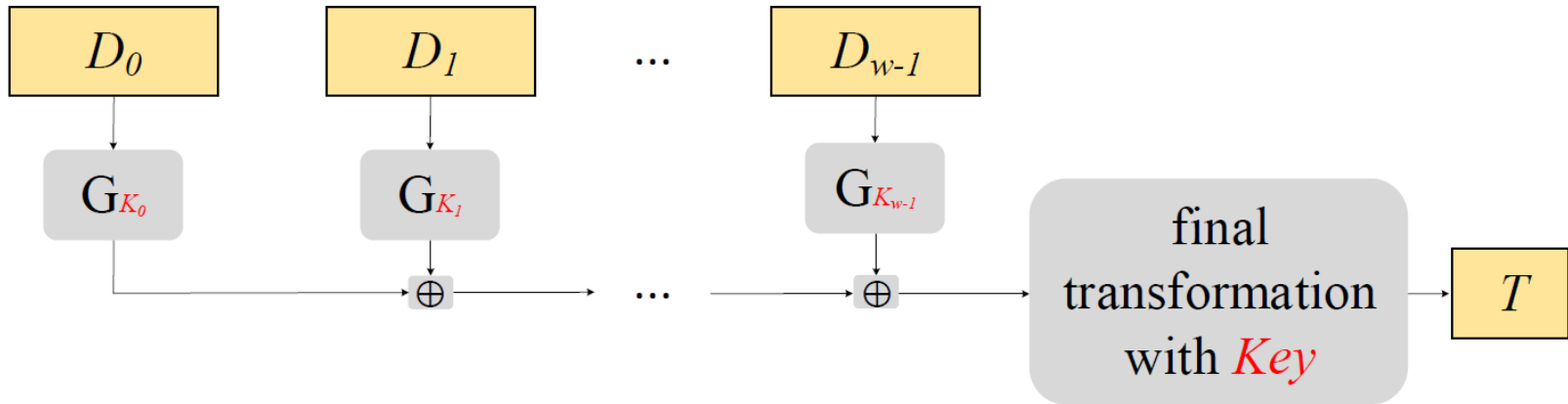
Problem

To construct incremental re-keying friendly MAC scheme

Outline

1. Motivation
2. IQRA scheme
3. Security notions
4. Security bounds

IQRA scheme: idea



separate the keys used for different purposes
(for processing each block and final transformation)



it's possible to change each key independently
(if necessary) saving incrementality

IQRA scheme

❖ Based on:

- ✓ $F: \{0,1\}^k \times \{0,1\}^n \rightarrow \{0,1\}^n$ — underlying PRF
- ✓ quadratic polynomial

❖ Nonce-based

❖ Designed for fixed-length data and support only Replace operation

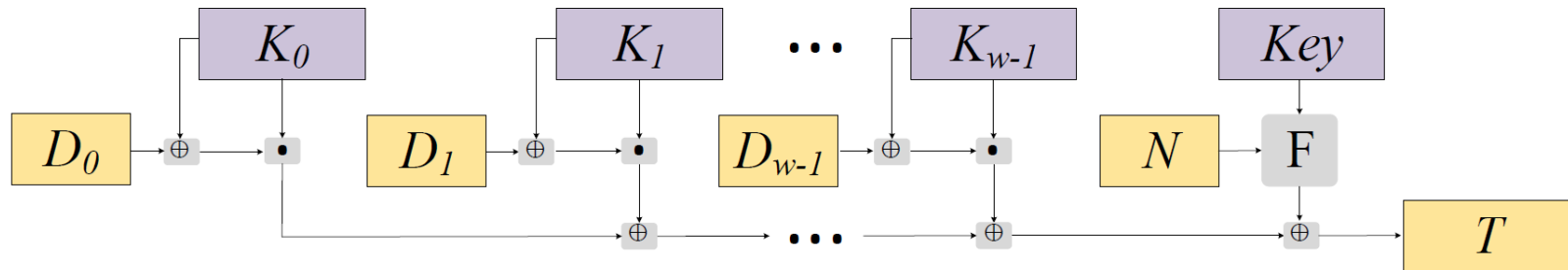
$$\bar{D} = (D_0, \dots, D_{w-1}), \quad D_i \in \{0,1\}^n$$

❖ Key space:

- ✓ $\bar{K} = (K_0, \dots, K_{w-1})$ — block keys
- ✓ *Key* — finalization key

IQRA scheme

$IQRA.Tag(\bar{K}, Key, N, \bar{D})$

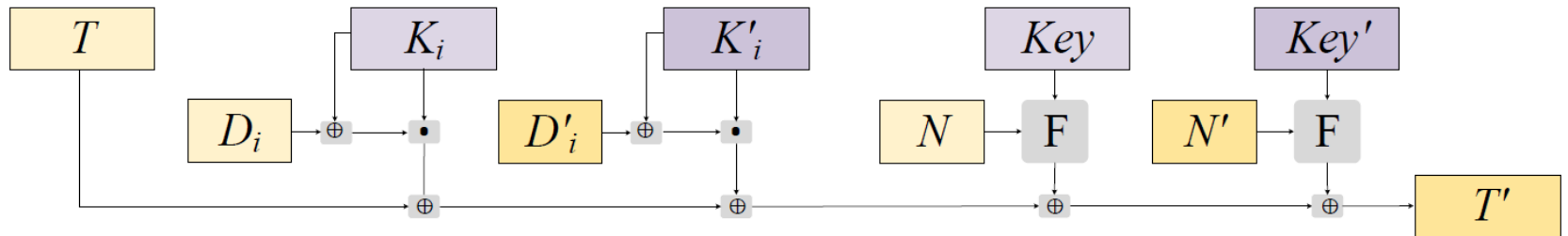


$$T = (D_0 \oplus K_0) \cdot K_0 \oplus \dots \oplus (D_{w-1} \oplus K_{w-1}) \cdot K_{w-1} \oplus F(Key, N)$$

cost: $w \text{ MUL} + 1 \text{ F}$

IQRA scheme

IQRA.Replace(*Key*, *Key'*, *K_i*, *K'_i*, *N*, *N'*, *D_i*, *D'_i*, *T*)

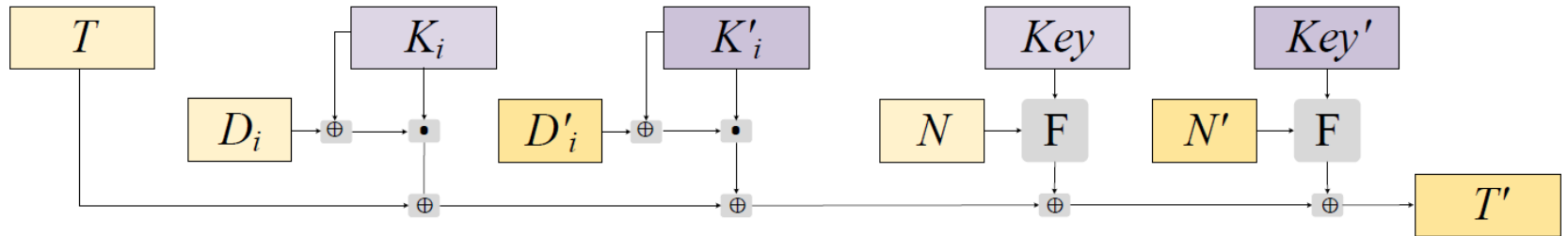


$$T' = T \oplus (D_i \oplus K_i) \cdot K_i \oplus (D'_i \oplus K'_i) \cdot K'_i \oplus F(Key, N) \oplus F(Key', N')$$

cost: $2 MUL + 2 F$

IQRA scheme

IQRA.Replace(*Key*, *Key'*, *K_i*, *K'_i*, *N*, *N'*, *D_i*, *D'_i*, *T*)



$$T' = T \oplus (D_i \oplus K_i) \cdot K_i \oplus (D'_i \oplus K'_i) \cdot K'_i \oplus F(\text{Key}, N) \oplus F(\text{Key}', N')$$

cost: $2 \text{ MUL} + 2 \text{ F}$

$K'_i = K_i$:

$$(D_i \oplus K_i) \cdot K_i \oplus (D'_i \oplus K'_i) \cdot K'_i = (D_i \oplus D'_i) \cdot K_i$$

IQRA scheme: keys



Where to find $(w + 1)$ keys?

IQRA scheme: keys



Where to find $(w + 1)$ keys?

Derive from K_{master} :

$$Key = KDF(K_{master}, 0x00 \parallel seed, k)$$

$$K_i = KDF(K_{master}, 0x01 \parallel i \parallel S_i, n), \quad i = 0, \dots, w - 1$$

$seed, \bar{S} = (S_0, \dots, S_{w-1})$ – key derivation parameters

IQRA scheme: keys



Where to find $(w + 1)$ keys?

Derive from K_{master} :

$$Key = KDF(K_{master}, 0x00 \parallel seed, k)$$

$$K_i = KDF(K_{master}, 0x01 \parallel i \parallel S_i, n), \quad i = 0, \dots, w - 1$$

$seed, \bar{S} = (S_0, \dots, S_{w-1})$ – key derivation parameters

How to store $seed, \bar{S}$?

- ❖ don't need to provide confidentiality and integrity
- ❖ smaller than keys

IQRA scheme: keys



Where to find $(w + 1)$ keys?

Derive from K_{master} :

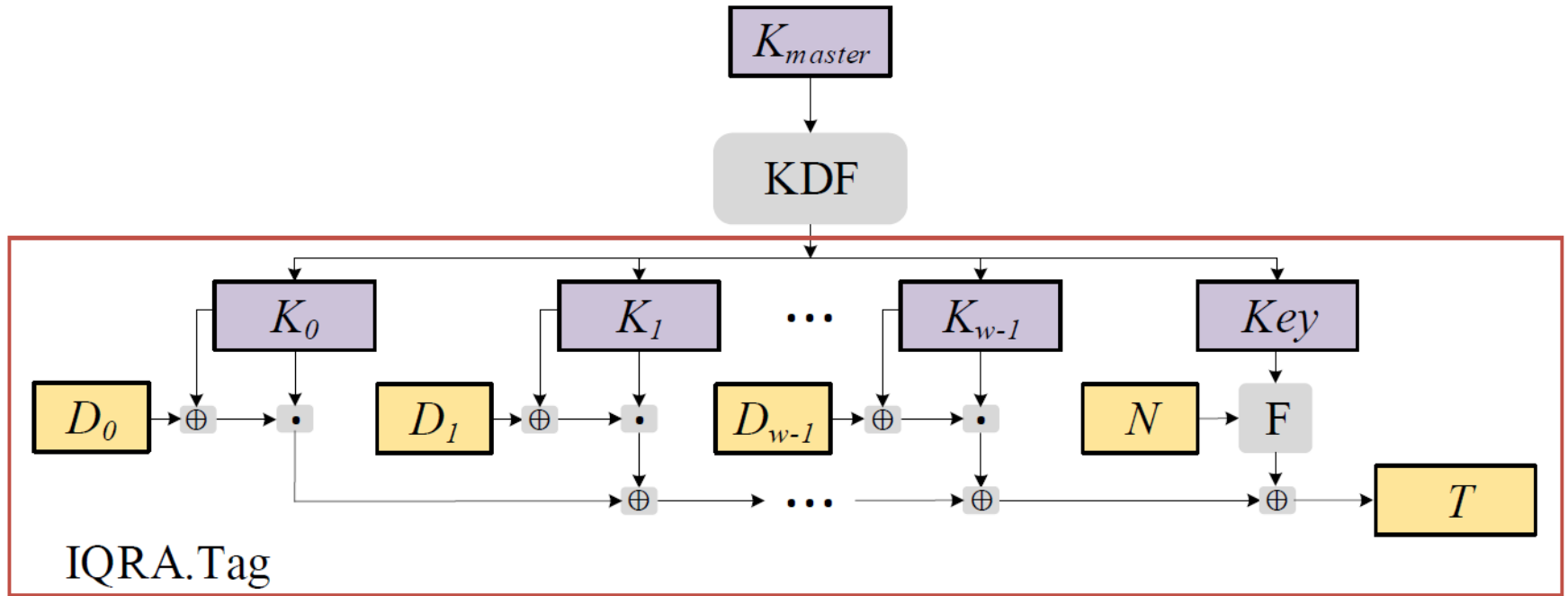
$$Key = KDF(K_{master}, 0x00 \parallel seed, k)$$

$$K_i = KDF(K_{master}, 0x01 \parallel i \parallel S_i, n), \quad i = 0, \dots, w - 1$$

$seed, \bar{S} = (S_0, \dots, S_{w-1})$ – key derivation parameters

Right choice of KDF parameters – no need to change K_{master} during life-cycle of IQRA scheme

[IQRA, KDF] scheme

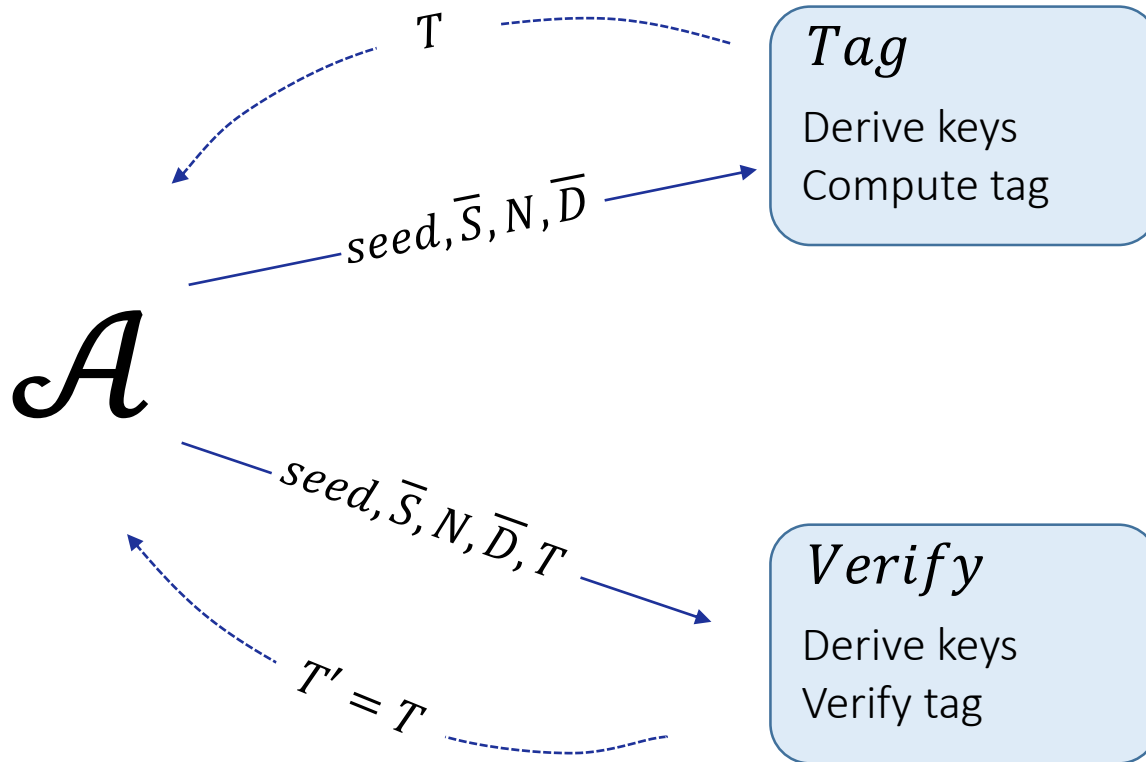


Outline

1. Motivation
2. IQRA scheme
3. Security notions
4. Security bounds

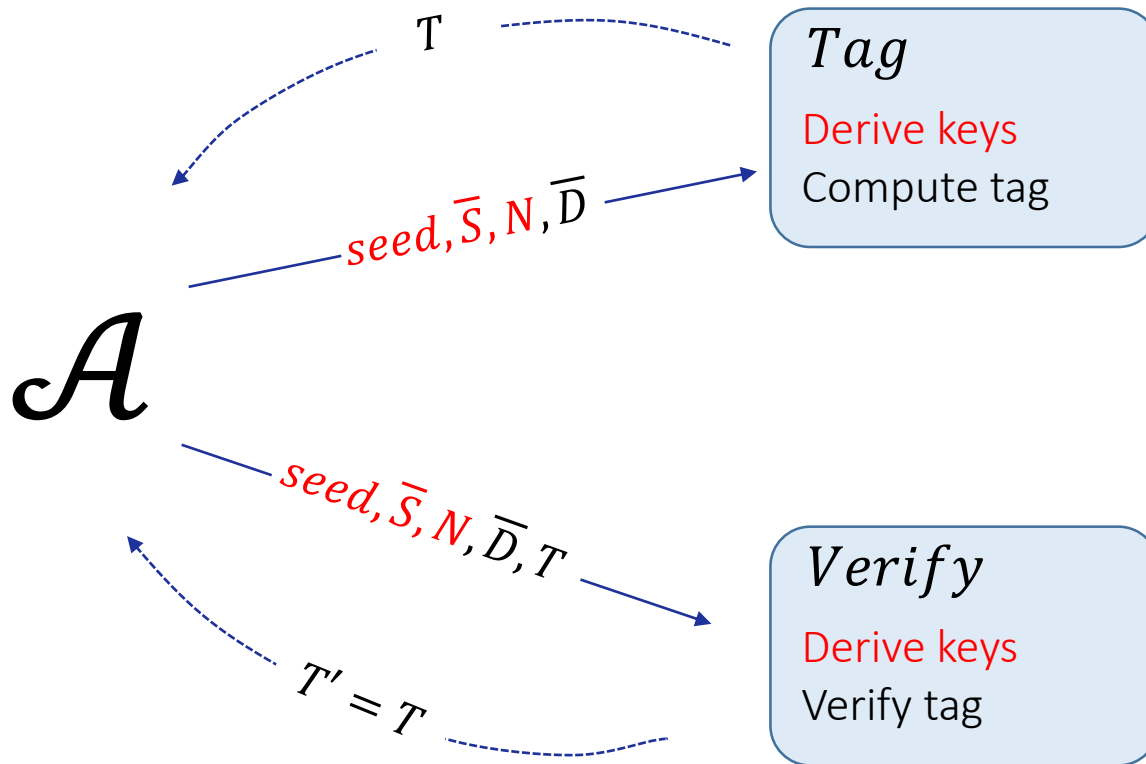
SUF-CSMA: idea

Strong Unforgeability under Chosen Settings and Message Attack



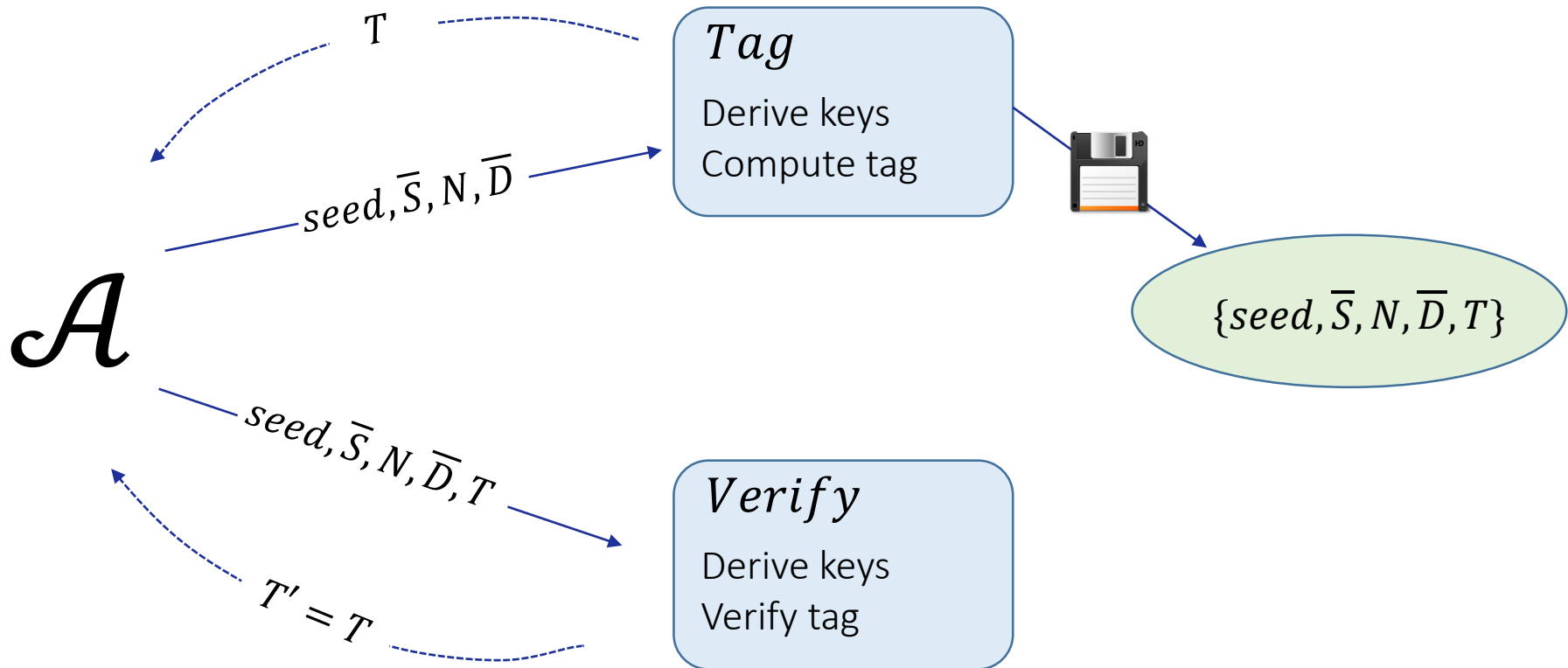
SUF-CSMA: idea

Strong Unforgeability under Chosen **Settings** and Message Attack



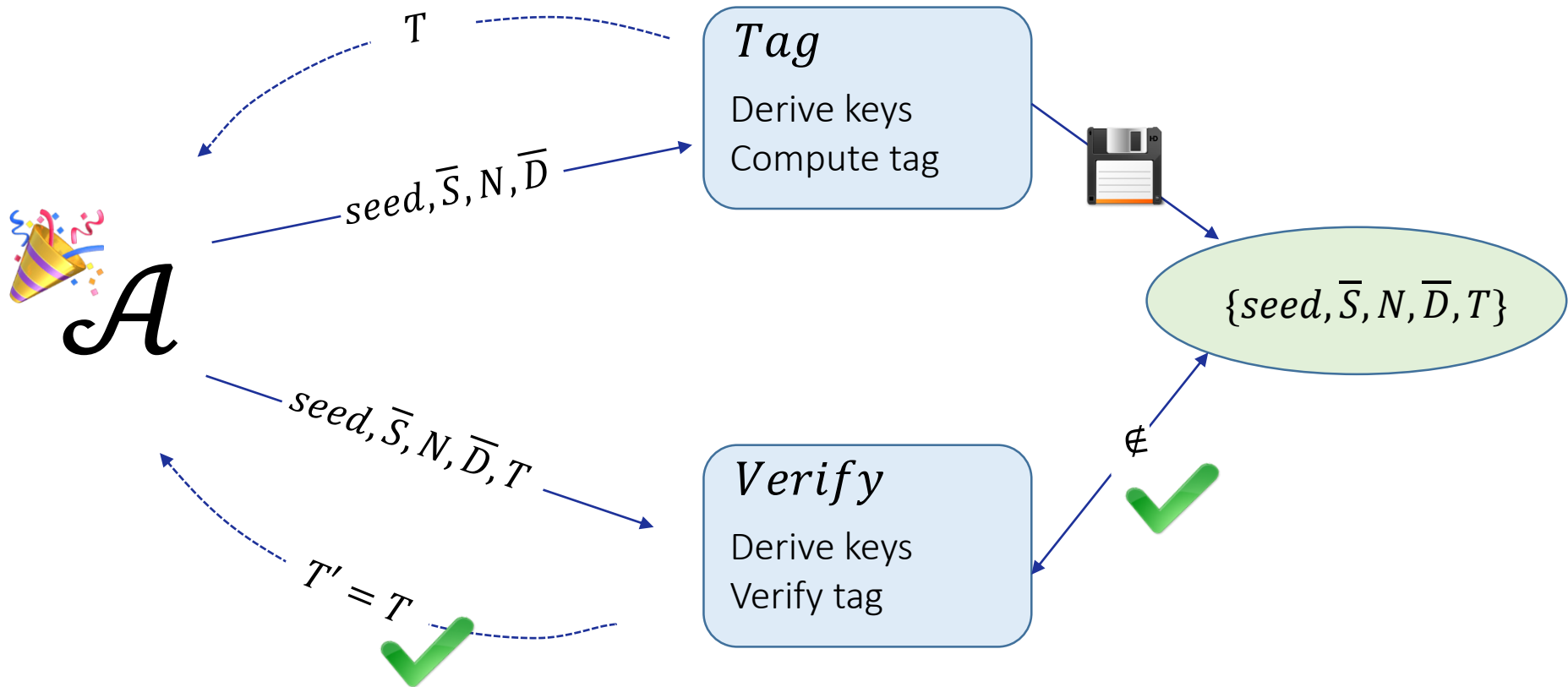
SUF-CSMA: idea

Strong Unforgeability under Chosen Settings and Message Attack



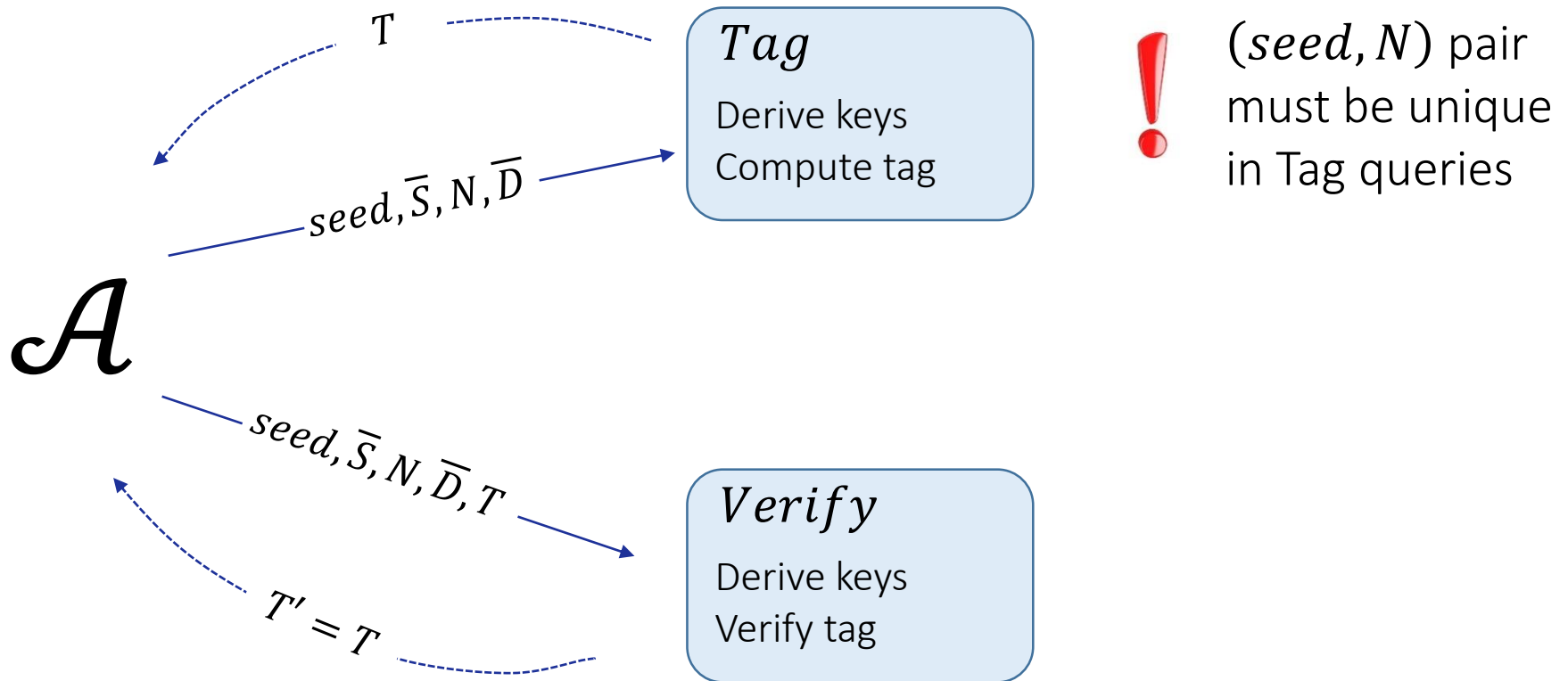
SUF-CSMA: idea

Strong Unforgeability under Chosen Settings and Message Attack



SUF-CSMA: idea

Strong Unforgeability under Chosen Settings and Message Attack



Design rationale

Why unique (*seed*, *N*)?

SUF-CSMA adversary can make two Tag queries with $D_i \neq 0$ and $D_i = 0$ and same (*seed*, *N*) and recover K_i value

$$\begin{cases} T = D_0 K_0 \oplus K_0^2 \oplus \dots \oplus D_i K_i \oplus K_i^2 \oplus \dots \oplus D_{w-1} K_{w-1} \oplus K_{w-1}^2 \oplus F(\text{Key}, N) \\ T' = D_0 K_0 \oplus K_0^2 \oplus \dots \oplus K_i^2 \oplus \dots \oplus D_{w-1} K_{w-1} \oplus K_{w-1}^2 \oplus F(\text{Key}, N) \end{cases}$$



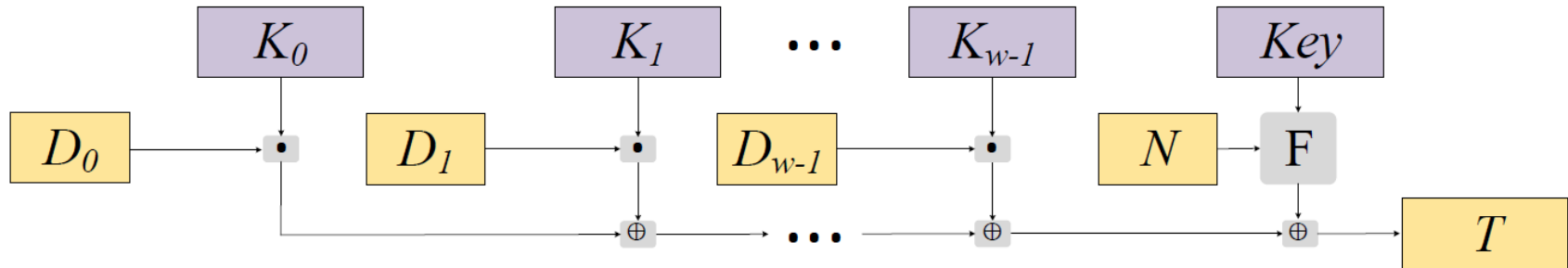
$$T \oplus T' = D_i K_i$$



$$K_i$$

Design rationale

Why quadratic polynomial?



$$\begin{aligned} T &= (D_0 \oplus \cancel{K_0}) \cdot K_0 \oplus \dots \oplus (D_{w-1} \oplus \cancel{K_{w-1}}) \cdot K_{w-1} \oplus F(Key, N) = \\ &= D_0 \cdot K_0 \oplus \dots \oplus D_{w-1} \cdot K_{w-1} \oplus F(Key, N) \end{aligned}$$

SUF-CSMA adversary can make forgery with $D_i = 0$ since the tag is independent from K_i (and thus S_i) value.

Outline

1. Motivation
2. IQRA scheme
3. Security notions
4. Security bounds

Security bounds: general case

$$\text{Adv}_{[IQRA_F, KDF]}^{\text{SUF-CSMA}}(A) \leq \text{Adv}_{KDF}^{\text{PRF}^*}(B) + d \cdot \text{Adv}_F^{\text{PRF}}(C) + \frac{q_{\text{ver}}}{2^{n-1}}$$

PRF^* – variable-length output PRF

d – number of distinct seed values in A 's queries

$q_{\text{tag}}/q_{\text{ver}}$ – number of A 's queries to Tag/Verify oracle

Adversary B :

❖ $(w + 1)(q_{\text{tag}} + q_{\text{ver}})$ queries

Adversary C :

❖ r queries, where r – maximal number of distinct nonce values queried with the same seed value

Security bounds: general case

$$Adv_{[IQRA_F, KDF]}^{SUF-CSMA}(A) \leq Adv_{KDF}^{PRF^*}(B) + d \cdot Adv_F^{PRF}(C) + \frac{q_{ver}}{2^{n-1}}$$

PRF^* – variable-length output PRF

d – number of distinct seed values in A 's queries

q_{tag}/q_{ver} – number of A 's queries to Tag/Verify oracle



The obtained bound does not depend on the amount of data processed using the same block keys $K_0, \dots, K_{w-1}$

The requirement to change block keys may follow from side-channel restrictions

Security bounds: block cipher case

Extend Bernstein results on multiple functions case

(Bernstein, D.J. «Stronger Security Bounds for Permutations», 2005)

$$Adv_{[IQRA_E, KDF]}^{SUF-CSMA}(A) \leq Adv_{KDF}^{PRF^*}(B) + d \cdot Adv_E^{PRP}(C) + \delta_n(r_1) \dots \delta_n(r_d) \frac{q_{ver}}{2^{n-1}}$$

$$\delta_n(q) = \left(1 - \frac{q-1}{2^n}\right)^{-q/2}$$

r_i — number of distinct nonce values queried with each seed value

Thank you for your attention!
Questions?

babueva@cryptopro.ru

lah@cryptopro.ru

alekseev@cryptopro.ru

nikiforova@cryptopro.ru

svs@cryptopro.ru