Ketje and Keyak

Guido Bertoni[1] Joan Daemen[1]
Michaël Peeters[2] Gilles Van Assche[1] Ronny Van Keer[1]

[1]STMicroelectronics

[2]NXP Semiconductors

DIAC 2014

Outline

Overview

- Inspired by KECCAK and DUPLEX
- KEYAK targeting high performances
  - Using reduced-round KECCAK-$f[1600]$ or KECCAK-$f[800]$
  - Optionally parallelizable
- KETJE targeting lightweight
  - Using reduced-round KECCAK-$f[400]$ or KECCAK-$f[200]$

Overview

- Inspired by KECCAK and DUPLEX
- KEYAK targeting high performances
  - Using reduced-round KECCAK-$f[1600]$ or KECCAK-$f[800]$
  - Optionally parallelizable
- KETJE targeting lightweight
  - Using reduced-round KECCAK-$f[400]$ or KECCAK-$f[200]$

Overview

- Inspired by KECCAK and DUPLEX
- KEYAK targeting high performances
  - Using reduced-round KECCAK-$f[1600]$ or KECCAK-$f[800]$
  - Optionally parallelizable
- KETJE targeting lightweight
  - Using reduced-round KECCAK-$f[400]$ or KECCAK-$f[200]$

Two approaches

Keyak:

- DuplexWrap
- A (strong) permutation
  - fixed #rounds
- Block-oriented
- Cryptanalysis
  - permutation-level

Ketje:

- MonkeyWrap
- A (thin) round function
  - #rounds in phases
- Stream-oriented
- Cryptanalysis
  - round function + construction

Two approaches

Keyak:

- DuplexWrap
- A (strong) permutation
  - fixed #rounds
- Block-oriented
- Cryptanalysis
  - permutation-level

Ketje:

- MonkeyWrap
- A (thin) round function
  - #rounds in phases
- Stream-oriented
- Cryptanalysis
  - round function + construction

Outline

KEYAK goals

- Nonce-based AE function
- 128-bit security (incl. multi-target)
- Sequence of header-body pairs
  - keeping the state during the session
- Optionally parallelizable
- Using reduced-round KECCAK-f[1600] or KECCAK-f[800], to allow
  - implementation re-use
  - cryptanalysis re-use
  - reasonable side-channel protections

(... and because we like it ...)

KEYAK goals

- Nonce-based AE function
- 128-bit security (incl. multi-target)
- Sequence of header-body pairs
  - keeping the state during the session
- Optionally parallelizable
- Using reduced-round KECCAK-$f[1600]$ or KECCAK-$f[800]$, to allow
  - implementation re-use
  - cryptanalysis re-use
  - reasonable side-channel protections

(... and because we like it ...)

KEYAK goals

- Nonce-based AE function
- 128-bit security (incl. multi-target)
- Sequence of header-body pairs
  - keeping the state during the session
- Optionally parallelizable
- Using reduced-round KECCAK-$f[1600]$ or KECCAK-$f[800]$, to allow
  - implementation re-use
  - cryptanalysis re-use
  - reasonable side-channel protections

(... and because we like it ...)

Keyak goals

- Nonce-based AE function
- 128-bit security (incl. multi-target)
- Sequence of header-body pairs
  - keeping the state during the session
- Optionally parallelizable
- Using reduced-round Keccak-f[1600] or Keccak-f[800], to allow
  - implementation re-use
  - cryptanalysis re-use
  - reasonable side-channel protections

(... and because we like it ...)

KEYAK goals

- Nonce-based AE function
- 128-bit security (incl. multi-target)
- Sequence of header-body pairs
  - keeping the state during the session
- Optionally parallelizable
- Using reduced-round KECCAK-$f[1600]$ or KECCAK-$f[800]$, to allow
  - implementation re-use
  - cryptanalysis re-use
  - reasonable side-channel protections

(... and because we like it ...)

KEYAK goals

- Nonce-based AE function
- 128-bit security (incl. multi-target)
- Sequence of header-body pairs
  - keeping the state during the session
- Optionally parallelizable
- Using reduced-round KECCAK-$f[1600]$ or KECCAK-$f[800]$, to allow
  - implementation re-use
  - cryptanalysis re-use
  - reasonable side-channel protections

(... and because we like it ...)

KEYAK goals

- Nonce-based AE function
- 128-bit security (incl. multi-target)
- Sequence of header-body pairs
  - keeping the state during the session
- Optionally parallelizable
- Using reduced-round KECCAK-$f[1600]$ or KECCAK-$f[800]$, to allow
  - implementation re-use
  - cryptanalysis re-use
  - reasonable side-channel protections

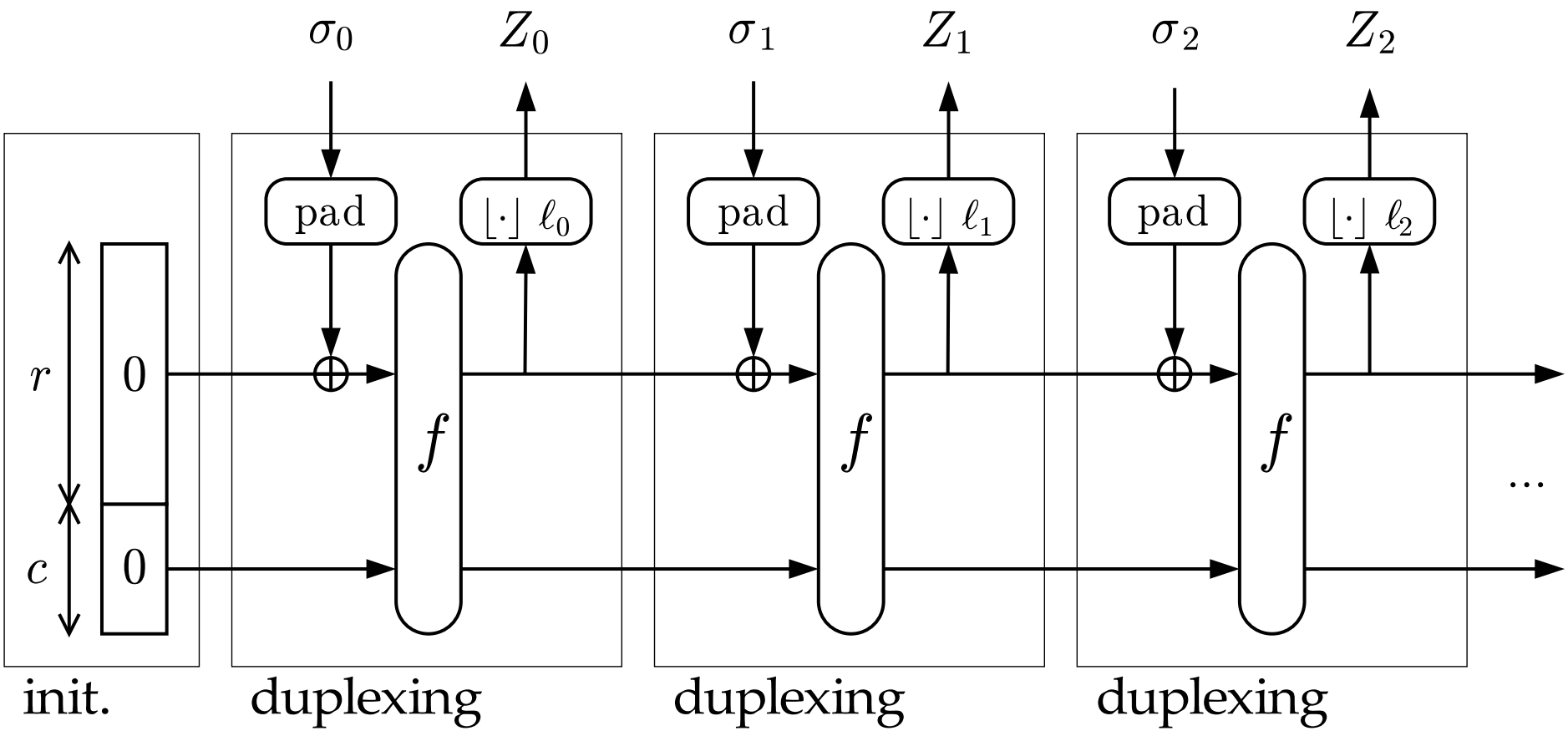
(... and because we like it ...)

KEYAK goals

- Nonce-based AE function
- 128-bit security (incl. multi-target)
- Sequence of header-body pairs
  - keeping the state during the session
- Optionally parallelizable
- Using reduced-round KECCAK-$f[1600]$ or KECCAK-$f[800]$, to allow
  - implementation re-use
  - cryptanalysis re-use
  - reasonable side-channel protections

(... and because we like it ...)

Duplex layer

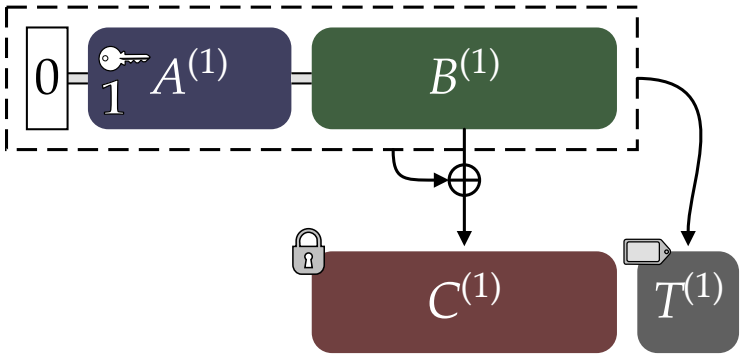


KECCAK-$p[1600, n_r = 12]$ or KECCAK-$p[800, n_r = 12]$

DuplexWrap layer

DuplexWrap

- is a nonce-based authenticated encryption mode;
- works on sequences of header-body pairs.



$A^{(1)}$ contains the key and must be unique, e.g.,

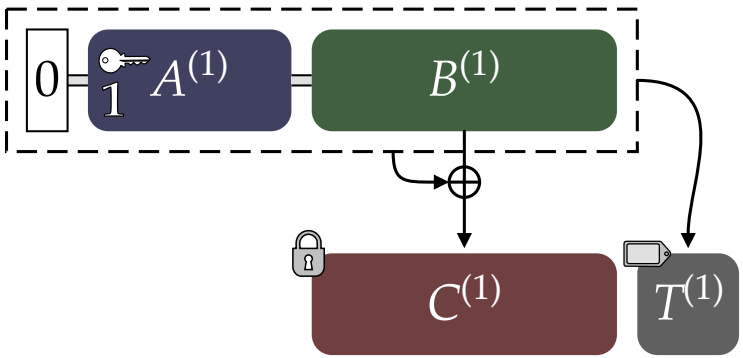
- $A^{(1)}$ contains a session key used only once;
- $A^{(1)}$ contains a key and a nonce.

In general: $A^{(1)} = \text{key}||\text{nonce}||\text{associated data}$.

DuplexWrap layer

DuplexWrap

- is a nonce-based authenticated encryption mode;
- works on sequences of header-body pairs.



$A^{(1)}$ contains the key and must be unique, e.g.,

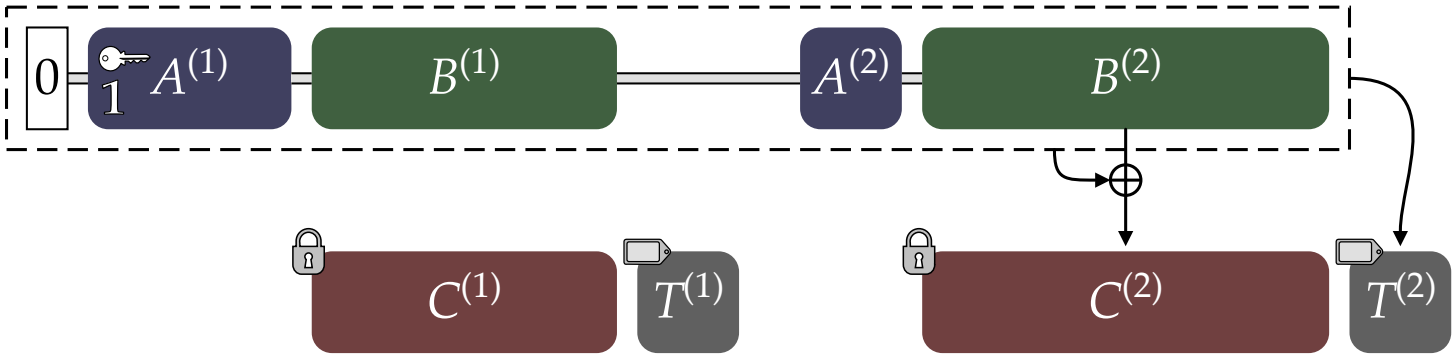
- $A^{(1)}$ contains a session key used only once;
- $A^{(1)}$ contains a key and a nonce.

In general: $A^{(1)} = \text{key}||\text{nonce}||\text{associated data}$.

DuplexWrap layer

DuplexWrap

- is a nonce-based authenticated encryption mode;
- works on sequences of header-body pairs.



$A^{(1)}$ contains the key and must be unique, e.g.,

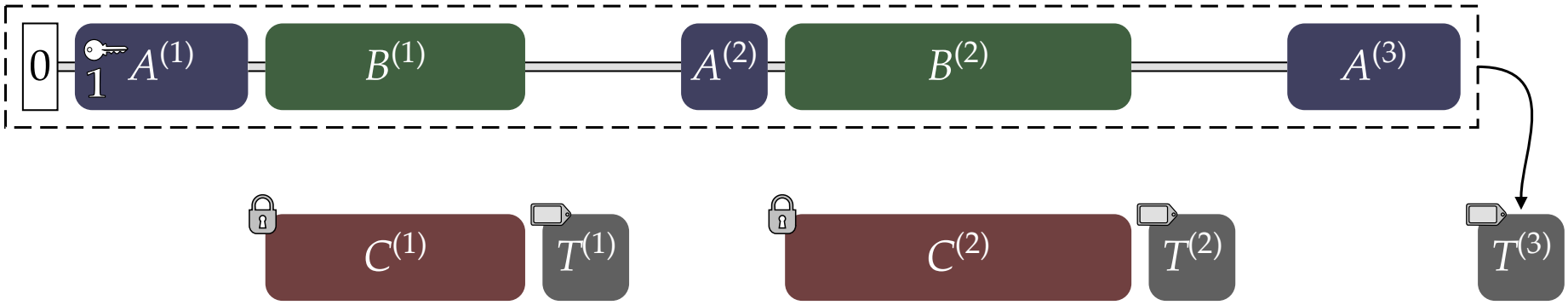
- $A^{(1)}$ contains a session key used only once;
- $A^{(1)}$ contains a key and a nonce.

In general: $A^{(1)} = \text{key}||\text{nonce}||\text{associated data}$.

DuplexWrap layer

DuplexWrap

- is a nonce-based authenticated encryption mode;
- works on sequences of header-body pairs.

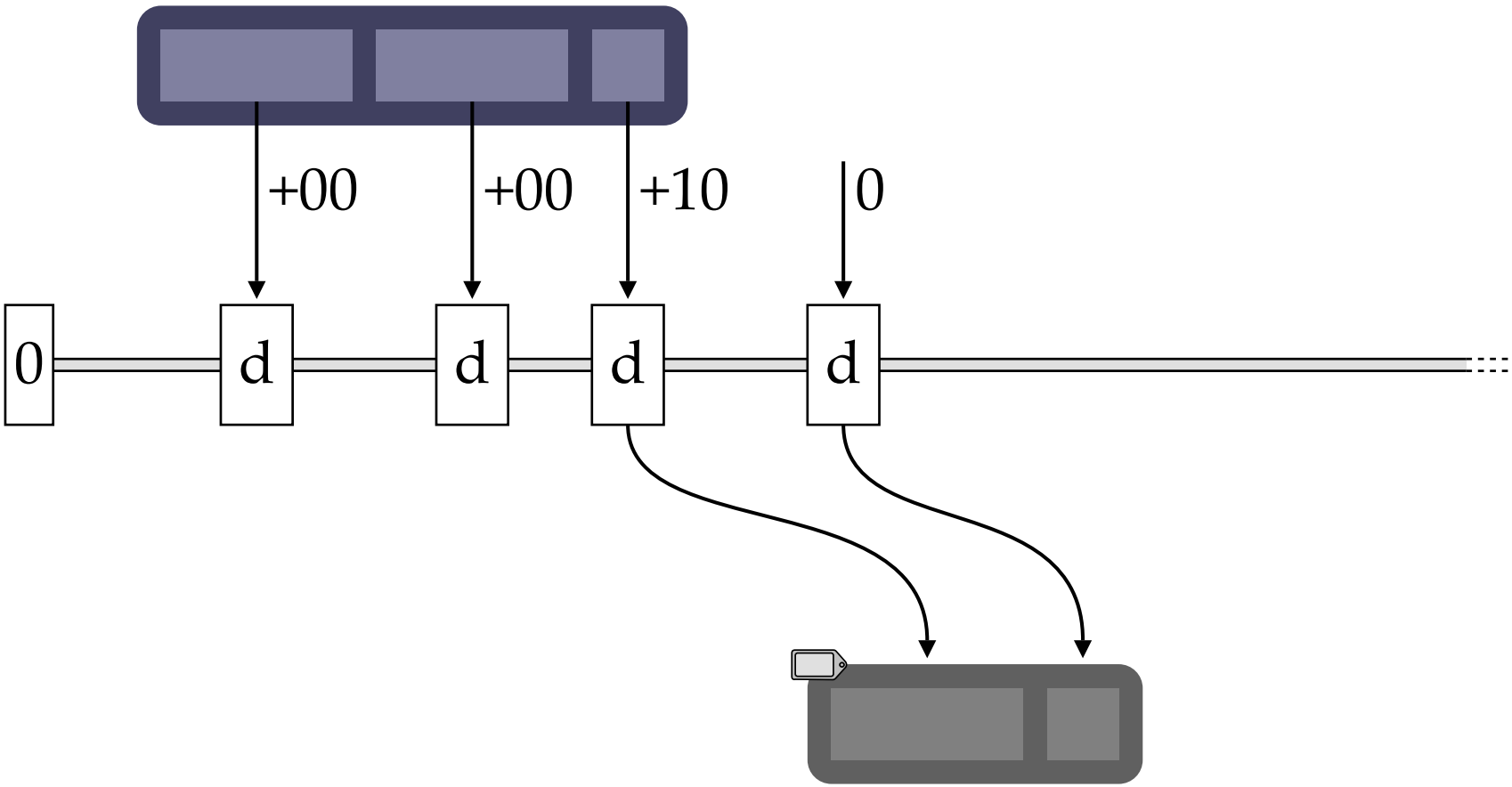


$A^{(1)}$ contains the key and must be unique, e.g.,

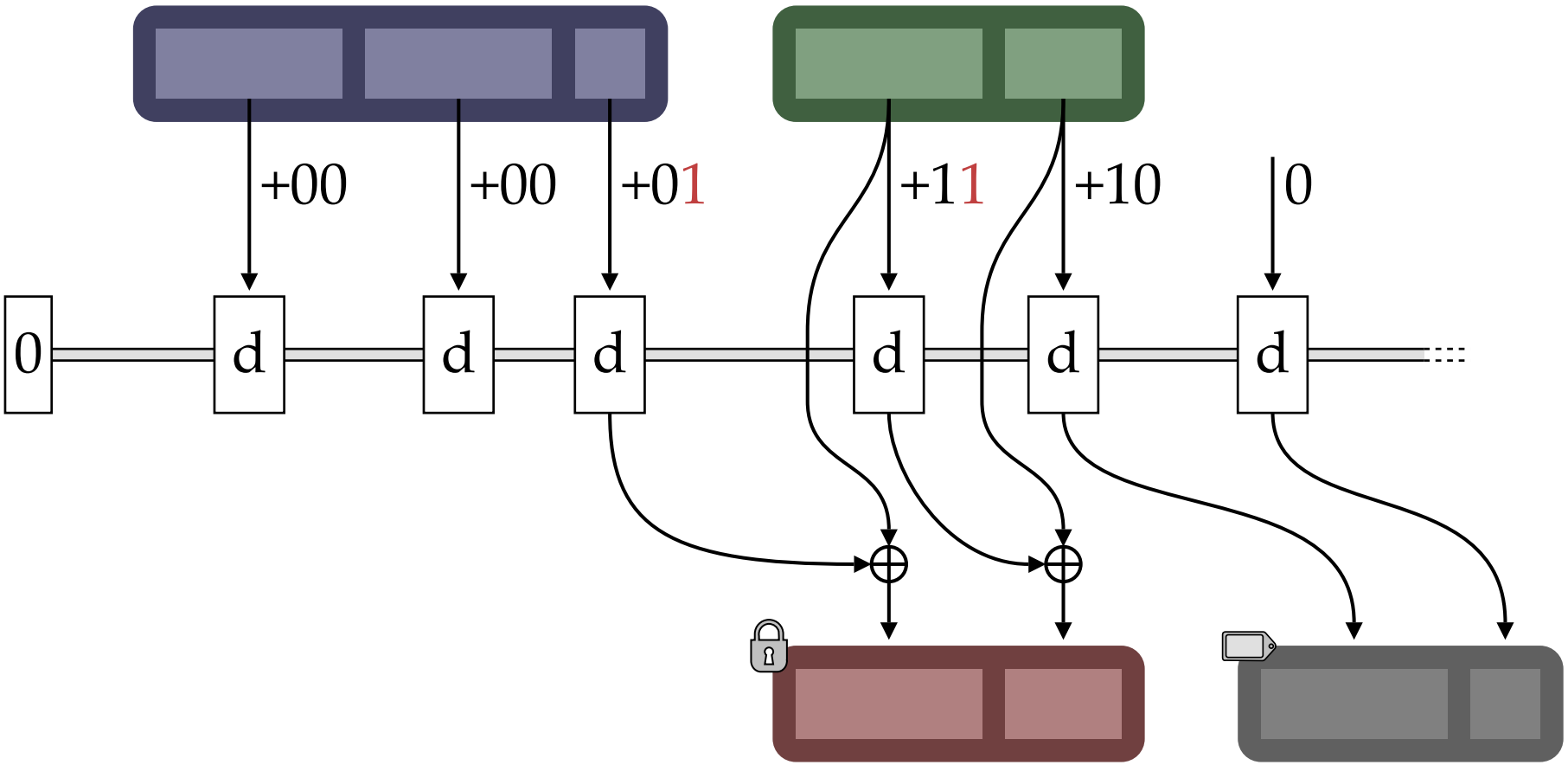
- $A^{(1)}$ contains a session key used only once;
- $A^{(1)}$ contains a key and a nonce.

In general: $A^{(1)} = \text{key}||\text{nonce}||\text{associated data}$.

Inside DuplexWrap



Inside DuplexWrap



Keyak instances and efficiency

Name	Width b	Parallelism P
Ocean Keyak	1600	4
Sea Keyak	1600	2
Lake Keyak	1600	1
River Keyak	800	1

- Processing for Lake Keyak
  - long messages: about 50 % of SHAKE128
  - short messages: 24 rounds
- Working memory footprint
  - reasonable on high- and middle-end platforms
  - not ideal on constrained platforms

Security of KEYAK

Generic security of KEYAK thanks to a combination of results:

- Sound tree hashing modes [IJIS 2013] for parallelized modes
- Keyed sponge indistinguishability [SKEW 2011 + work in progress]
- SPONGEWRAP generic security [SAC 2011], adapted to DUPLEXWRAP

Safety margin against shortcut attacks:

- Practical attacks up to 6 rounds [Dinur et al. SHA-3 2014]
- Academic attacks up to 9 rounds [Dinur et al. SHA-3 2014]

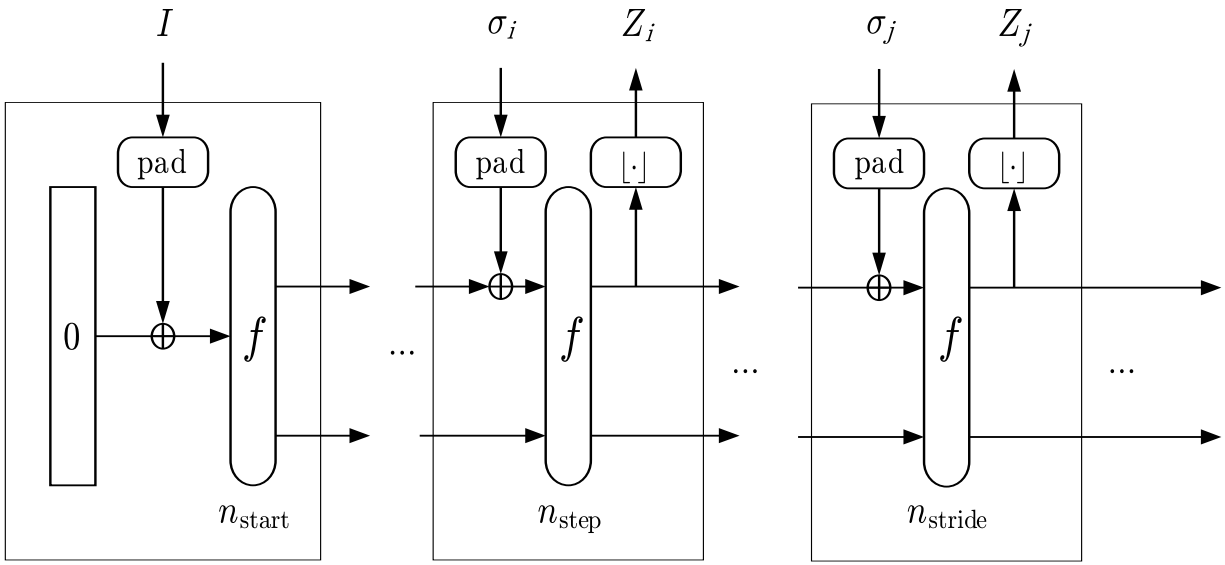
Outline

KETJE goals

- Nonce-based AE function
- 96-bit or 128-bit security (incl. multi-target)
- Sequence of header-body pairs
  - keeping the state during the session
- Small footprint
- Target niche: secure channel protocol on secure chips
  - banking card, ID, (U)SIM, secure element, FIDO, etc.
  - secure chip has strictly incrementing counter
- Using reduced-round KECCAK-$f[400]$ or KECCAK-$f[200]$, to allow
  - implementation re-use
  - cryptanalysis re-use
  - reasonable side-channel protections

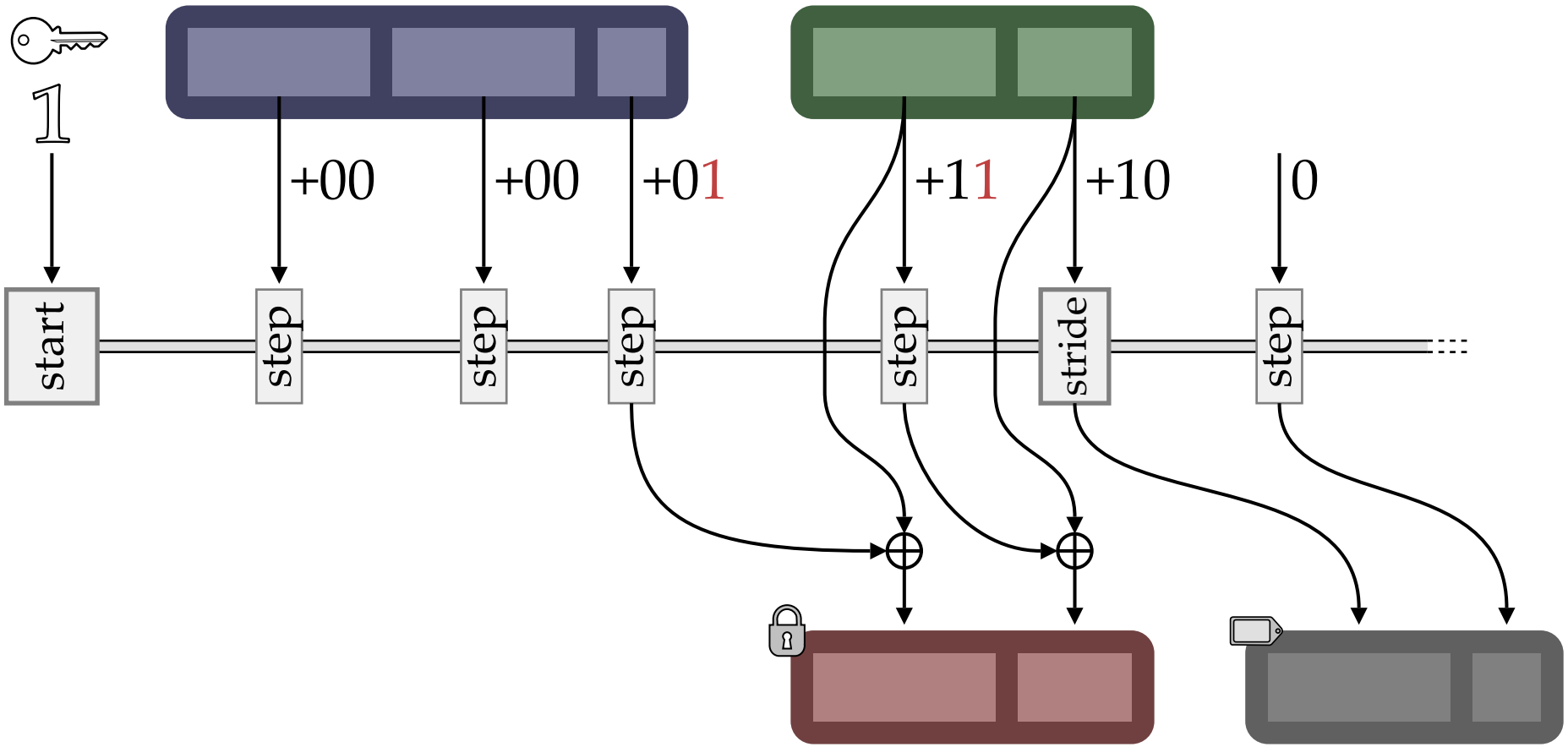
(... and because we like it ...)

Inside KETJE: the MONKEYDUPLEX layer



- $n_{\text{start}} = 12$ rounds should provide strong instance separation
- $n_{\text{step}} = 1, r = 2b/25$ should avoid single-instance state retrieval
- $n_{\text{stride}} = 6$ rounds should avoid a forgery with one instance

Inside MonkeyWrap



KETJE instances and lightweight features

feature		KETJE JR	KETJE SR
state size		25 bytes	50 bytes
block size		2 bytes	4 bytes
processing		**computational cost**	
initialization	per session	12 rounds	12 rounds
wrapping	per block	1 round	1 round
8-byte tag comp.	per message	9 rounds	7 rounds

Outline

Current developments

- Optimized software implementations
  - Gross estimations can be derived from KECCAK
  - LAKE KEYAK expected twice faster than SHAKE128
  - There might be interesting improvement with new AVX512 (VPTERNLOG, rotations and 32 registers)
- Hardware implementations

Conclusions

Thanks for your attention!