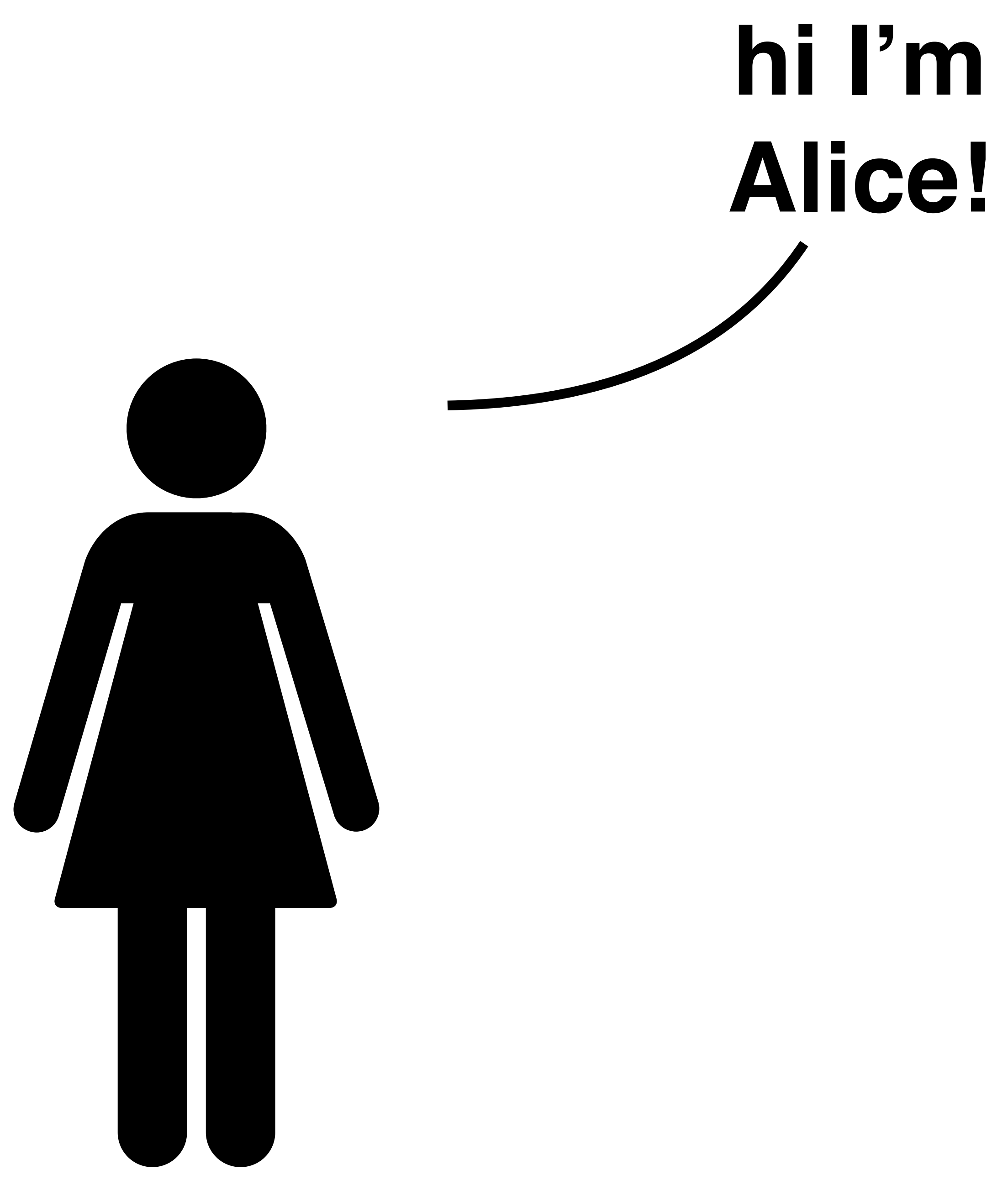
Analysis & Design of Lightweight Authenticated Encryption Schemes

supervised by **Marine Minier**

Paul Huynh | November, 26 2020 | virtual defense

Part I
Alice, Bob & the IoT

The story of Alice & Bob

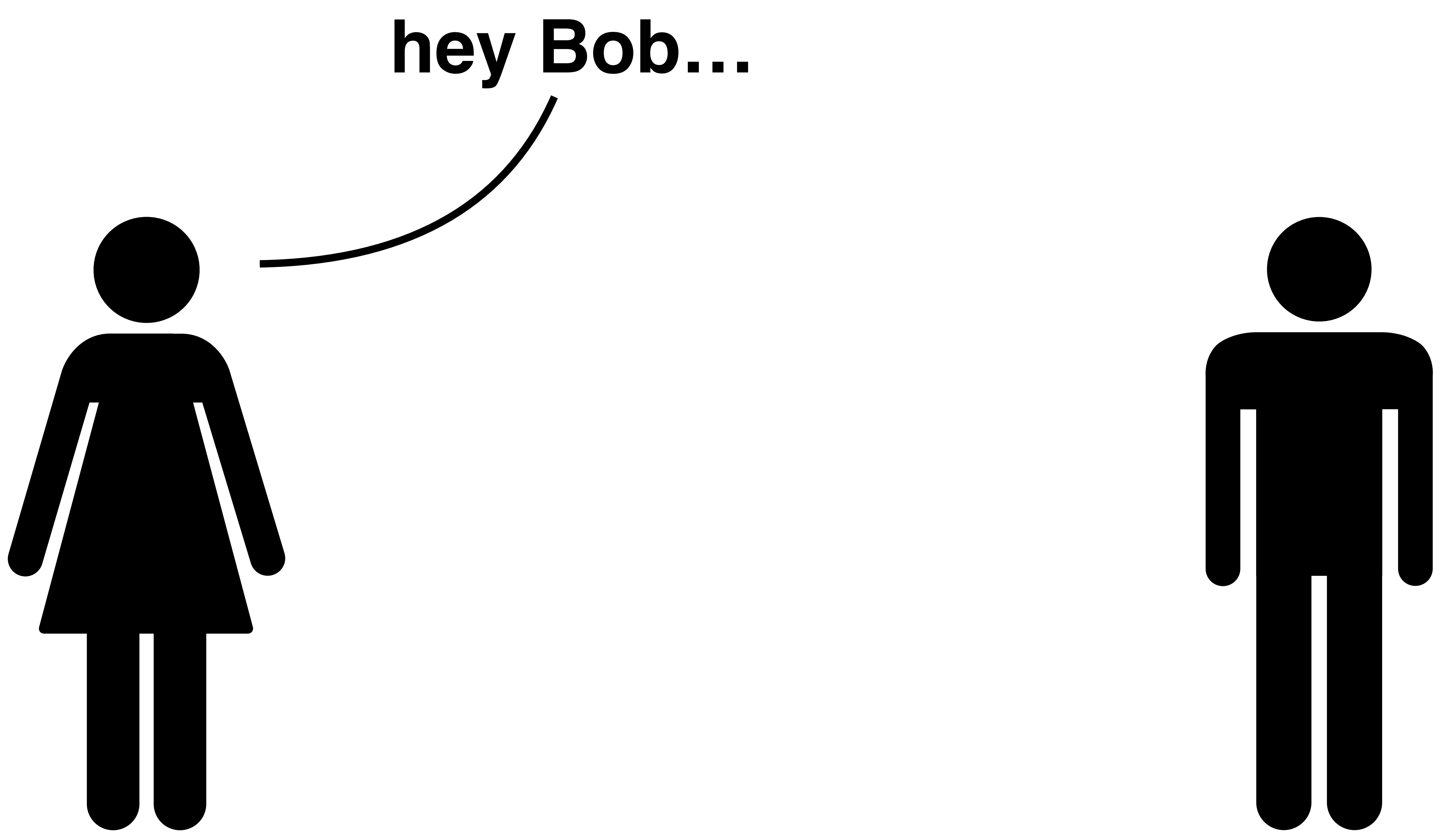


Meet Alice.

The story of Alice & Bob

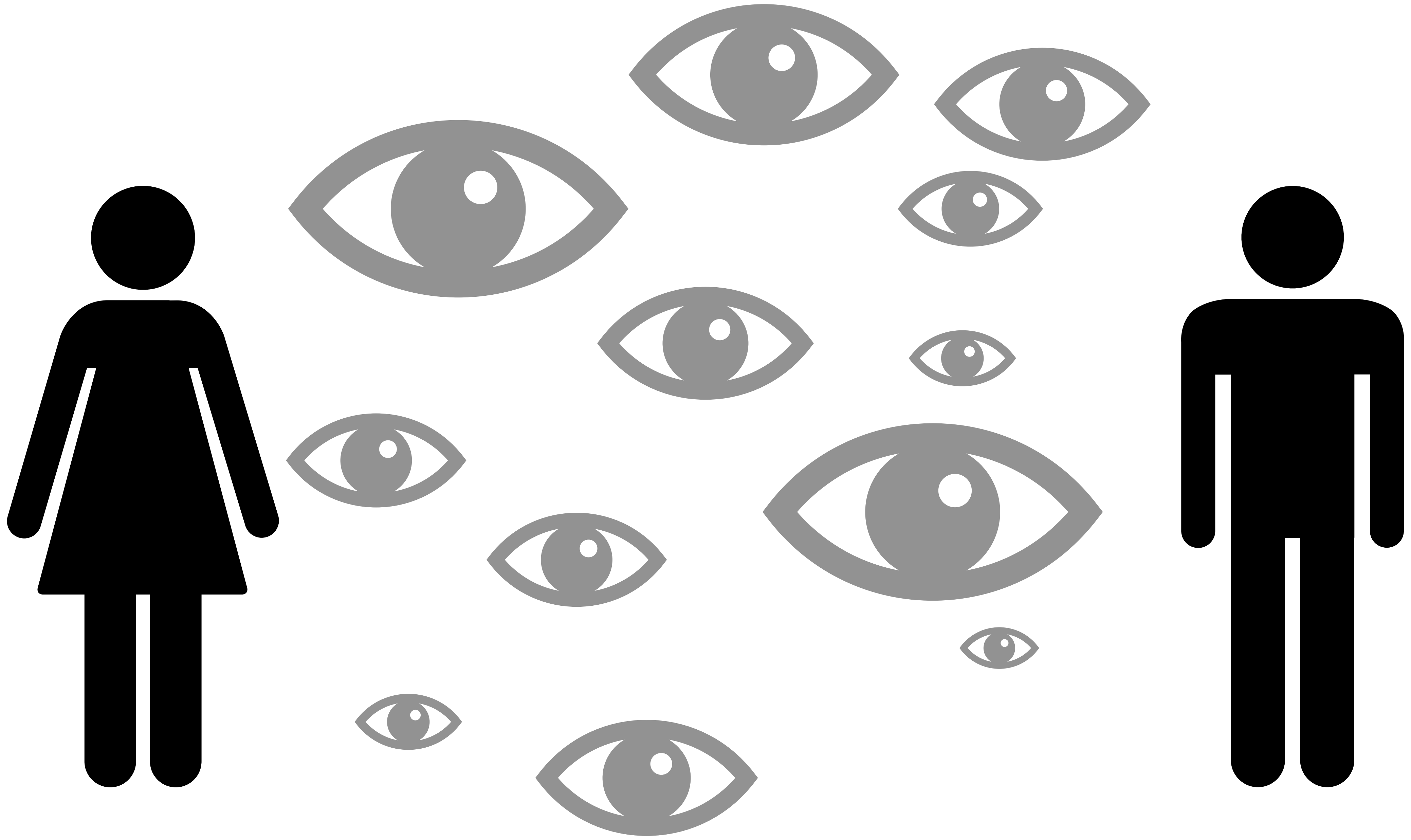
Meet Bob.

The story of Alice & Bob



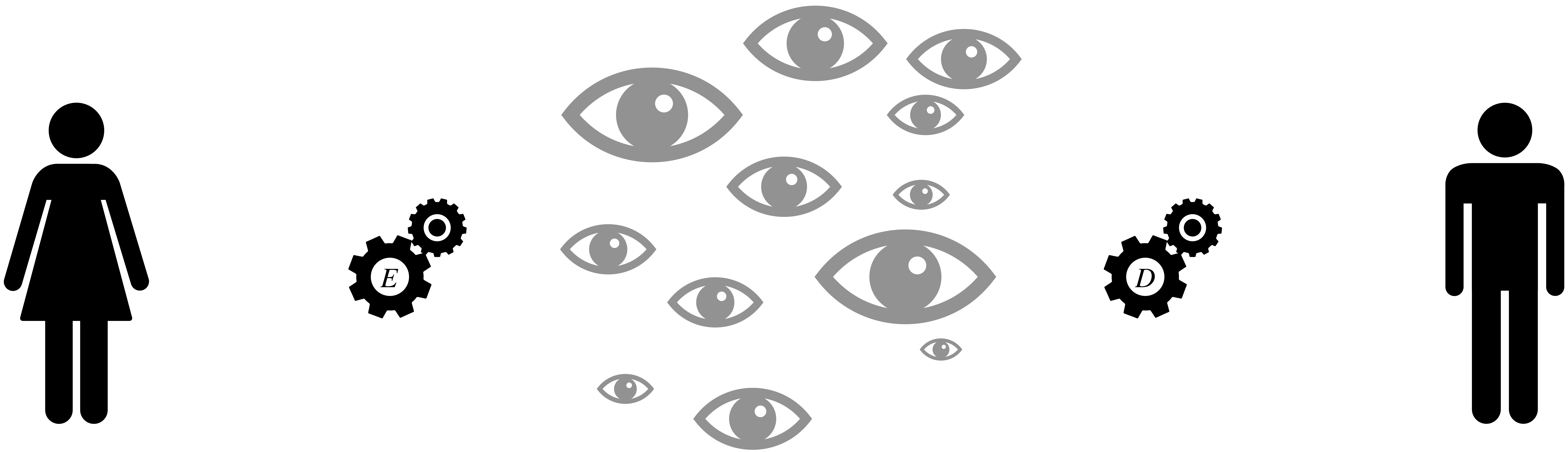
Alice wants to send Bob a message...

The story of Alice & Bob



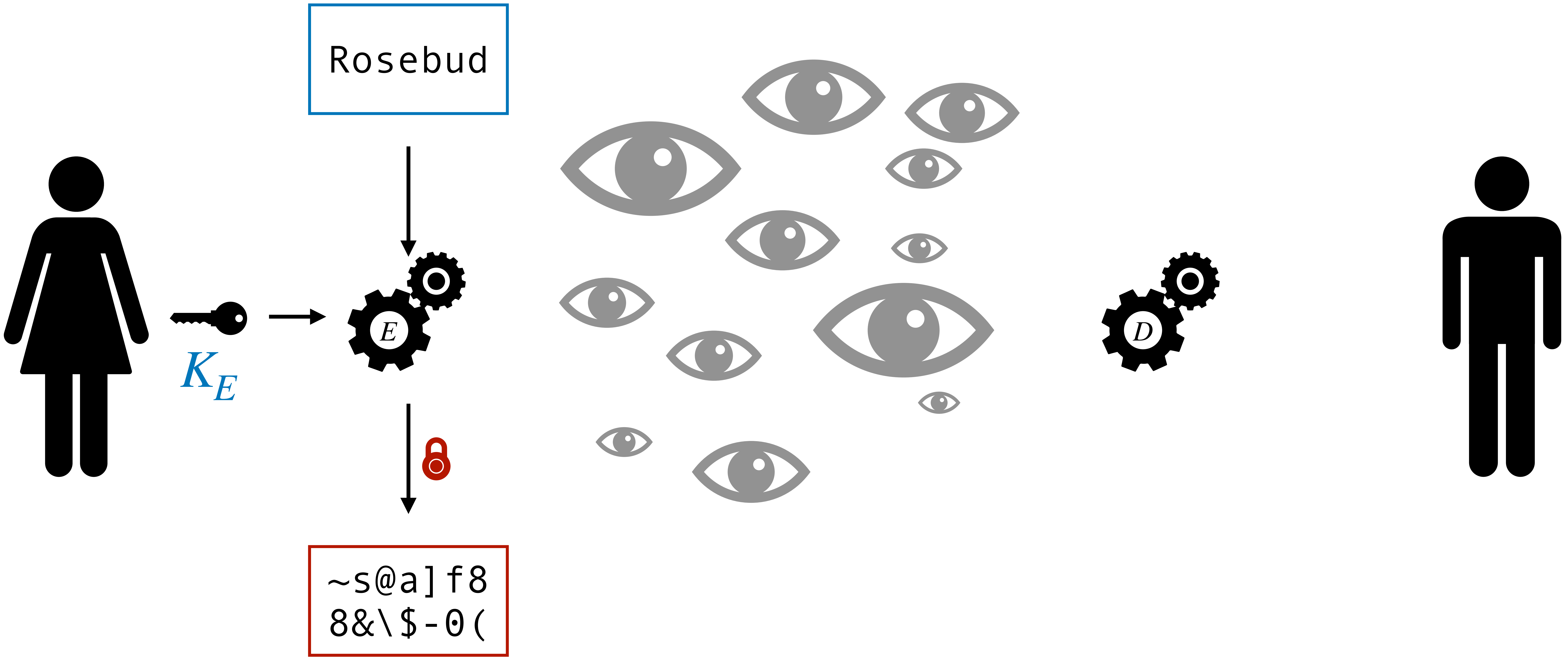
...but the channel is not secure.

The story of Alice & Bob



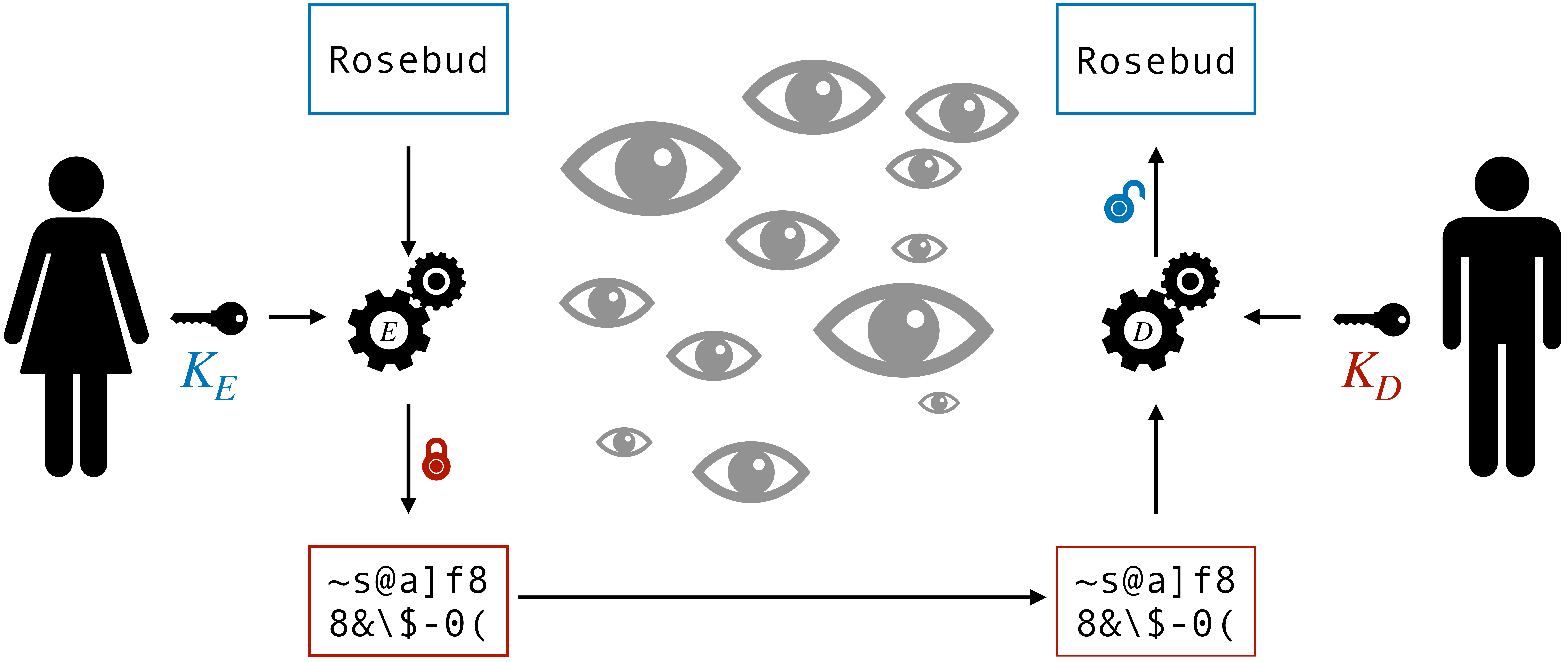
They need **encryption**.

The story of Alice & Bob



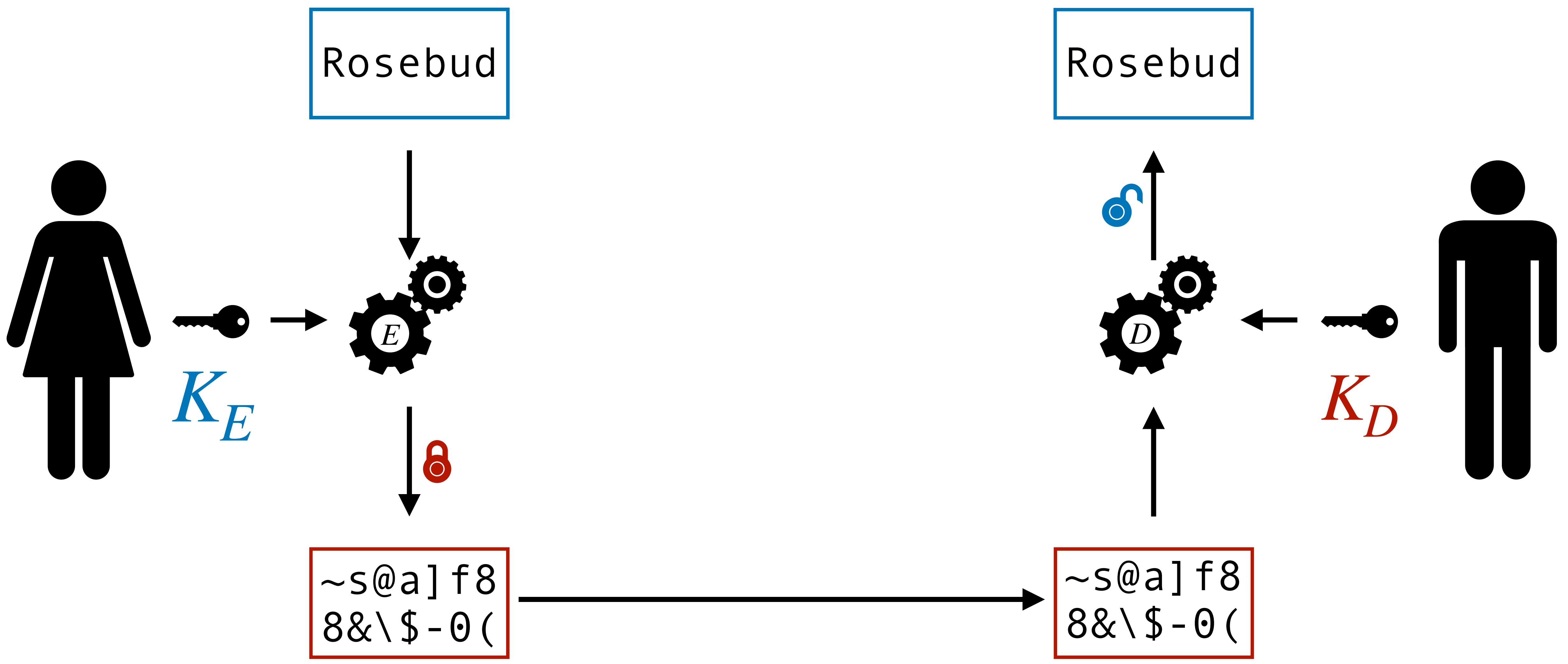
Encryption is parameterized by a key K_E.

The story of Alice & Bob



Decryption depends on a secret K_D associated to K_E.

The story of Alice & Bob



Private-key cryptography

$K_D = K_E = K$

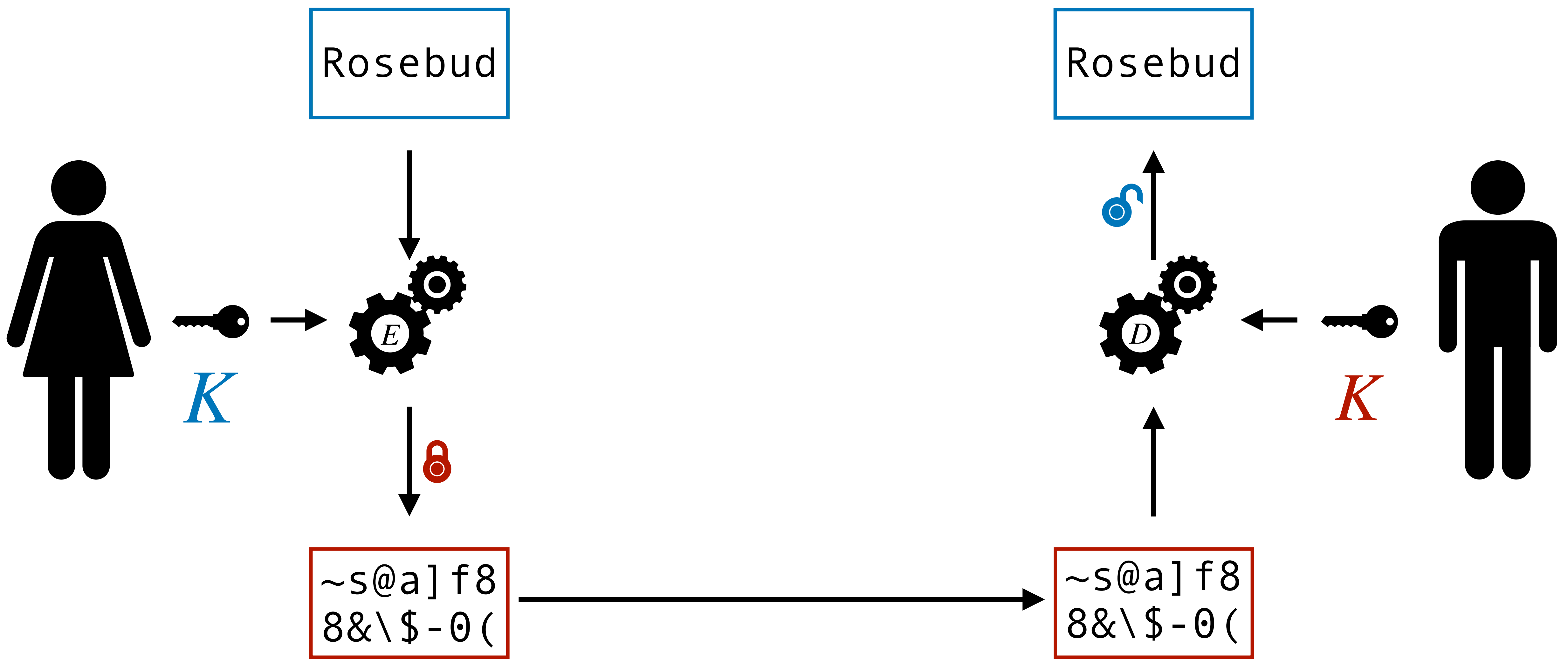
K is a shared secret

Public-key cryptography

$K_D \neq K_E$

K_E is public

The story of Alice & Bob



Private-key cryptography

$K_D = K_E = K$

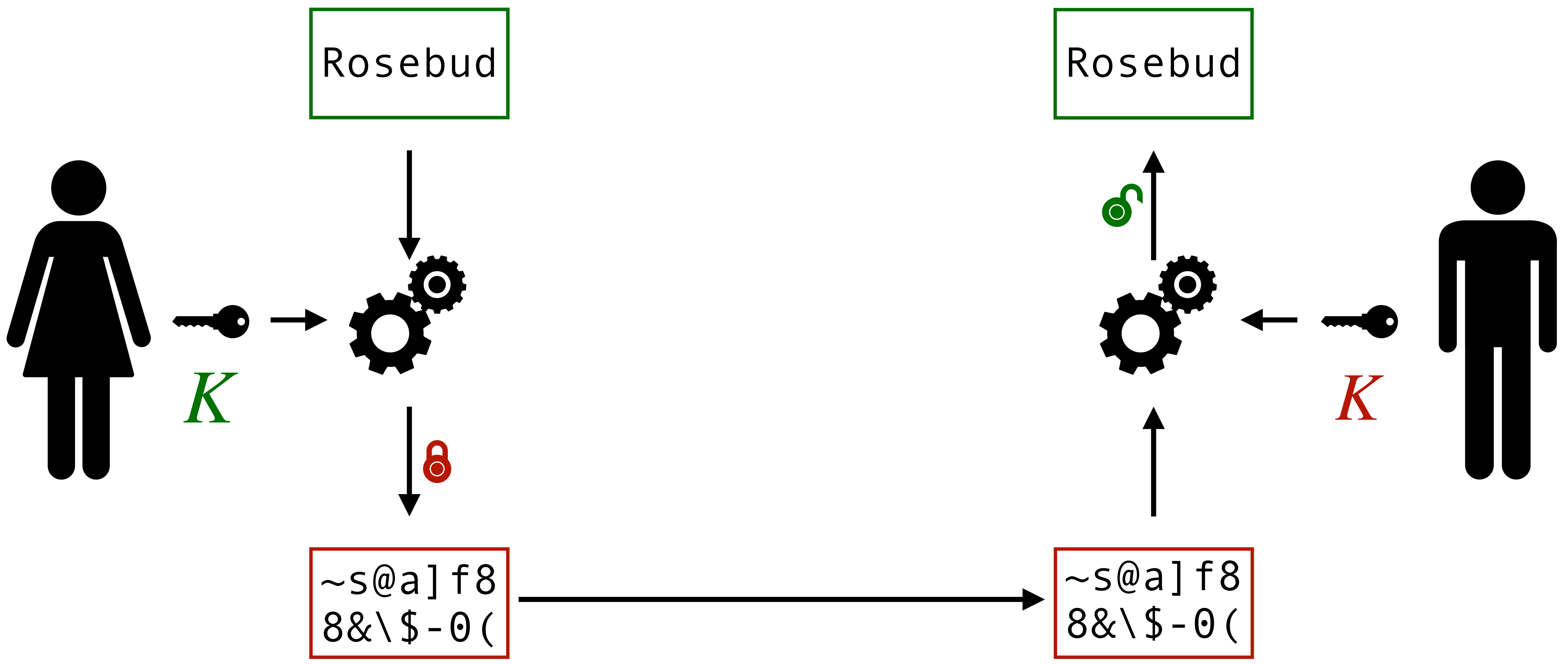
K is a shared secret

Public-key cryptography

$K_D \neq K_E$

K_E is public

The story of Alice & Bob



Private-key cryptography

$K_D = K_E = K$

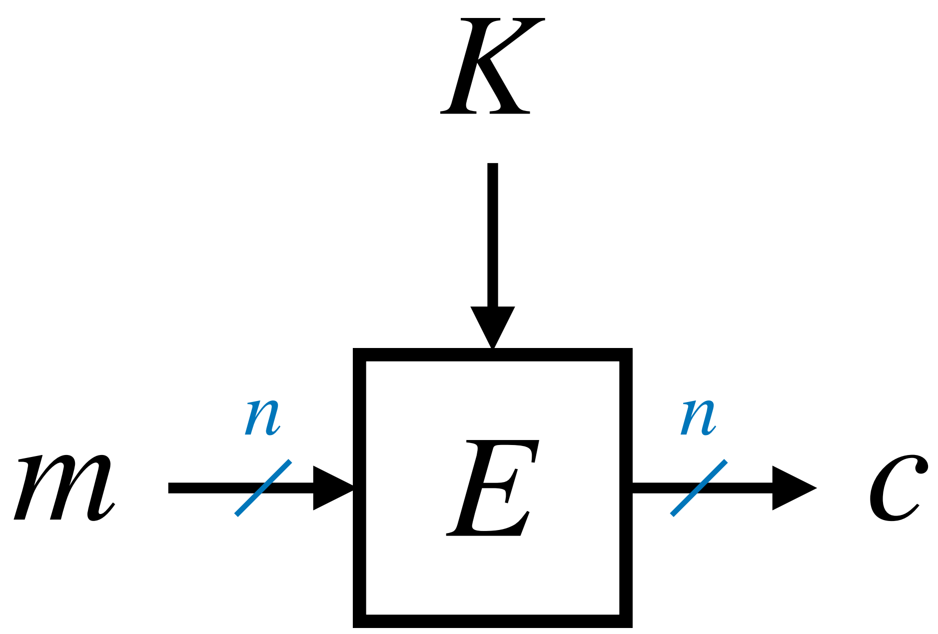
K is a shared secret

Block ciphers

Stream ciphers

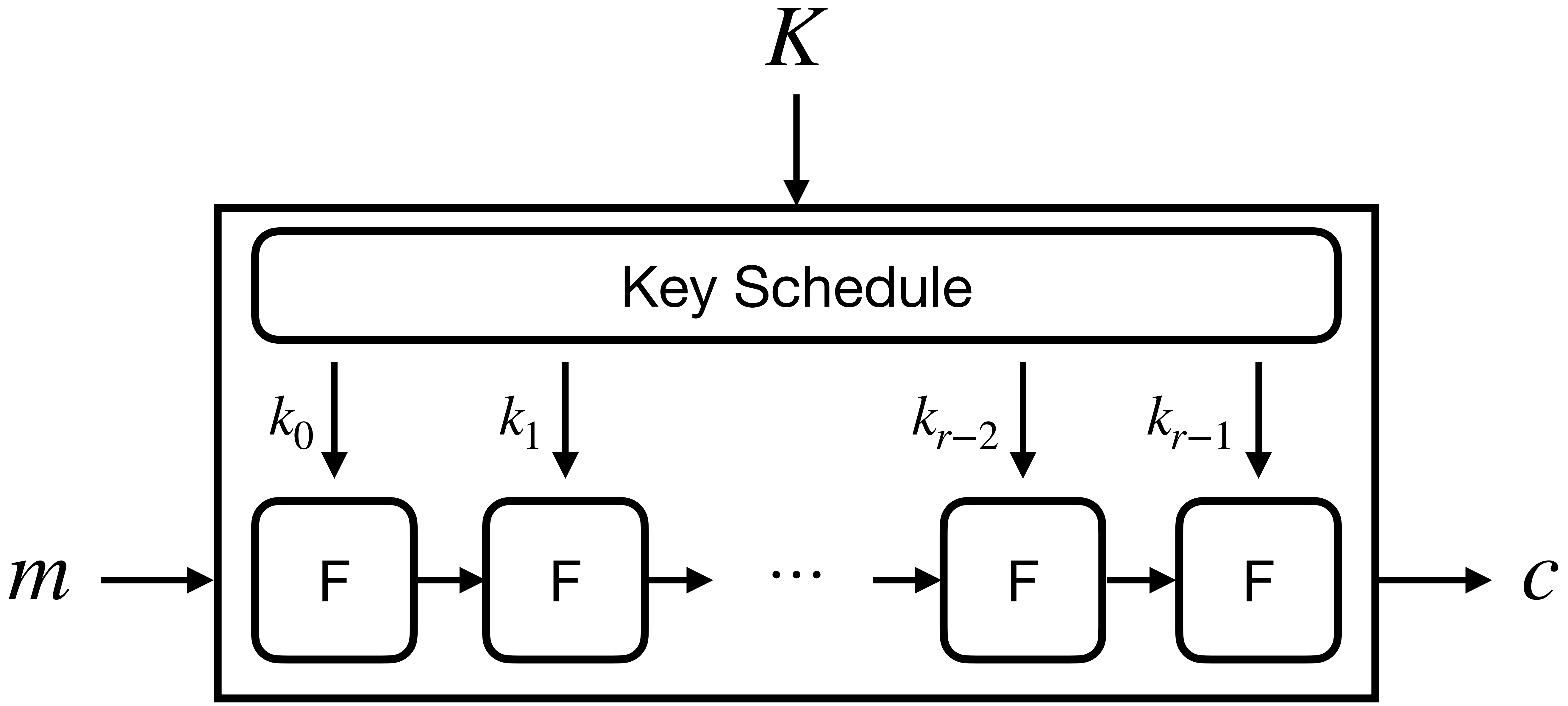
Block Ciphers

A block cipher with block size n and key size k is a family of 2^k permutations of n bits $(E_K)_{K \in \mathbb{F}_2^k}$, indexed by a key $K \in \mathbb{F}_2^n$.



Combined with a **mode of operation** describing how $(E_K)_{K \in \mathbb{F}_2^k}$ can be used for encrypting messages of any length.

Iterated Block Ciphers

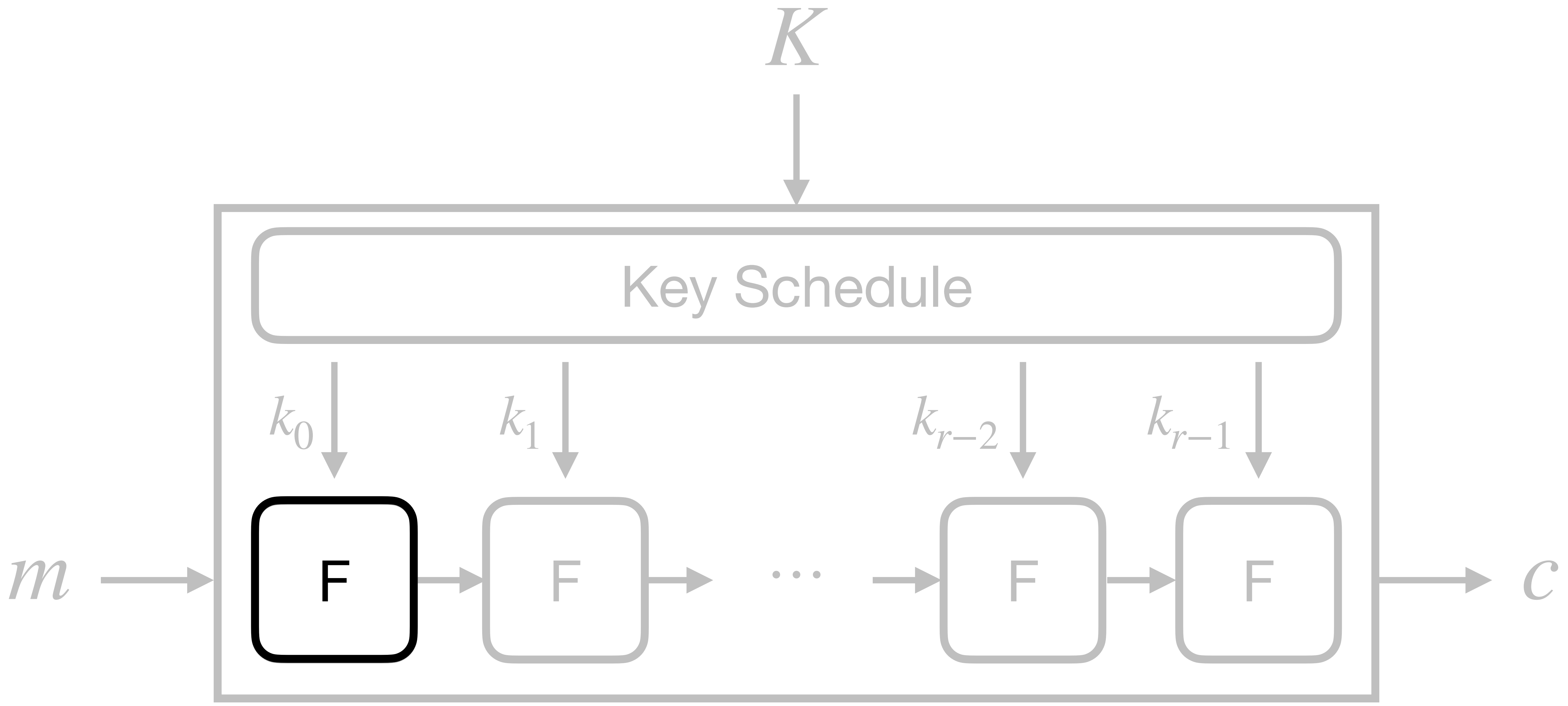


$$c = E_K(m) = \mathsf{F}_{k_{r-1}} \circ \ldots \circ \mathsf{F}_{k_0}(m)$$

F is the same keyed permutation of $\mathbb{F}_2^n$

- simple analysis
- cost-effective implementation

Iterated Block Ciphers

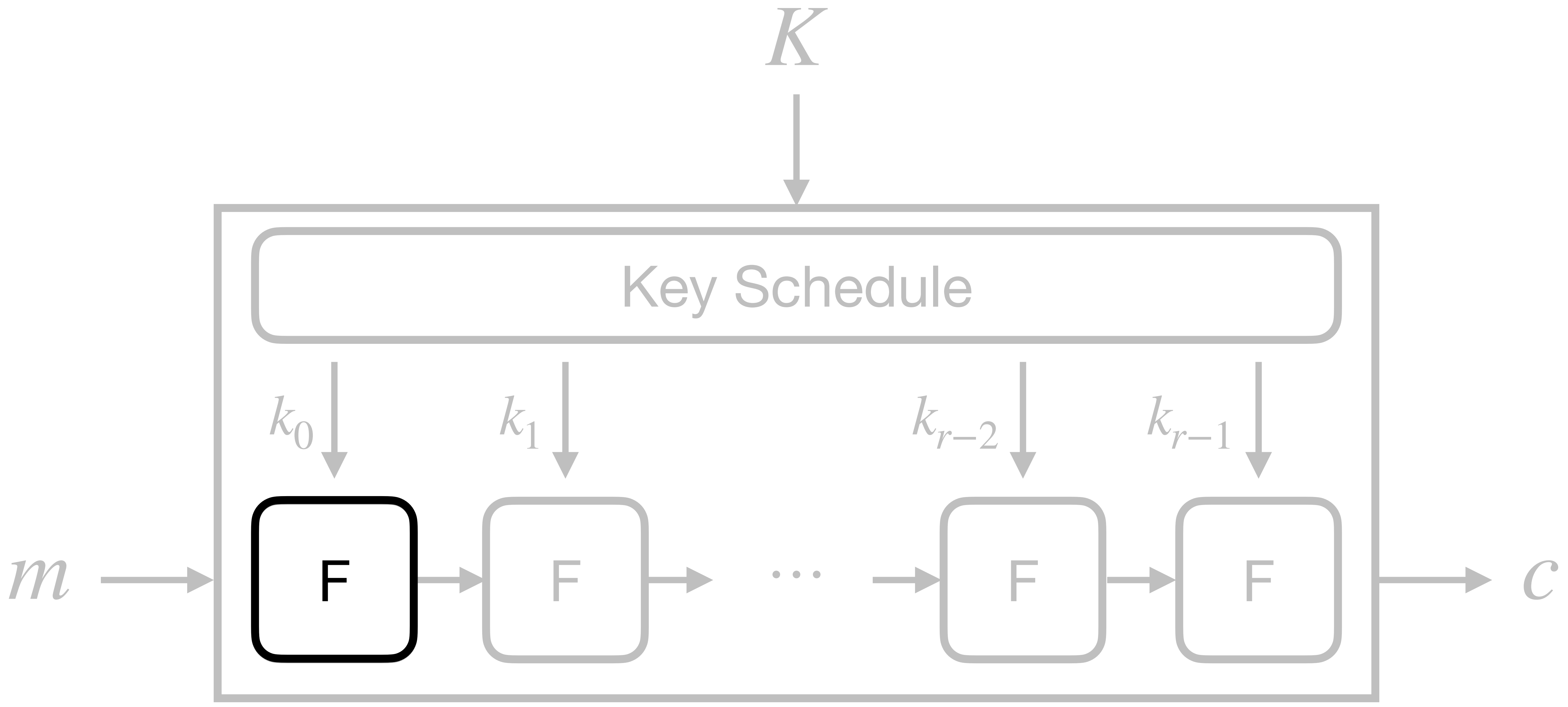


$$c = E_K(m) = \mathsf{F}_{k_{r-1}} \circ \ldots \circ \mathsf{F}_{k_0}(m)$$

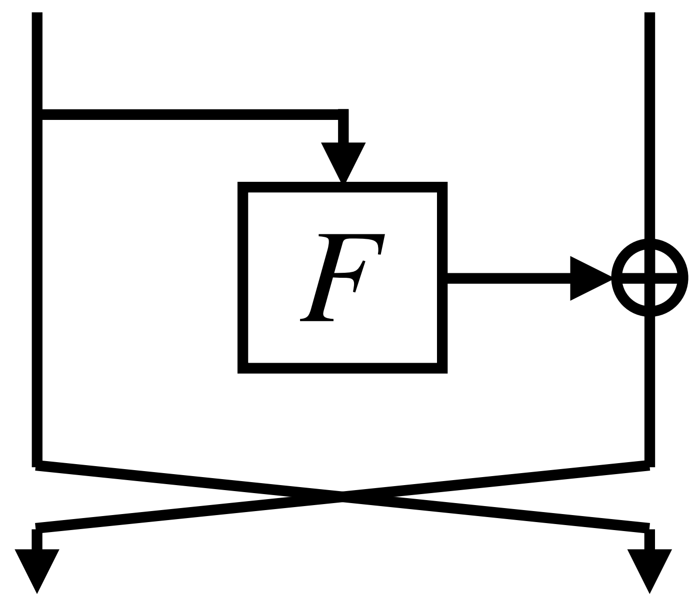
F is the same keyed permutation of $\mathbb{F}_2^n$

- simple analysis
- cost-effective implementation

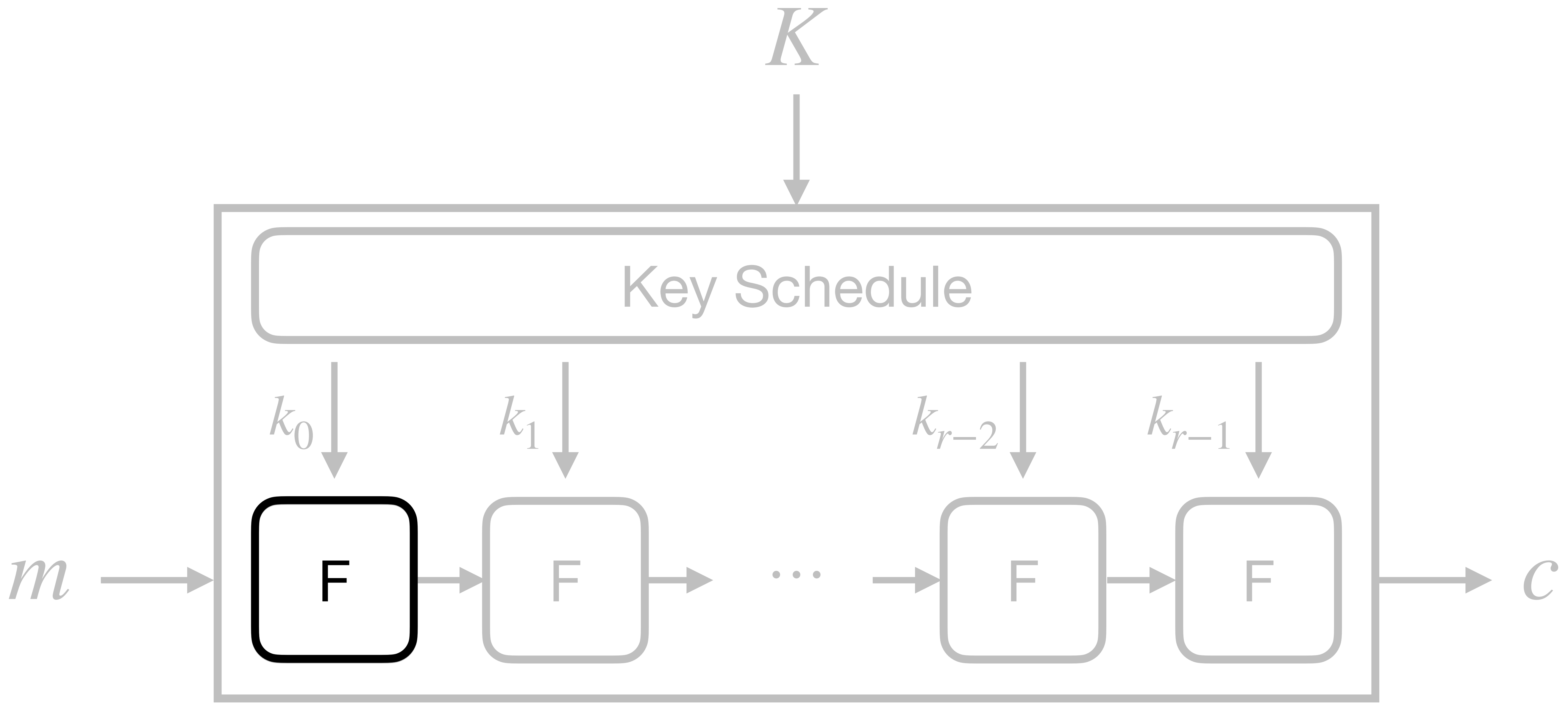
Iterated Block Ciphers



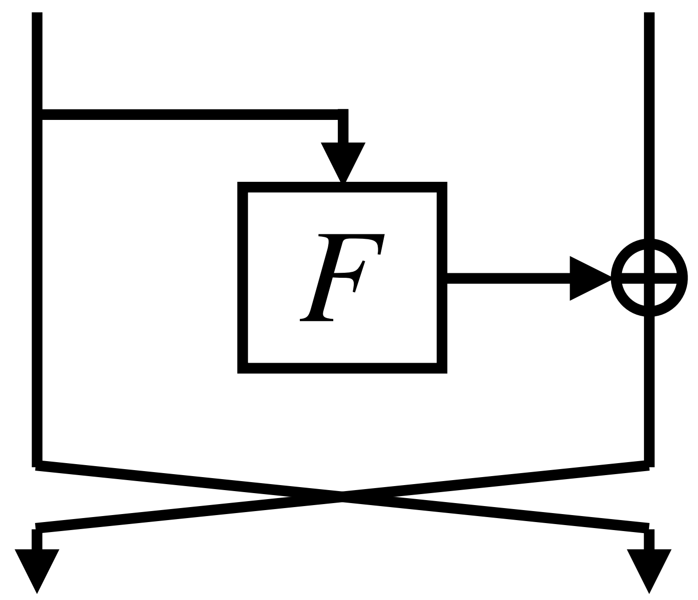
Feistel Networks



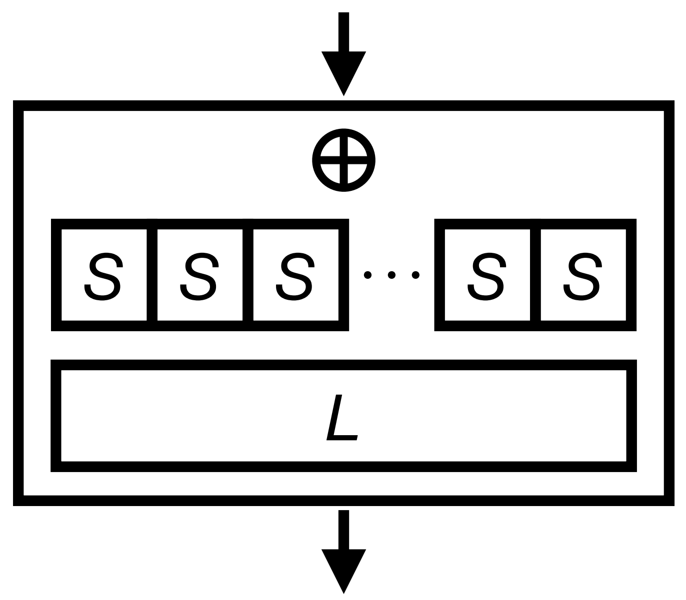
Iterated Block Ciphers



Feistel Networks



Substitution-Permutation Networks



Feistel Networks

Block Cipher Cryptographic System
Feistel, 1974

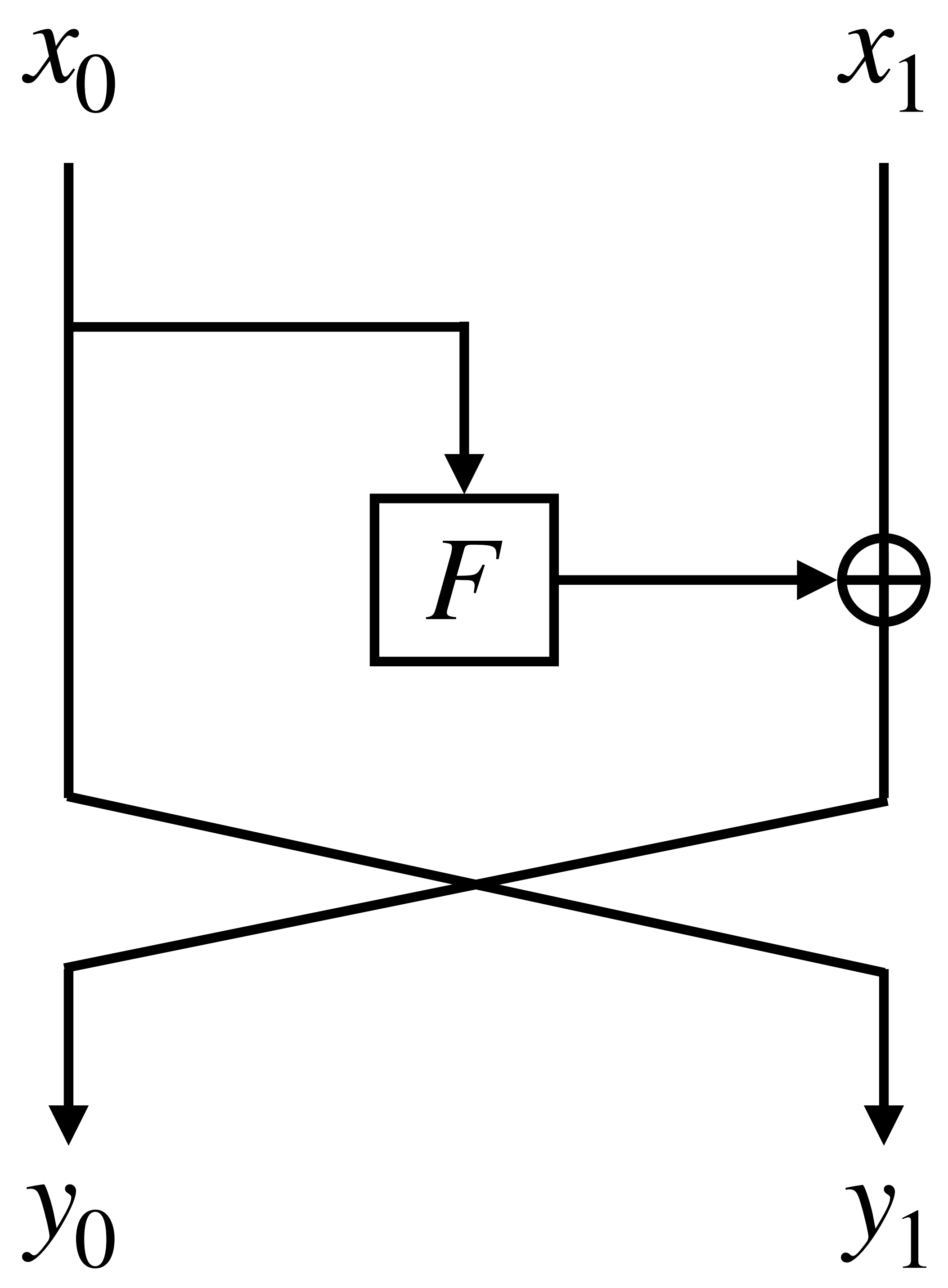
- State split into two halves:

$$y_1 = x_0$$
$$y_0 = x_1 \oplus F_k(x_0)$$

- Invertible even if the **Feistel function** F is not.

- Decryption is the same up to the permutation of the two halves
  → Reduced code size / circuitry

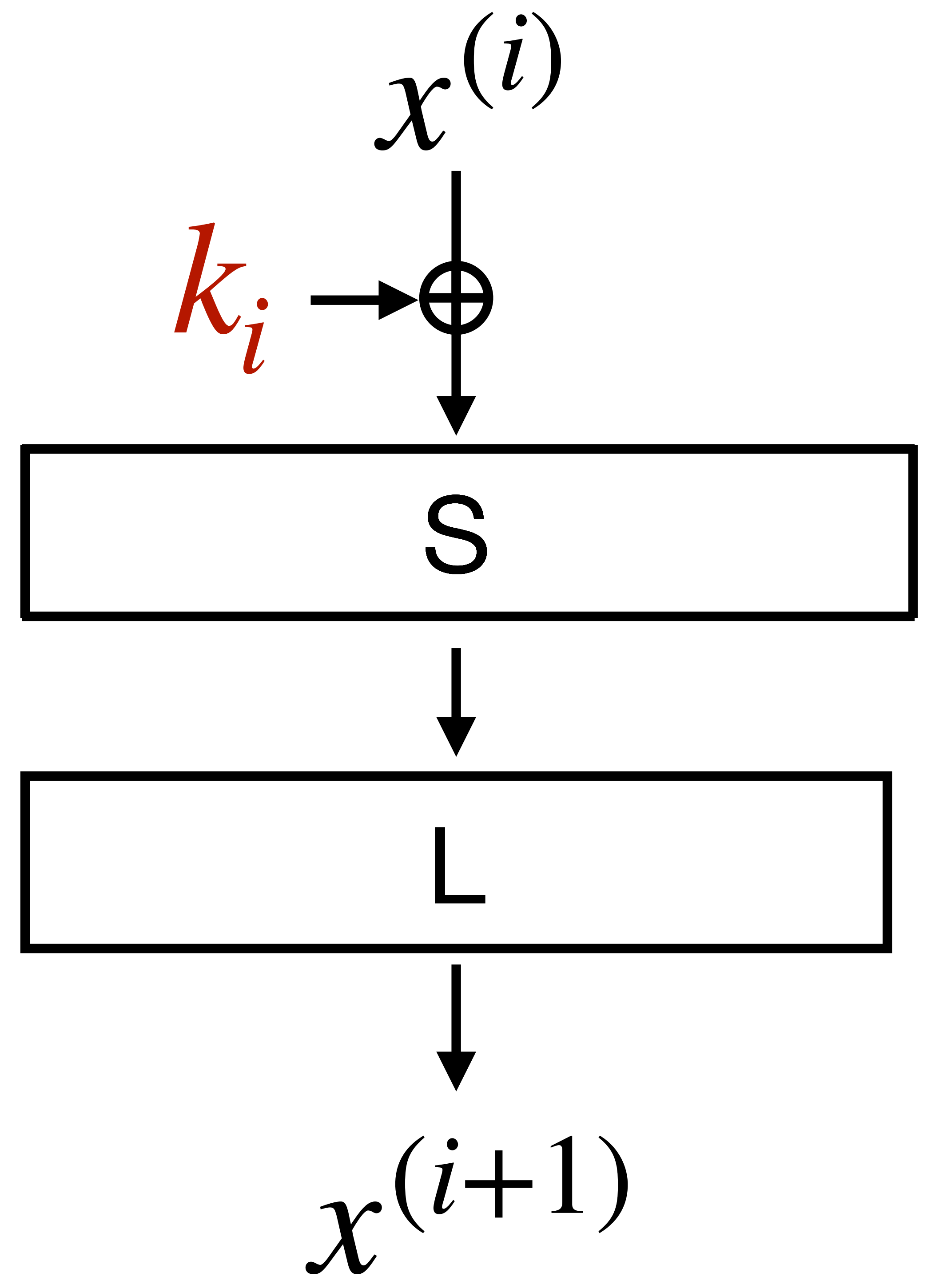
- Variants
  Generalized Feistel Networks
  [Zheng, Matsumoto & Imai, 89][Nyberg, 96]
  Extended Generalized Feistel Networks
  [Berger, Minier & Thomas, 14]



Substitution-Permutation Networks (SPN)

1. **Nonlinear layer S**
   for confusion

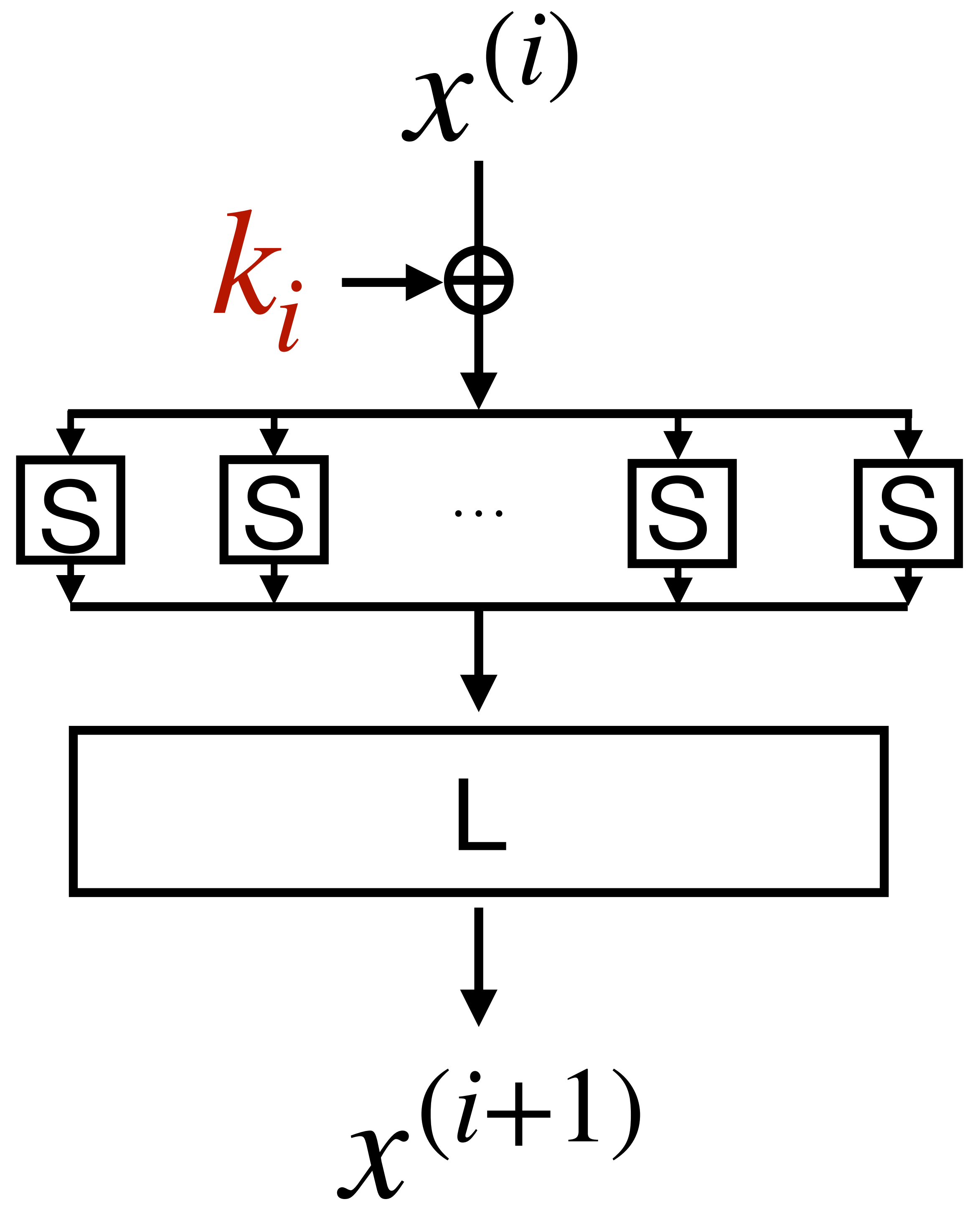
2. **Linear layer L**
   for diffusion



Substitution-Permutation Networks **(SPN)**

1. Small **substitution-based** permutations S for confusion

2. **Linear layer L** for diffusion

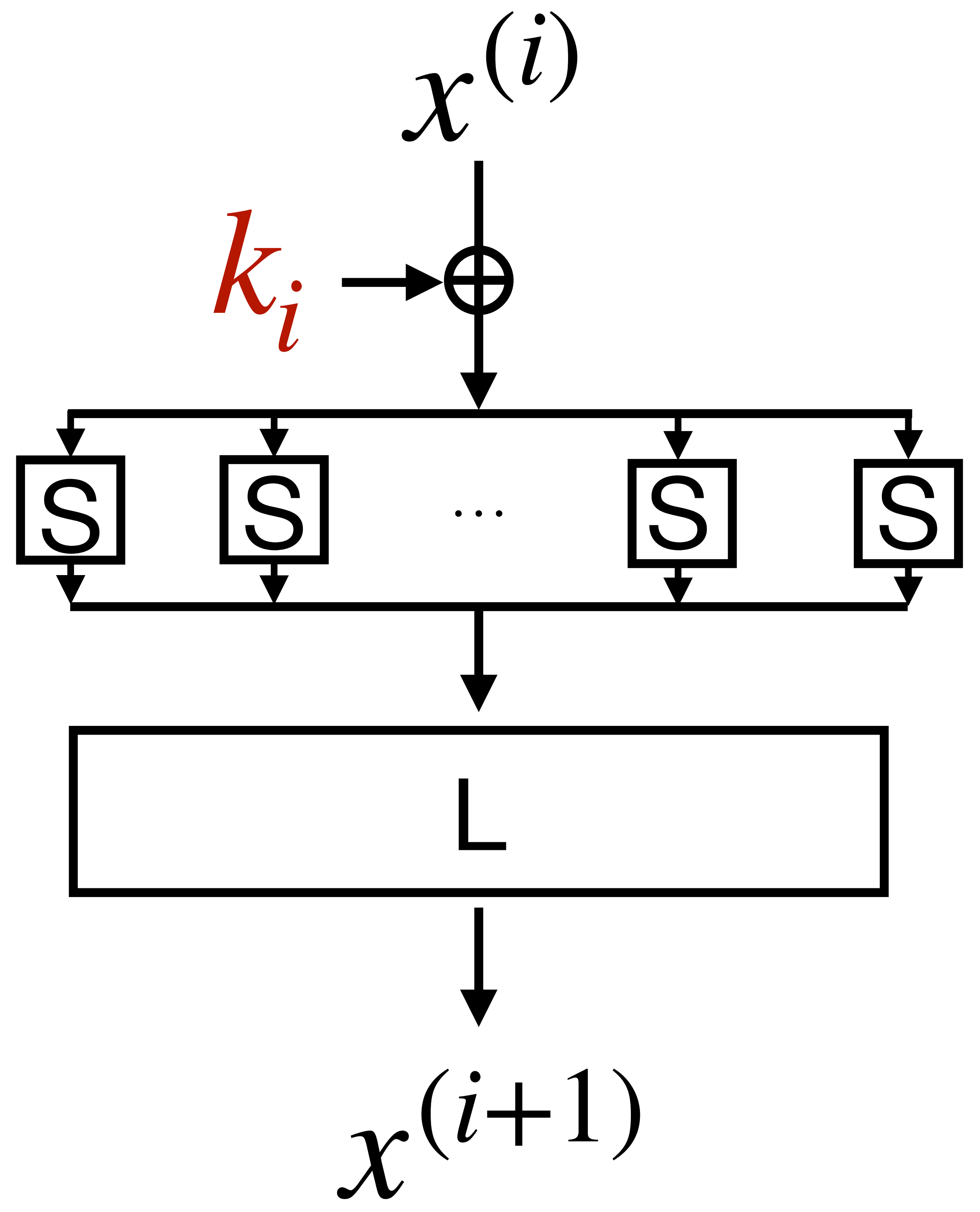


Substitution-Permutation Networks (SPN)

1. Small **substitution-based** permutations S for confusion

2. **Linear layer L** for diffusion

e.g. AES [Daemen, Rijmen 98] [FIPS PUB 197]



But…

"Alexa, what is lightweight cryptography?"

New applications/concepts

Internet of Things (IoT)
e.g. healthcare monitoring systems, automated management of supply chain, public transportation, driving assistance systems, smart home appliances

New constraints

Hardware: area, latency, throughput, power/energy consumption etc.
Software: execution time, latency, memory (ROM/RAM) requirements

But…

“Alexa, what is lightweight cryptography?”

New applications/concepts

Internet of Things (IoT)
e.g. healthcare monitoring systems, automated management of supply chain, public transportation, driving assistance systems, smart home appliances

New constraints

Hardware: area, latency, throughput, power/energy consumption etc.
Software: execution time, latency, memory (ROM/RAM) requirements

Need for cryptographic solutions tailored to constrained devices.

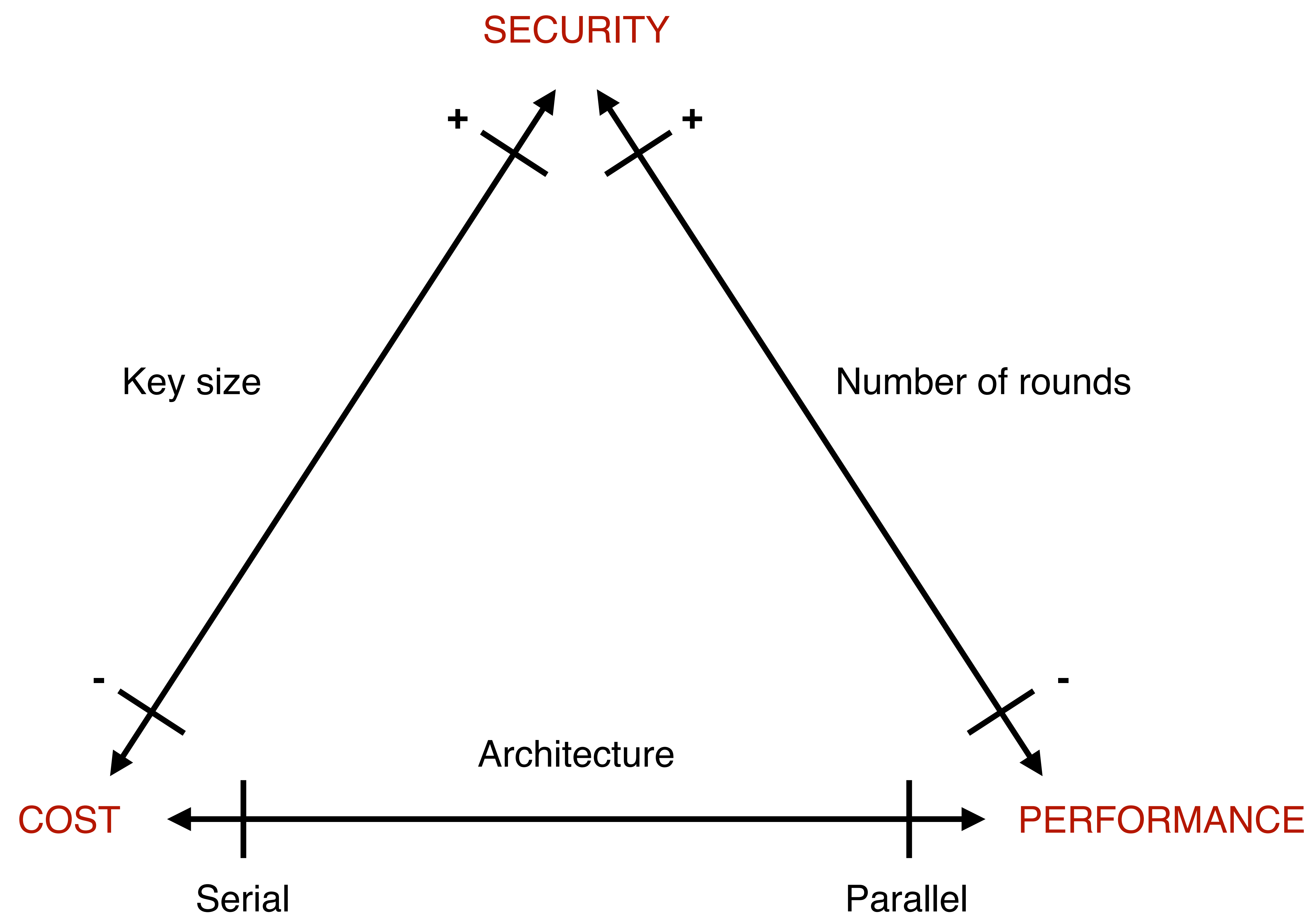
New Dedicated Designs

- **Smaller parameters**
  block sizes = 64 or 80 bits
  key length = 80, 96, 112 bits

- Many iterations of **simple round functions**, simple operations
  e.g. binary diffusion layer, 4-/3-bit S-Boxes, bit permutations

- Simplified key schedules

Many proposals
e.g. Present, Skinny, Simon, Speck



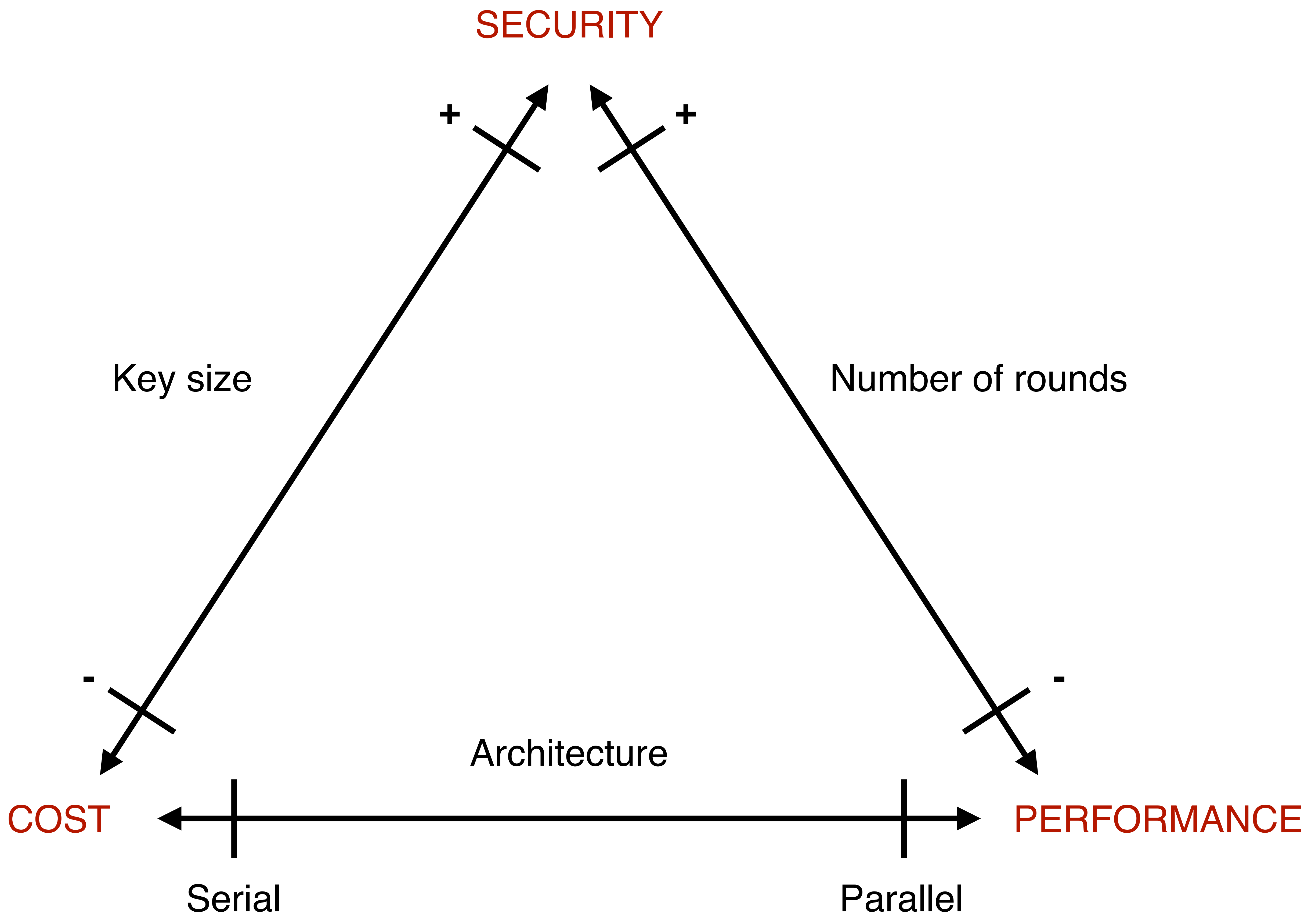
New Dedicated Designs

- **Smaller parameters**
  block sizes = 64 or 80 bits
  key length = 80, 96, 112 bits

- Many iterations of **simple round functions**, simple operations
  e.g. binary diffusion layer, 4-/3-bit S-Boxes, bit permutations

- Simplified key schedules

Many proposals
e.g. Present, Skinny, Simon, Speck



Which one should we use ?

NIST's standardization process

National Institute of Standards and Technology

- US standardization authority
- AES (1997-2000)
  SHA-3 (2007-2012)
  Post-quantum cryptography (since 2017)

NIST's standardization process

March 2017	NISTIR 8114, *Report on Lightweight Cryptography* Announcement of an open process to create a portfolio of lightweight cryptographic standards.
August 2018	**Call for algorithms.** Deadline for packages submissions: March 27, 2019.
April 2019	**Round 1** 57 submissions received, 56 selected
August 2019	**Round 2** 32 candidates remaining

Contributions

1. **Lilliput-AE: a New Lightweight Tweakable Block cipher for AEAD**
Alexandre Adomnicai, Thierry P. Berger, Christophe Clavier, Julien Francq, Paul Huynh, Virginie Lallemand, Kévin Le Gouguec, Marine Minier, Léo Reynaud and Gaël Thomas **[NIST LWC proposal]**

2. **Cryptanalysis Results on Spook**
Patrick Derbez, Paul Huynh, Virginie Lallemand, María Naya-Plasencia, Léo Perrin and André Schrottenloher **[CRYPTO 2020]**

3. **Skinny with Scalpel: Comparing Tools for Differential Cryptanalysis**
Stéphanie Delaune, Patrick Derbez, Paul Huynh, Marine Minier, Victor Mollimard and Charles Prud'homme **[ePrint 2020/1402]**

4. **On the Feistel Counterpart of the Boomerang Connectivity Table**
Hamid Boukerrou, Paul Huynh, Virginie Lallemand, Bimal Mandal and Marine Minier **[ToSC 2020]**

5. **Non-Triangular Self-Synchronizing Stream Ciphers**
Julien Francq, Loïc Besson, Paul Huynh, Philippe Guillot, Gilles Millerioux and Marine Minier **[TC - minor revision]**

In this presentation

1. **Lilliput-AE: a New Lightweight Tweakable Block cipher for AEAD**
Alexandre Adomnicai, Thierry P. Berger, Christophe Clavier, Julien Francq, Paul Huynh, Virginie Lallemand, Kévin Le Gouguec, Marine Minier, Léo Reynaud and Gaël Thomas **[NIST LWC proposal]**

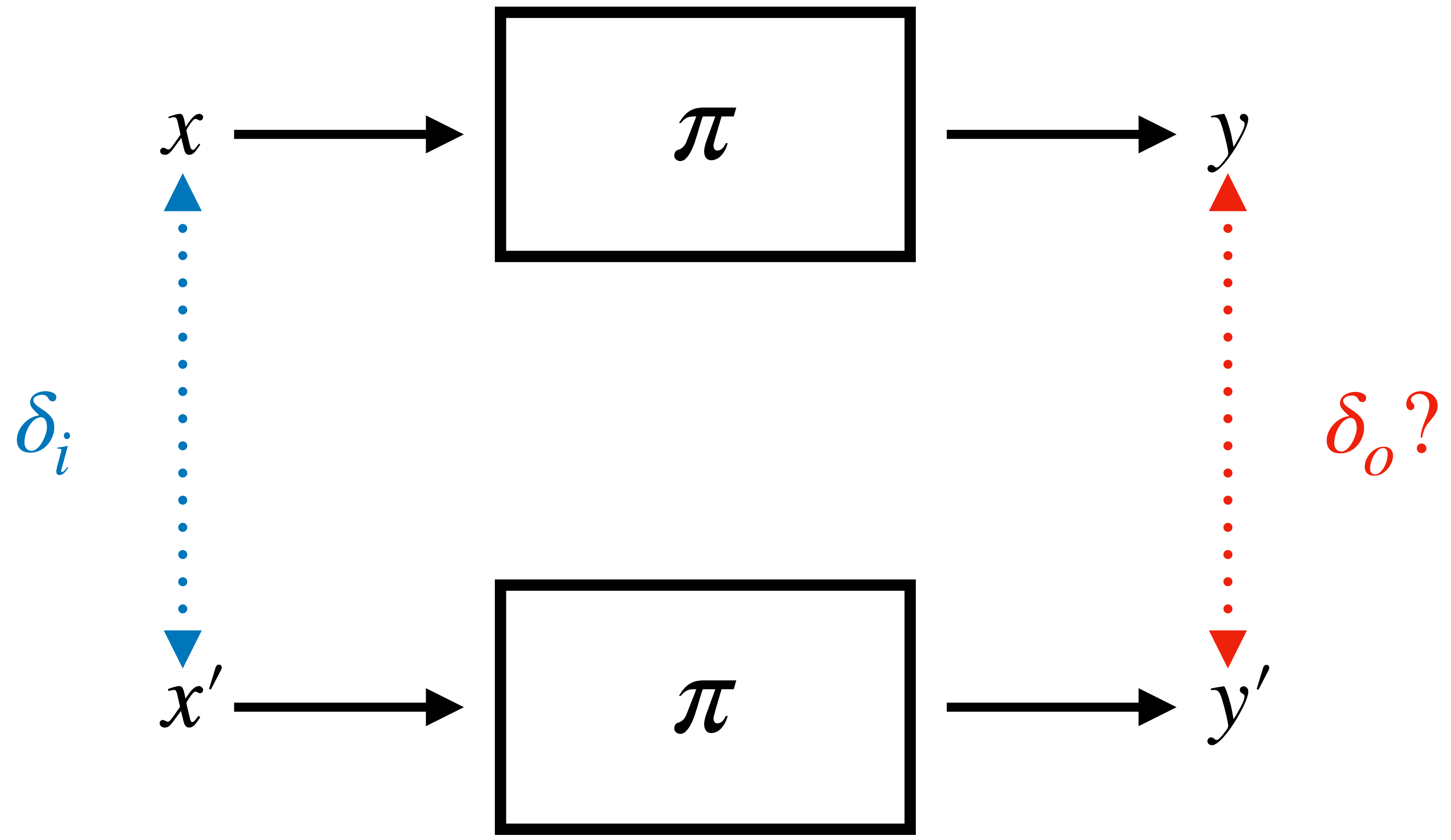
2. **Cryptanalysis Results on Spook**
Patrick Derbez, Paul Huynh, Virginie Lallemand, María Naya-Plasencia, Léo Perrin and André Schrottenloher **[CRYPTO 2020]**

3. **Skinny with Scalpel: Comparing Tools for Differential Cryptanalysis**
Stéphanie Delaune, Patrick Derbez, Paul Huynh, Marine Minier, Victor Mollimard and Charles Prud'homme **[ePrint 2020/1402]**

4. **On the Feistel Counterpart of the Boomerang Connectivity Table**
Hamid Boukerrou, Paul Huynh, Virginie Lallemand, Bimal Mandal and Marine Minier **[ToSC 2020]**

5. **Non-Triangular Self-Synchronizing Stream Ciphers**
Julien Francq, Loïc Besson, Paul Huynh, Philippe Guillot, Gilles Millerioux and Marine Minier **[TC - minor revision]**

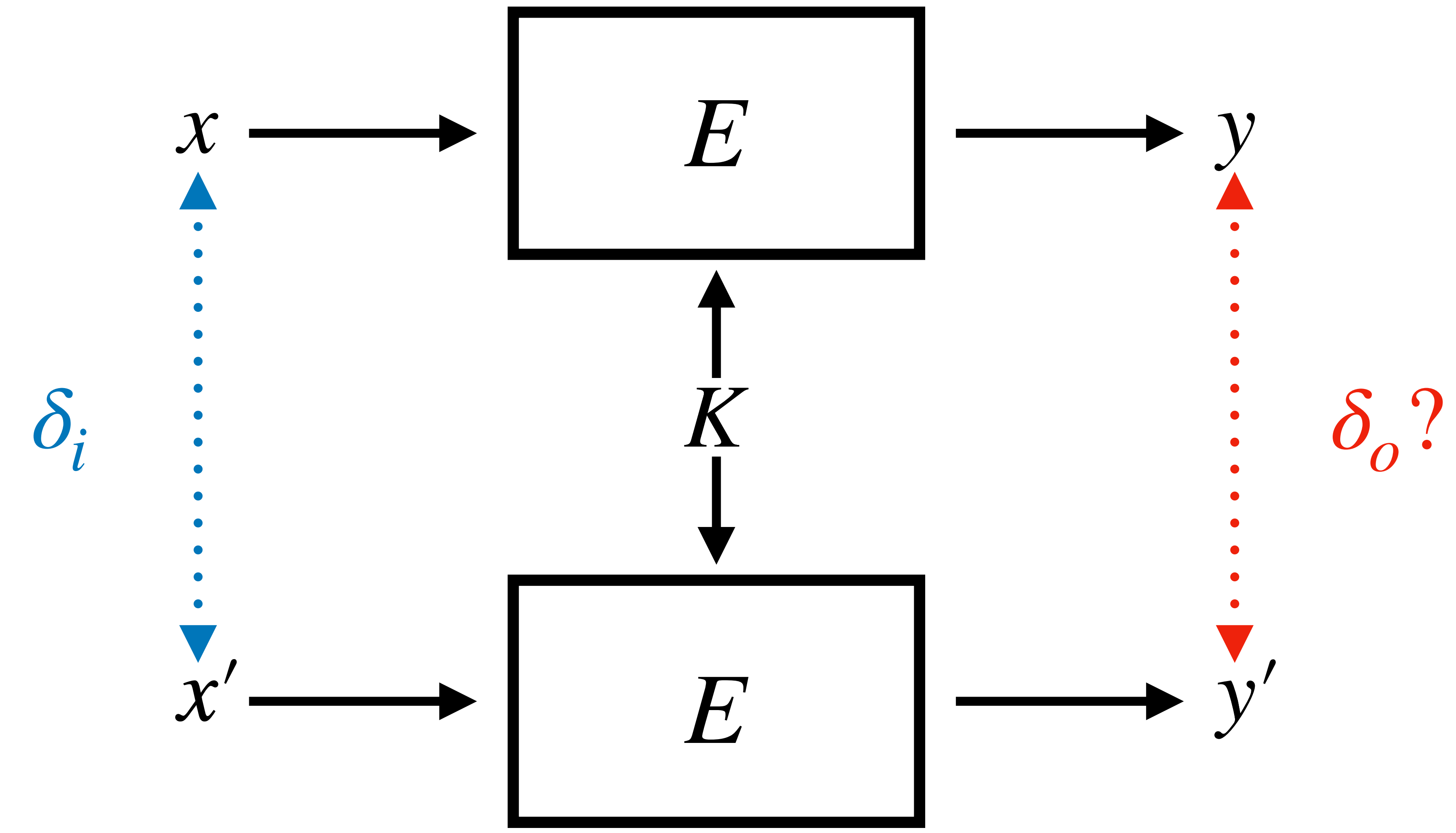
Differential cryptanalysis [Biham-Shamir 90]



For a random permutation π of $\mathbb{F}_2^n$, for any nonzero δ_i and δ_o

$$\Pr[\pi(x \oplus \delta_i) \oplus \pi(x) = \delta_o] = \frac{1}{2^n - 1}$$

Differential cryptanalysis [Biham-Shamir 90]



Exploit a **biais** in the distribution of output differences to build **differential distinguishers**.

- $(\delta_i \longrightarrow_E \delta_o)$ is a **differential**.
- E is **weak** if there exists a differential $(\delta_i \longrightarrow_E \delta_o)$ of **high probability** p.
  → round-key bits recovery in $\mathcal{O}(1/p)$
- $(\delta_i = \delta_0 \to \delta_1 \to \cdots \to \delta_r = \delta_o)$ is a **differential trail on r rounds**.

Part II
Cryptanalysis Results on Spook

Spook

Davide Bellizia, Francesco Berti, Olivier Bronchain, Gaëtan Cassiers, Sébastien Duval, Chun Guo, Gregor Leander, Gaëtan Leurent, Itamar Levi, Charles Momin, Olivier Pereira, Thomas Peters, François-Xavier Standaert, Balazs Udvarhelyi and Friedrich Wiemer

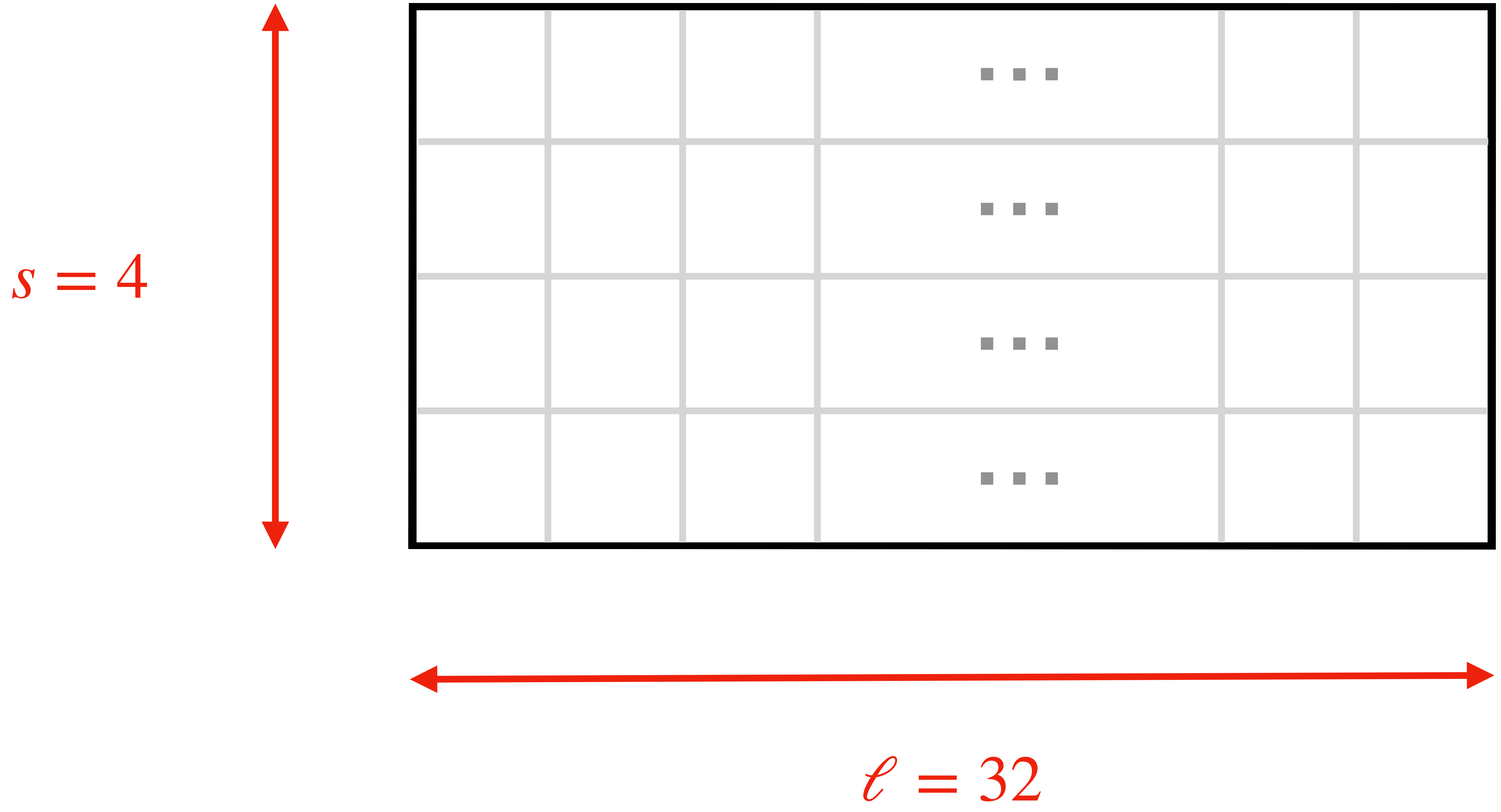
- 2nd round candidate to the **NIST LWC** standardization process
- Designed to achieve resistance against side-channel analysis and low-energy implementations
- **Authenticated Encryption** (AEAD) scheme
  - the Sponge One-Pass (S1P) mode of operation
  - the Clyde-128 tweakable block cipher
  - the Shadow permutation (512- or 384-bit state)

Summary of the results

- **Practical distinguishers**:
  - Shadow-512: 6 steps out of 6
  - Shadow-384: 5 steps out of 6
- **Practical forgeries** with 4-step Shadow for the S1P mode of operation (nonce misuse scenario)

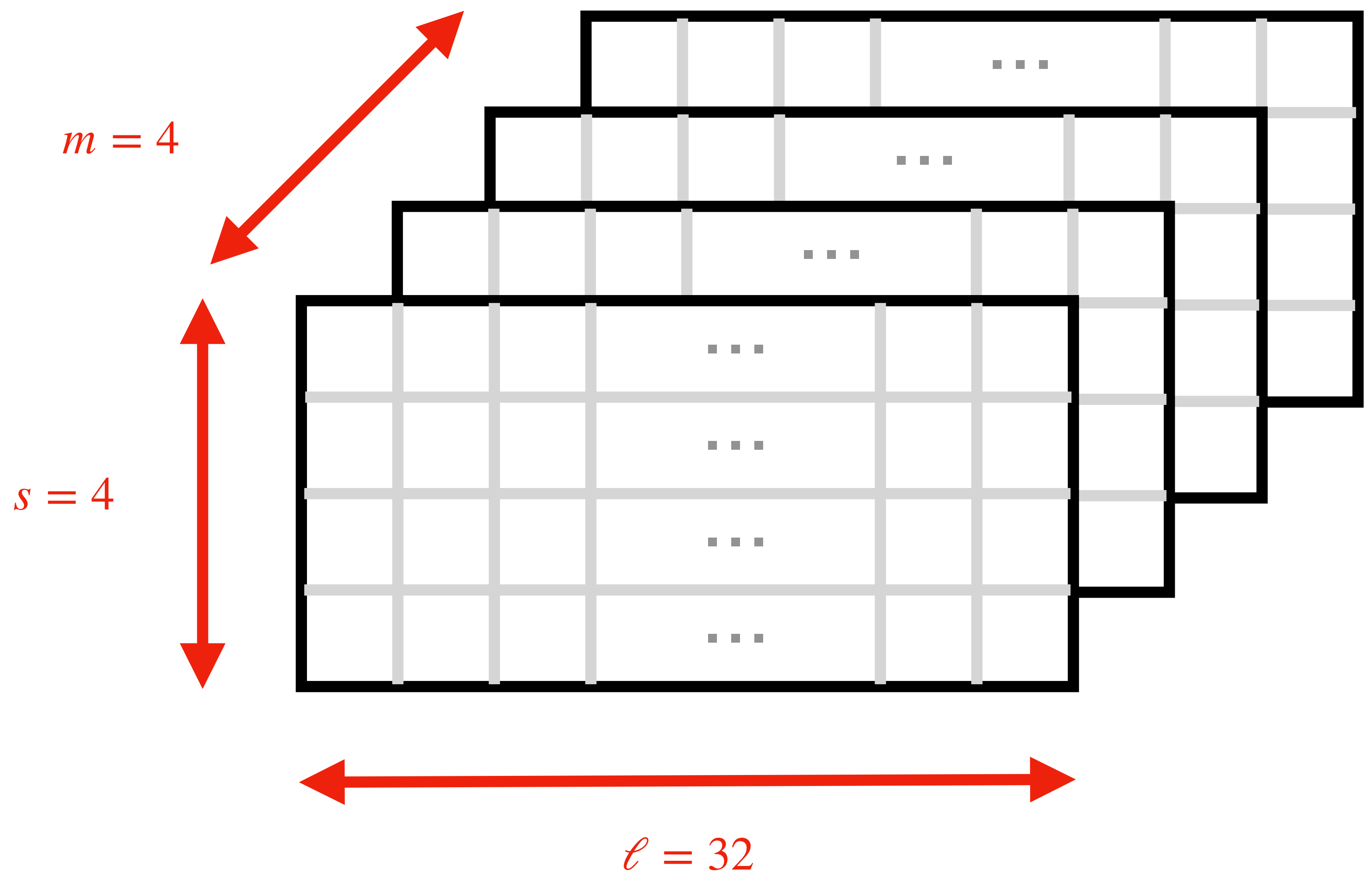
Description of Shadow

A Shadow bundle

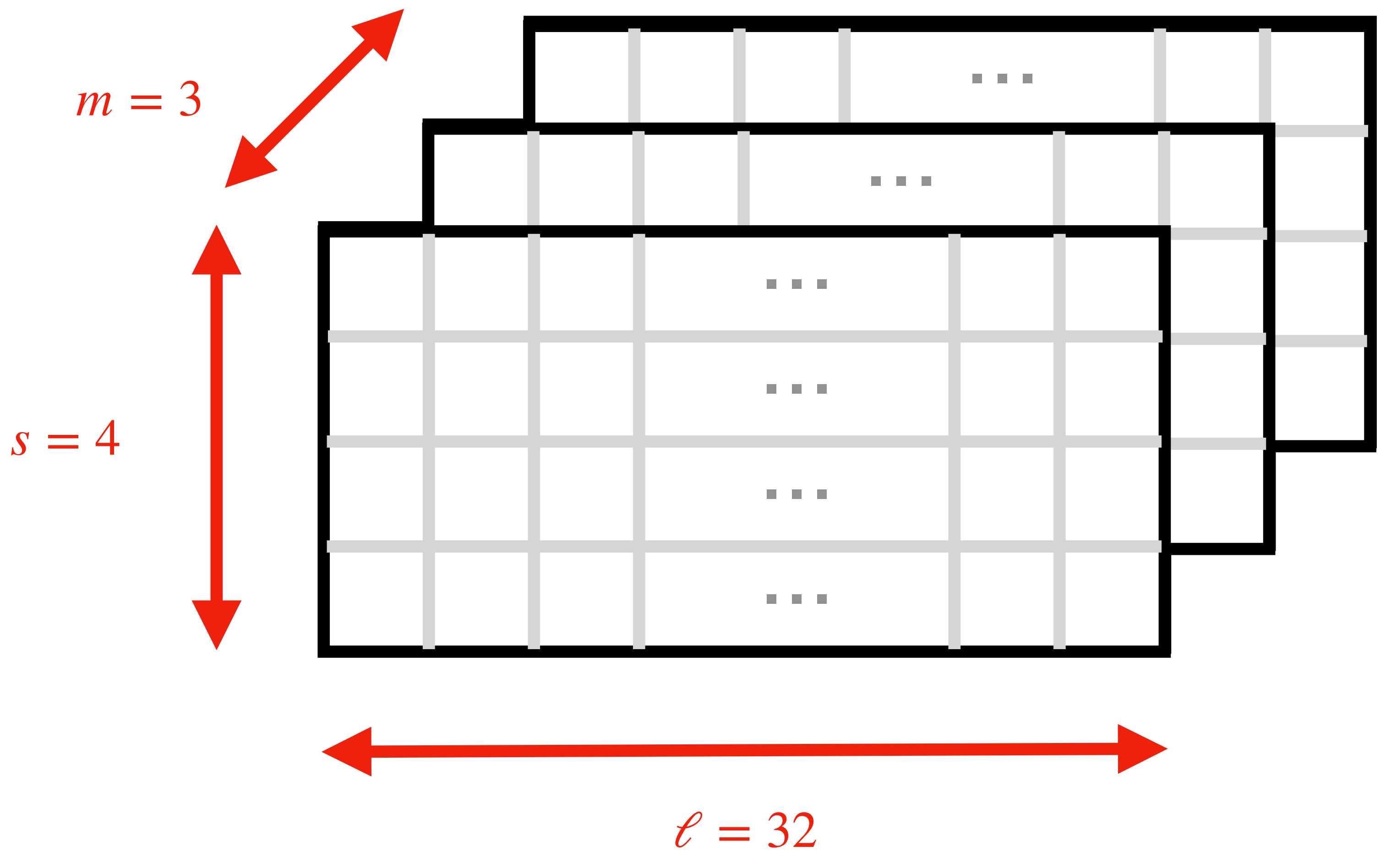


128 bits

A Shadow state

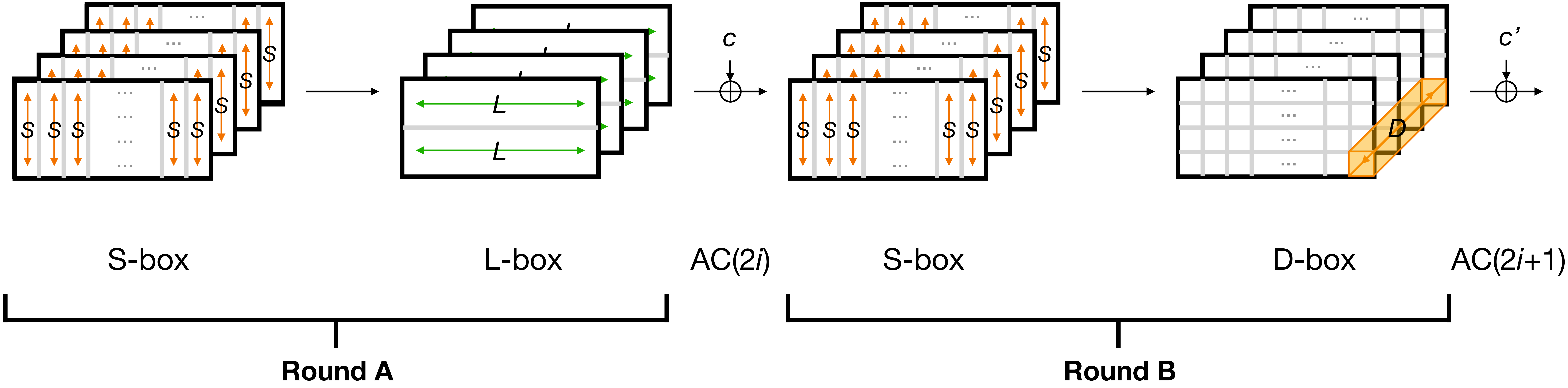


Shadow-512



Shadow-384

A Shadow encryption step

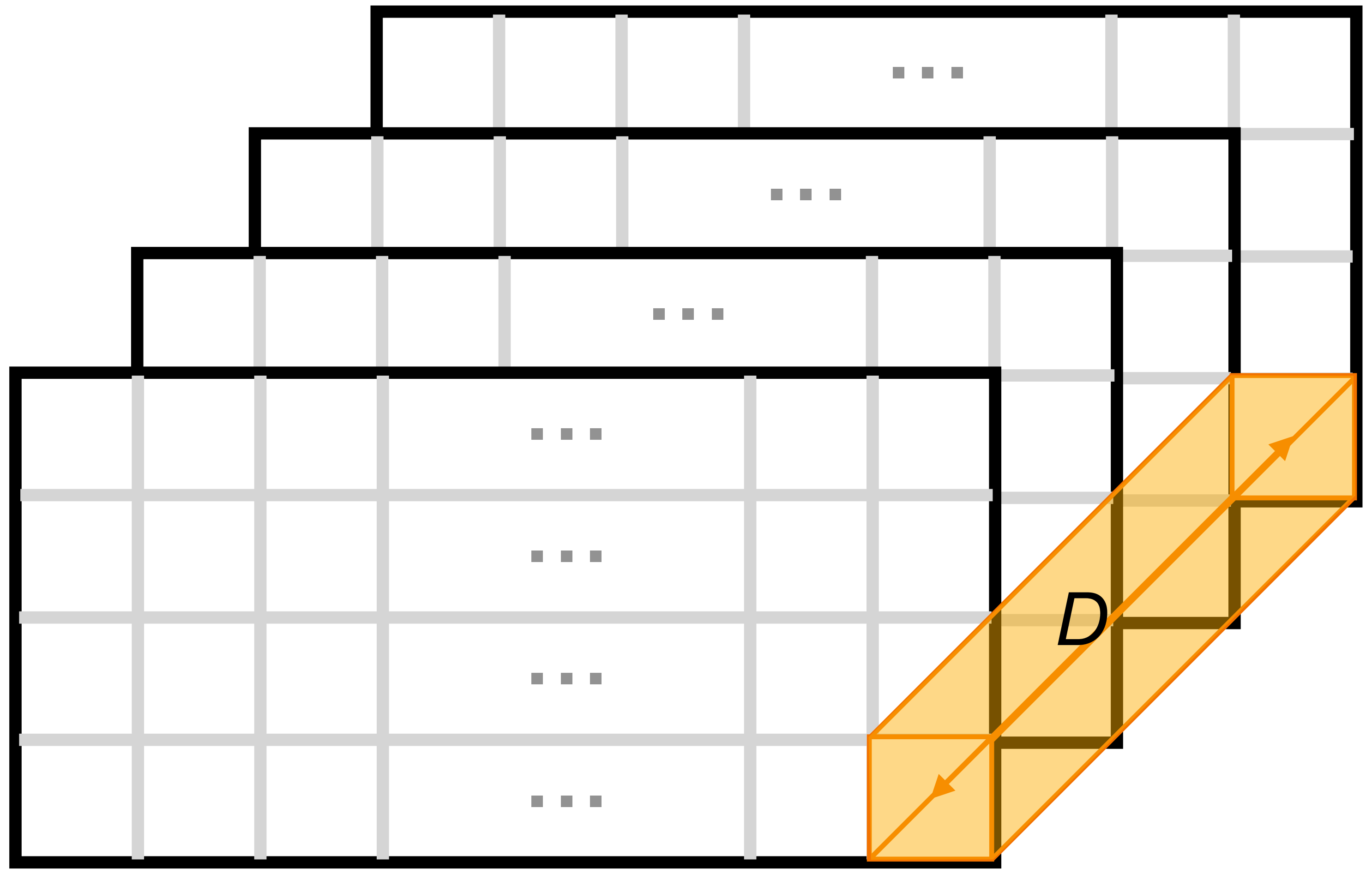


4-bit LFSR-generated constants added to **column *i* of bundle *i***

6 steps to complete encryption

The D-layer

D is the only diffusion layer between the *m* bundles



- Shadow-512:

$$D(a,b,c,d) = \begin{pmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix} \times \begin{pmatrix} a \\ b \\ c \\ d \end{pmatrix}$$

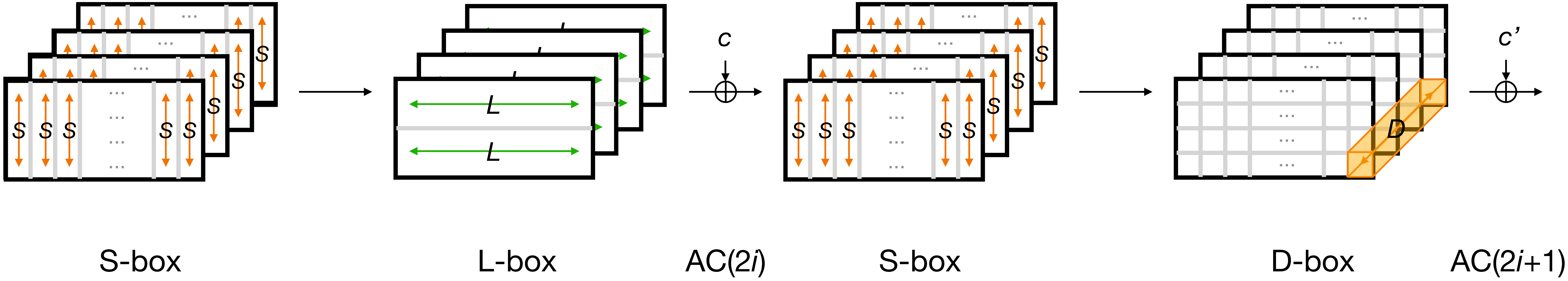
- Shadow-384:

$$D(a,b,c) = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{pmatrix} \times \begin{pmatrix} a \\ b \\ c \end{pmatrix}$$

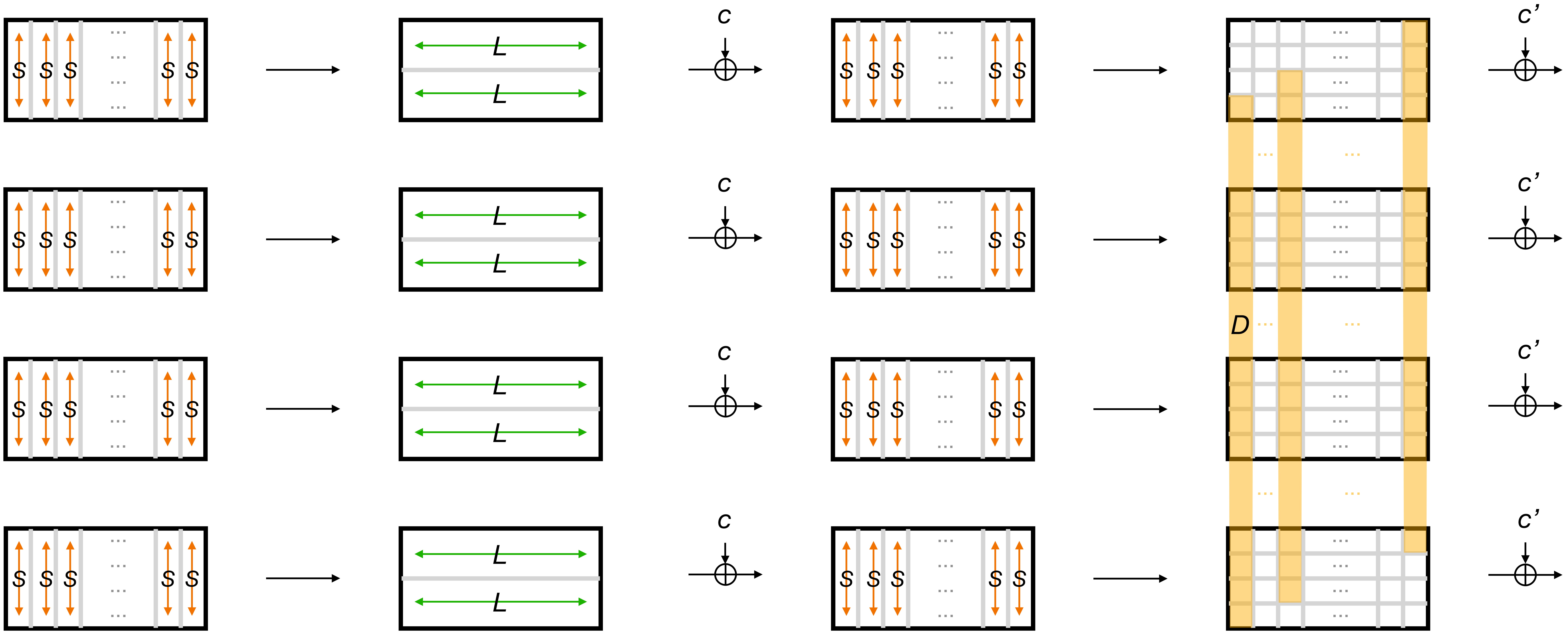
Core idea

Exploit the **similarity between the functions applied in parallel** on each bundle.

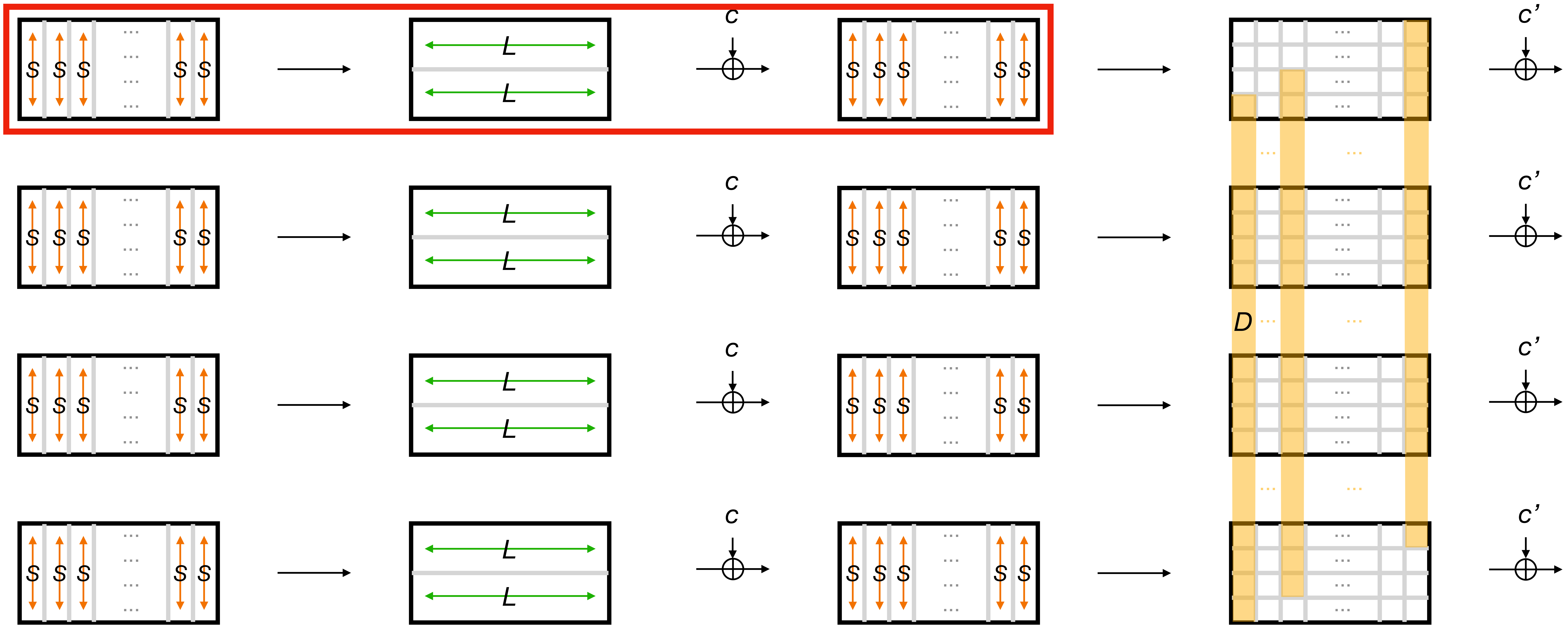
A Shadow step



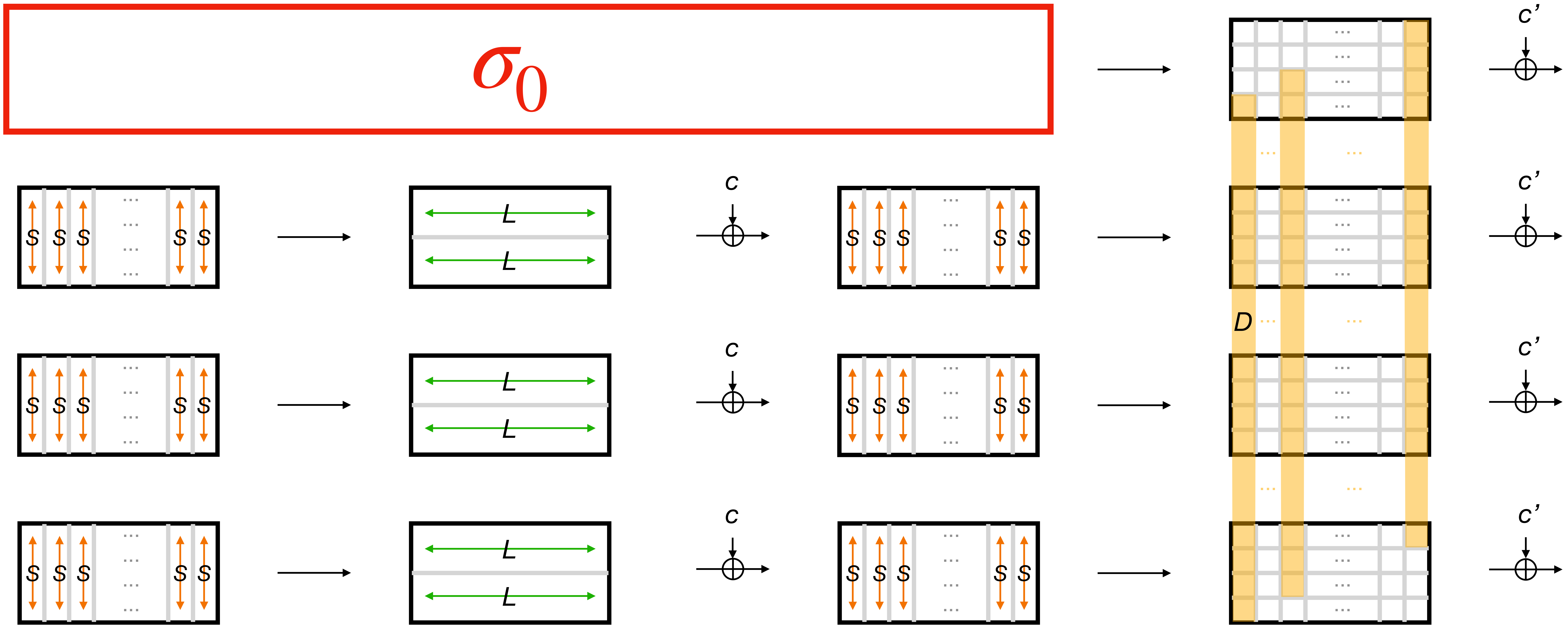
A Shadow step rewritten



A Shadow step rewritten



A Shadow step rewritten



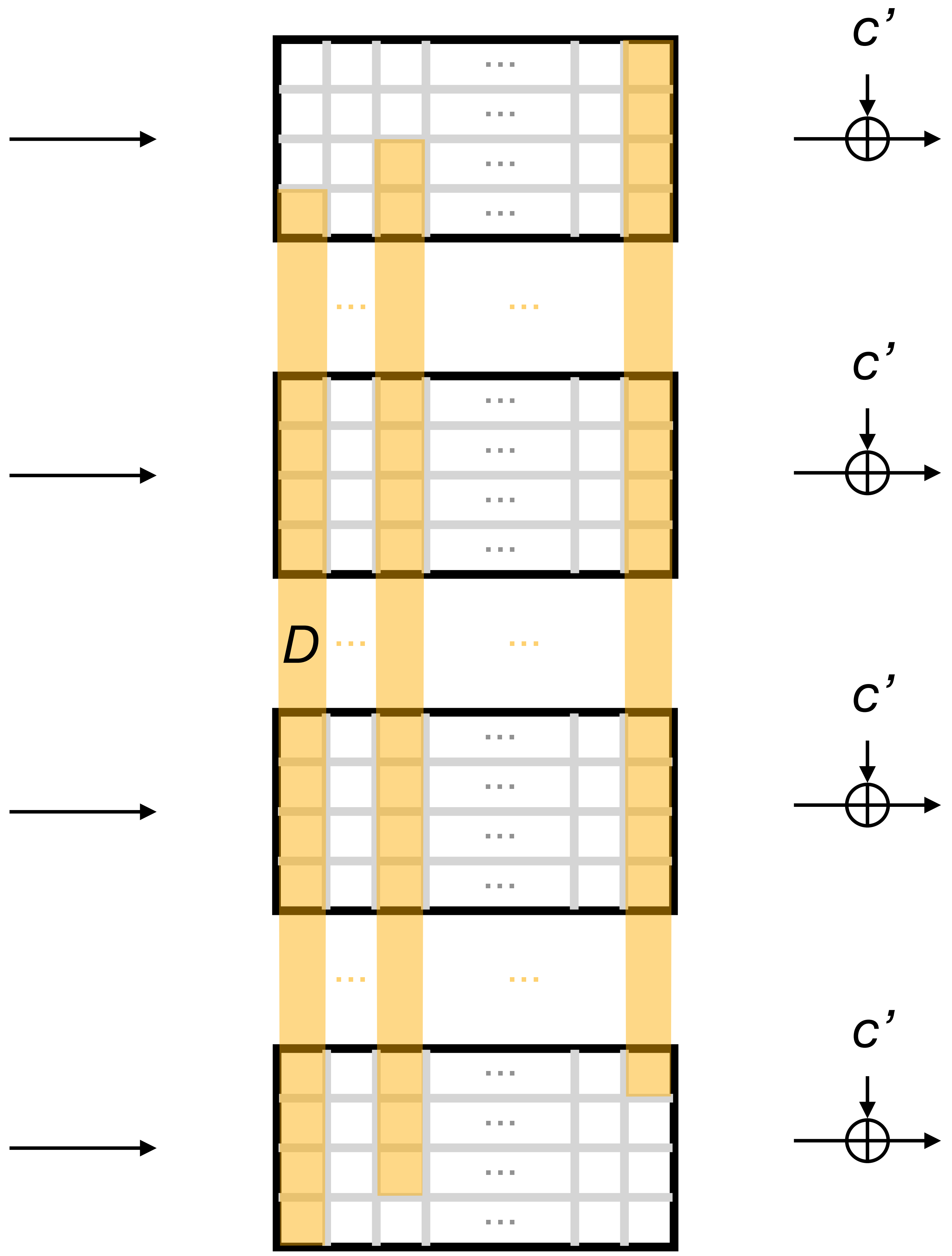
A Shadow step **rewritten**

σ_0

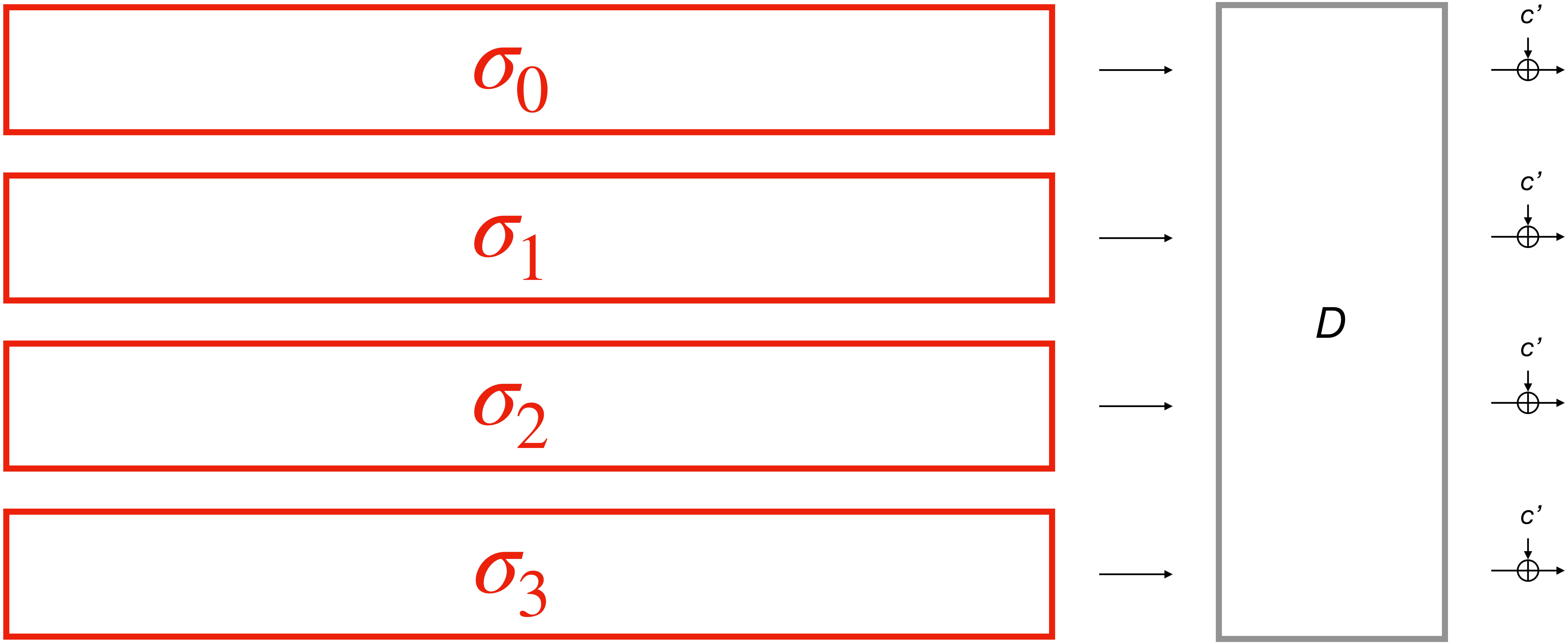
σ_1

σ_2

σ_3

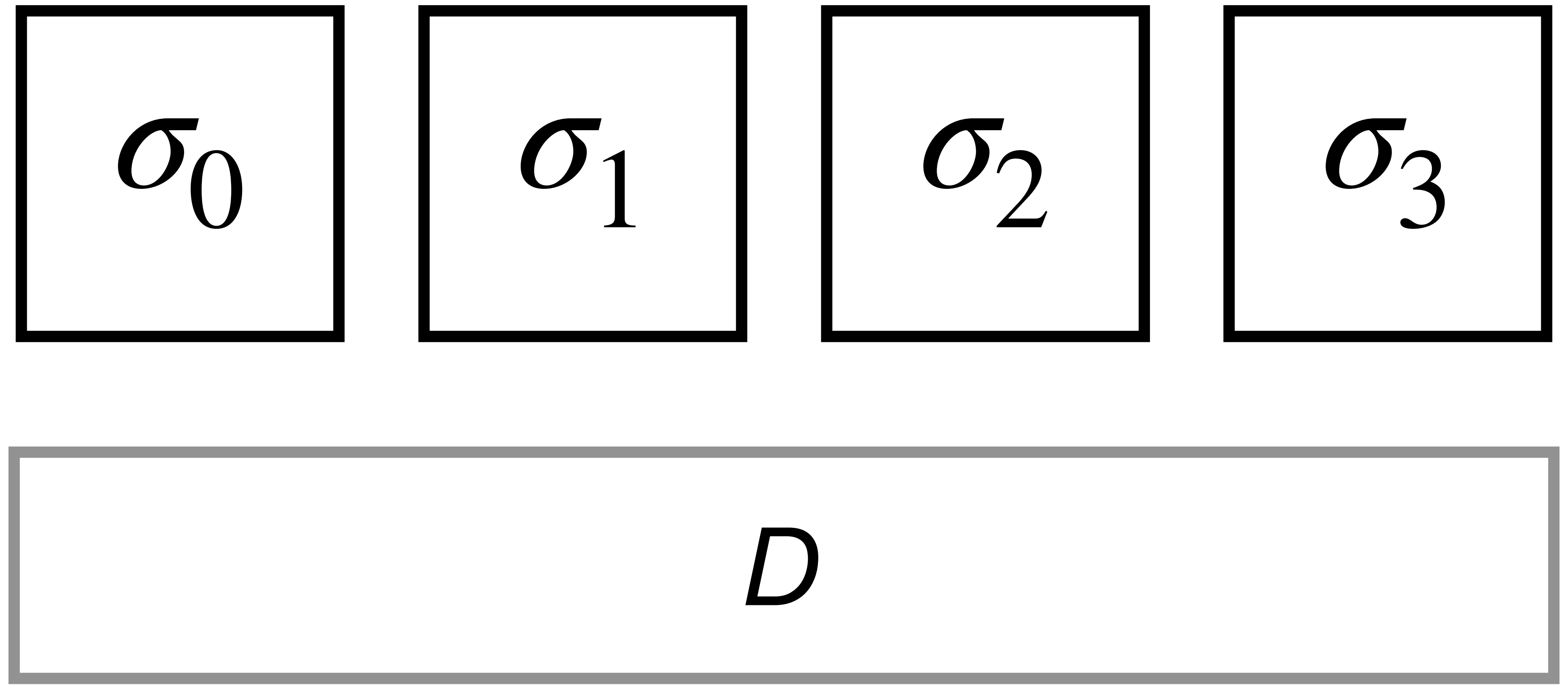


A Shadow step **rewritten**



A Shadow step rewritten

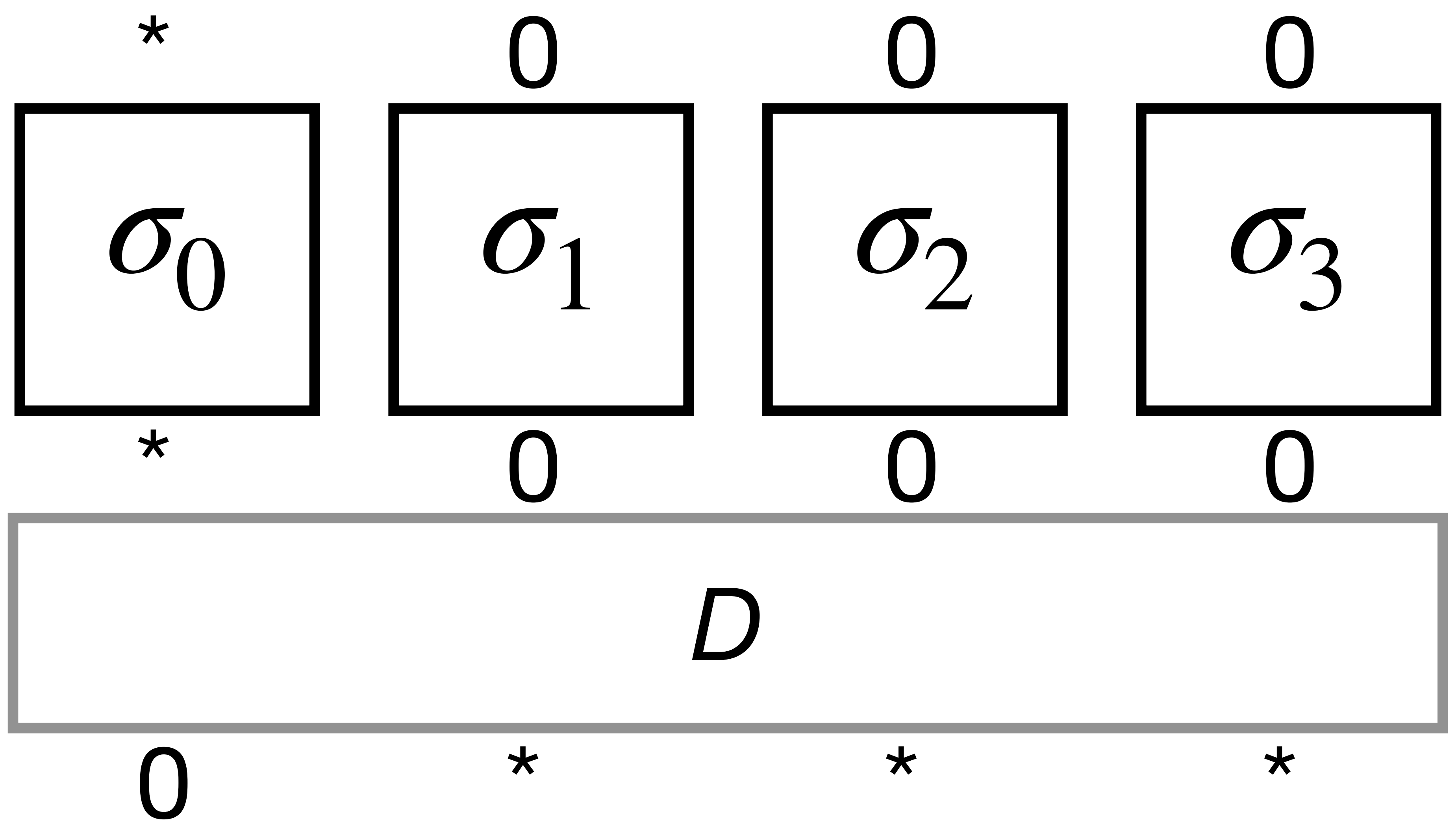
Seen as an SPN, using four 128-bit **Super S-boxes σ_i** interleaved with a linear permutation D operating on the full state.



A Shadow step rewritten

Seen as an SPN, using four 128-bit **Super S-boxes** σ_i interleaved with a linear permutation D operating on the full state.

Truncated differential distinguisher:

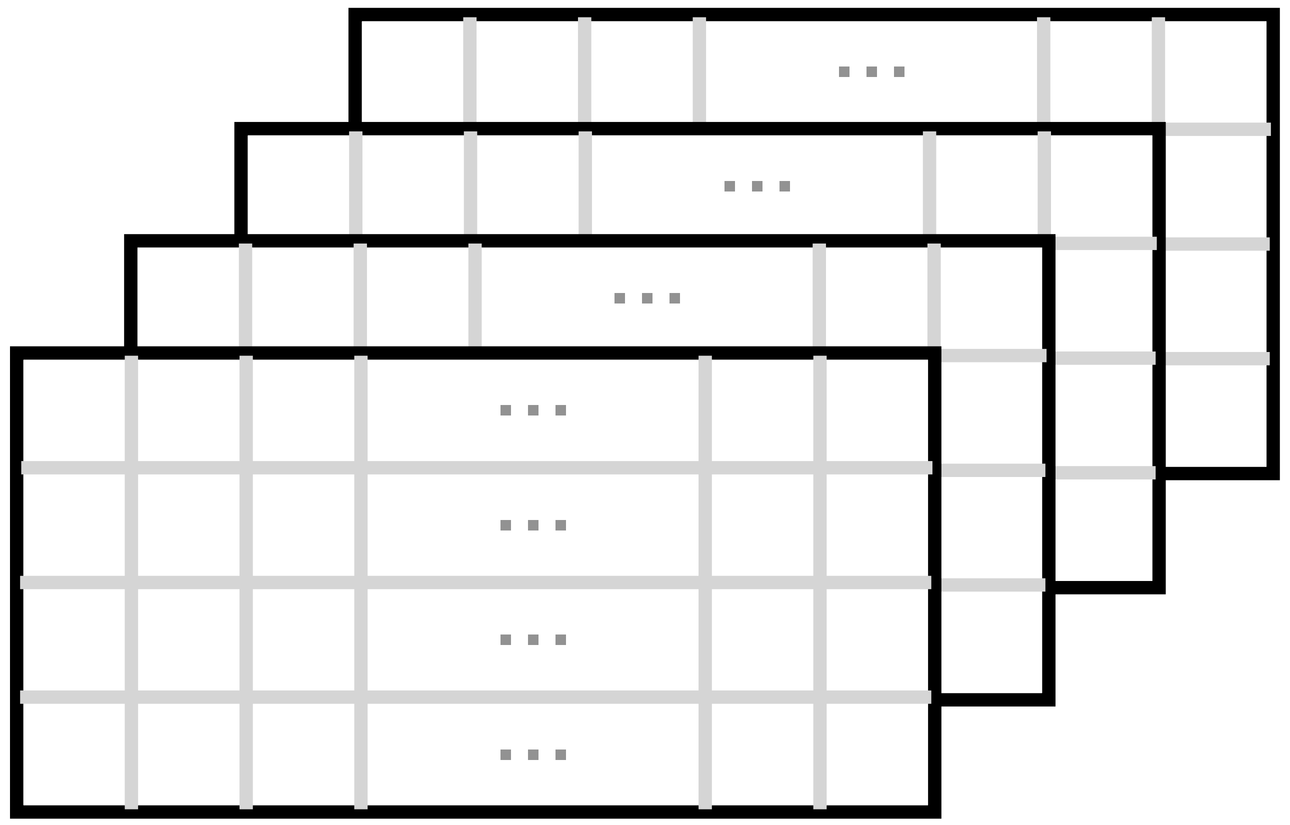


‘**0**’: no difference

‘*****’: undetermined difference

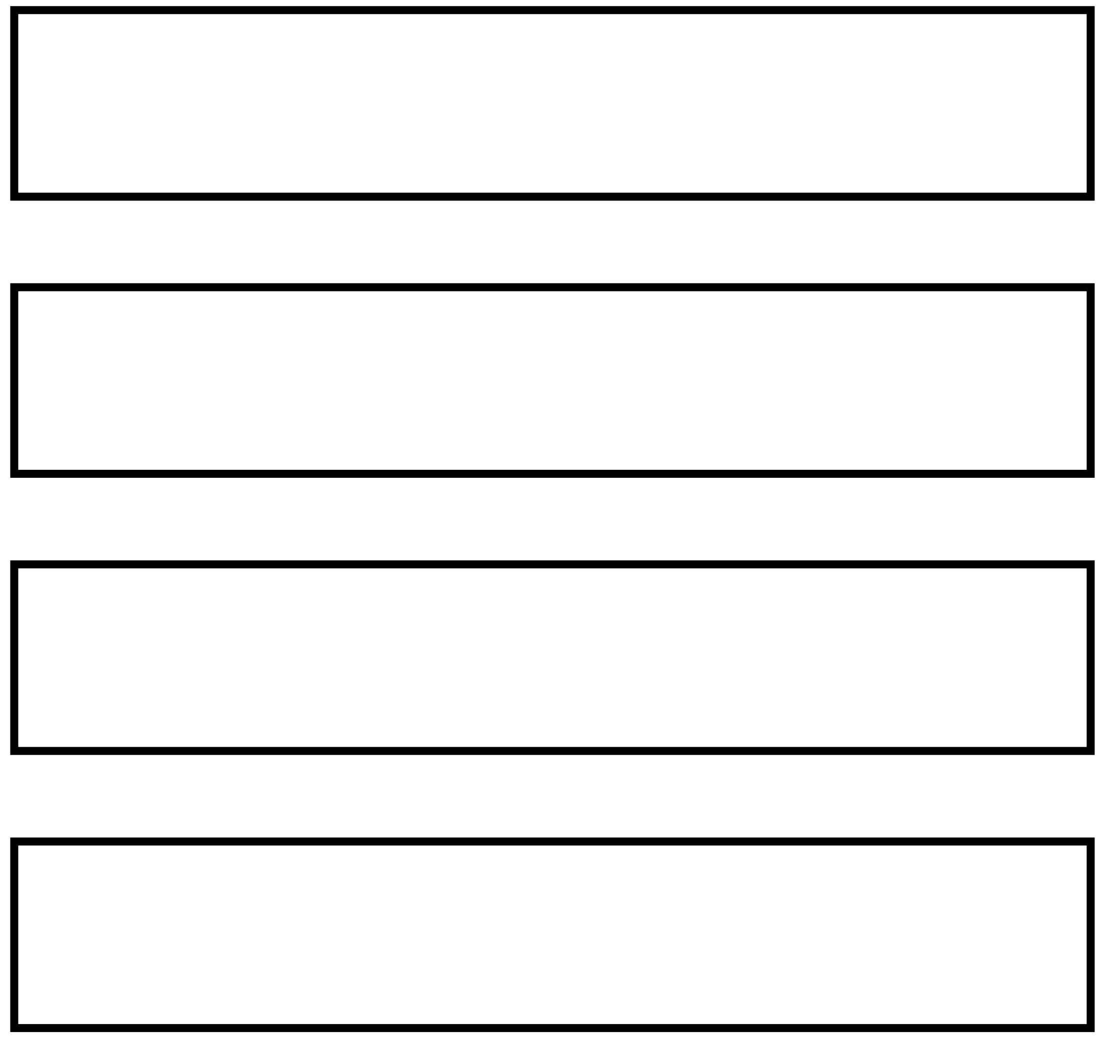
Structural observations

We call ***i*-identical** an internal state of Shadow in which i bundles are equal.



Structural observations

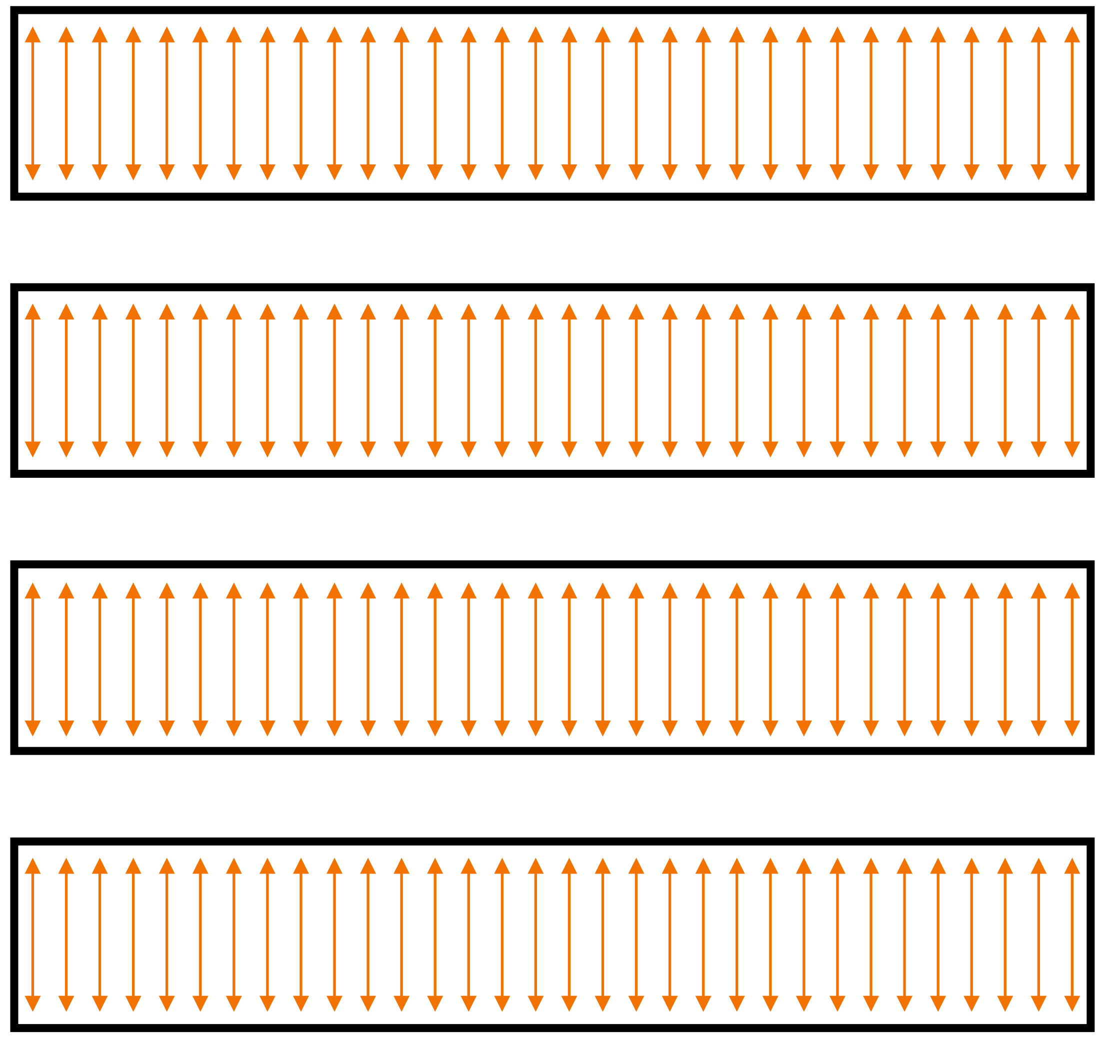
We call i-**identical** an internal state of Shadow in which i bundles are equal.



Initial state

Structural observations

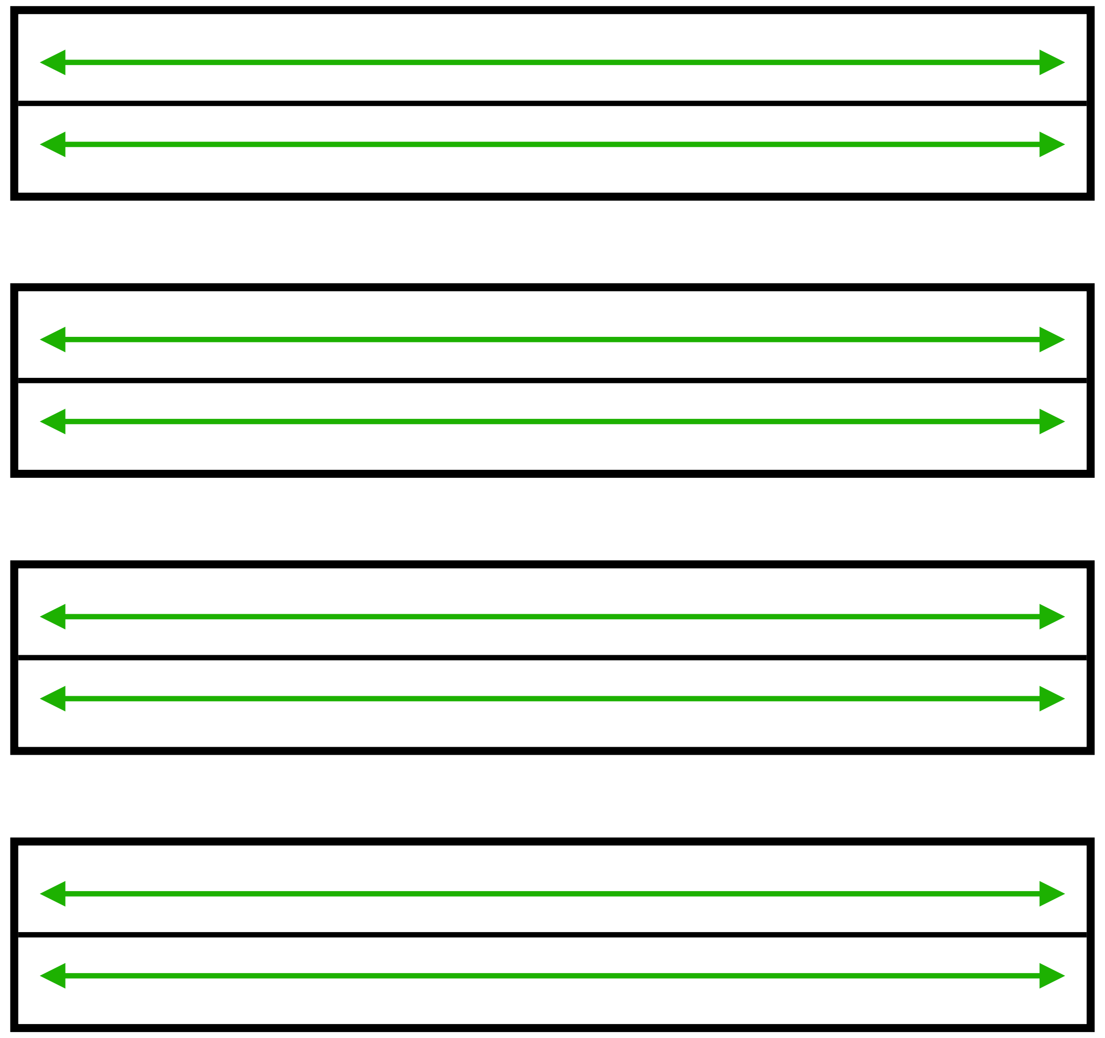
We call ***i*-identical** an internal state of Shadow in which i bundles are equal.



S-Box layer

Structural observations

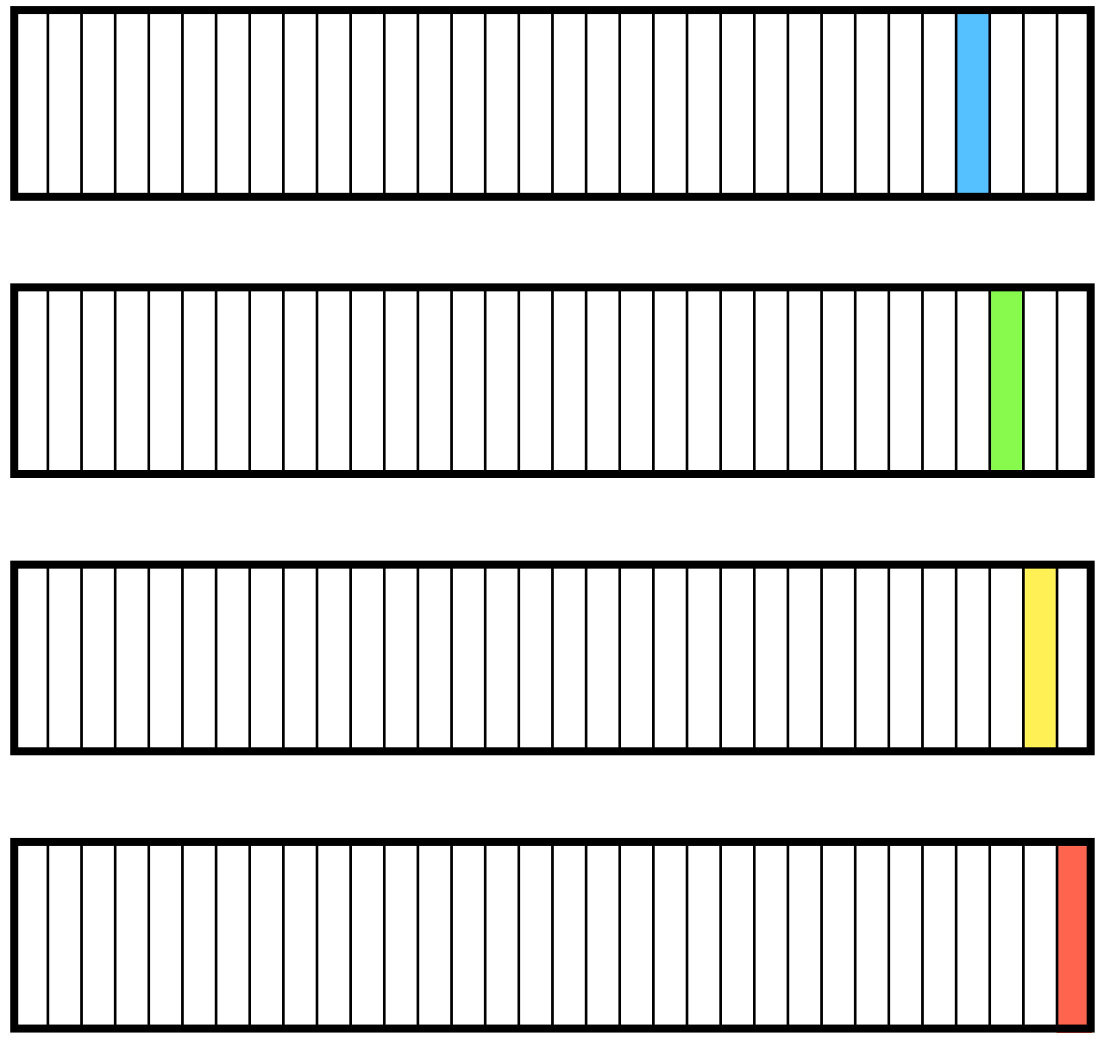
We call i**-identical** an internal state of Shadow in which i bundles are equal.



L-Box layer

Structural observations

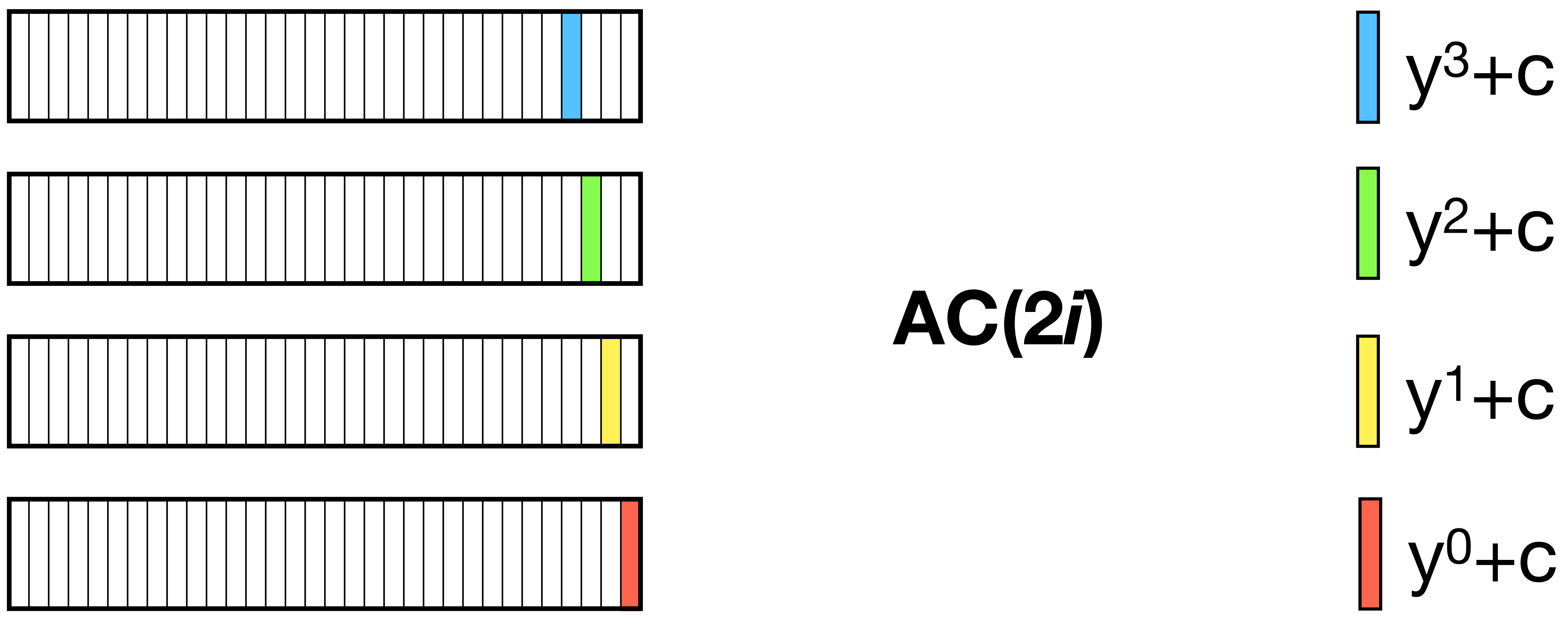
We call i**-identical** an internal state of Shadow in which i bundles are equal.



AC(2*i*)

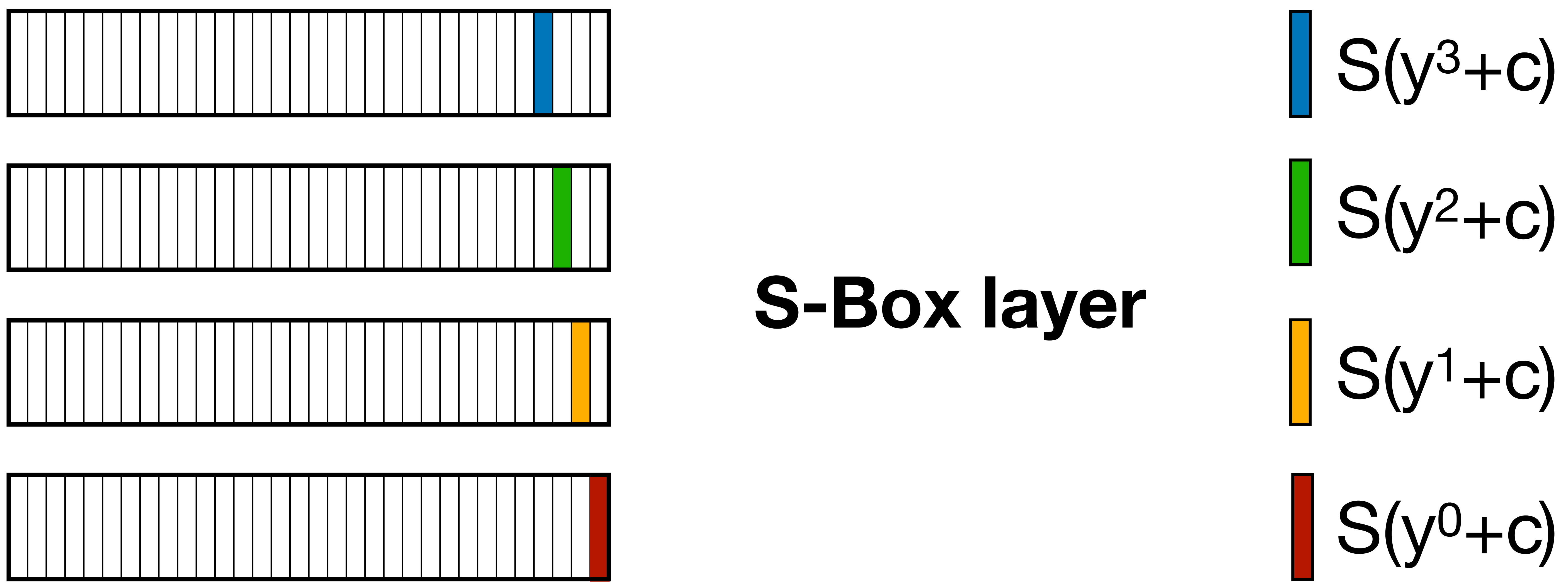
Structural observations

We call i-**identical** an internal state of Shadow in which i bundles are equal.



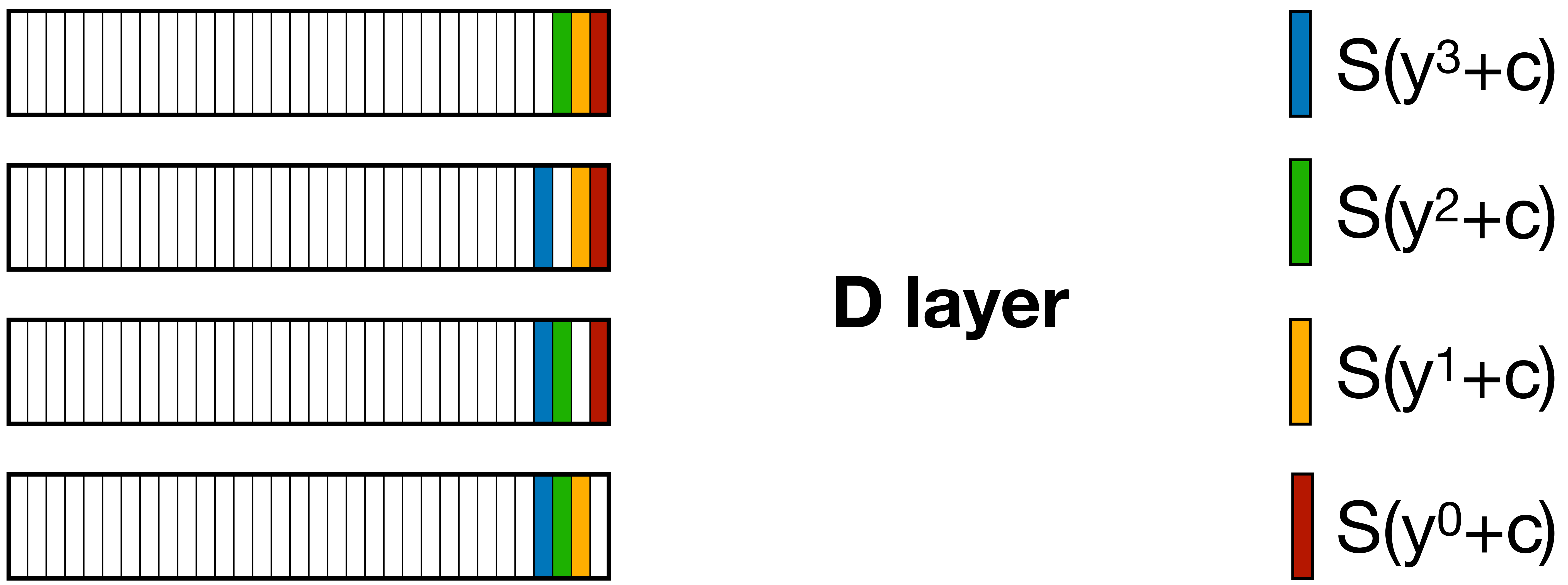
Structural observations

We call ***i*-identical** an internal state of Shadow in which i bundles are equal.



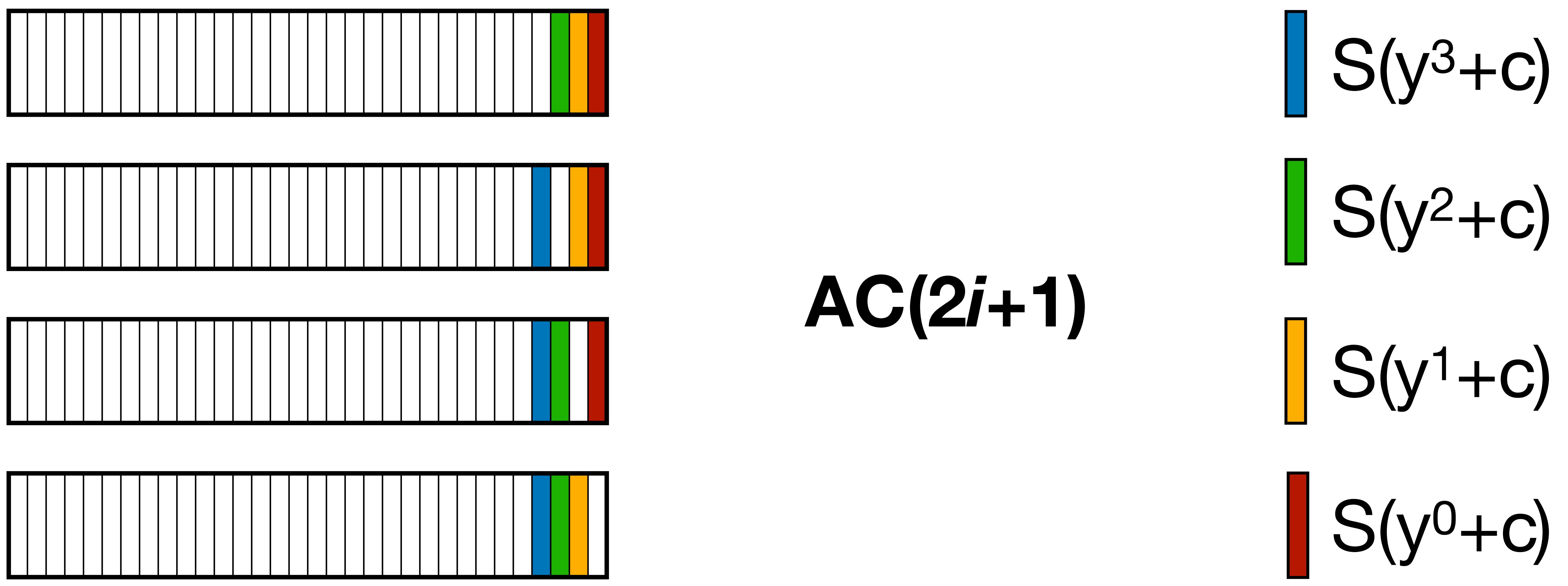
Structural observations

We call ***i*-identical** an internal state of Shadow in which i bundles are equal.



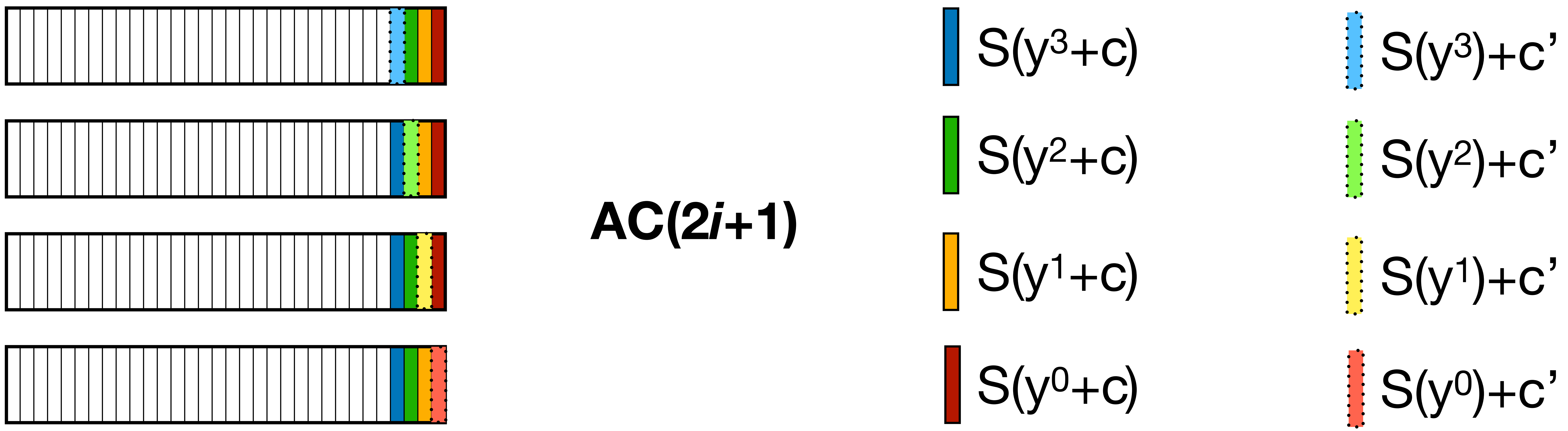
Structural observations

We call **i-identical** an internal state of Shadow in which i bundles are equal.



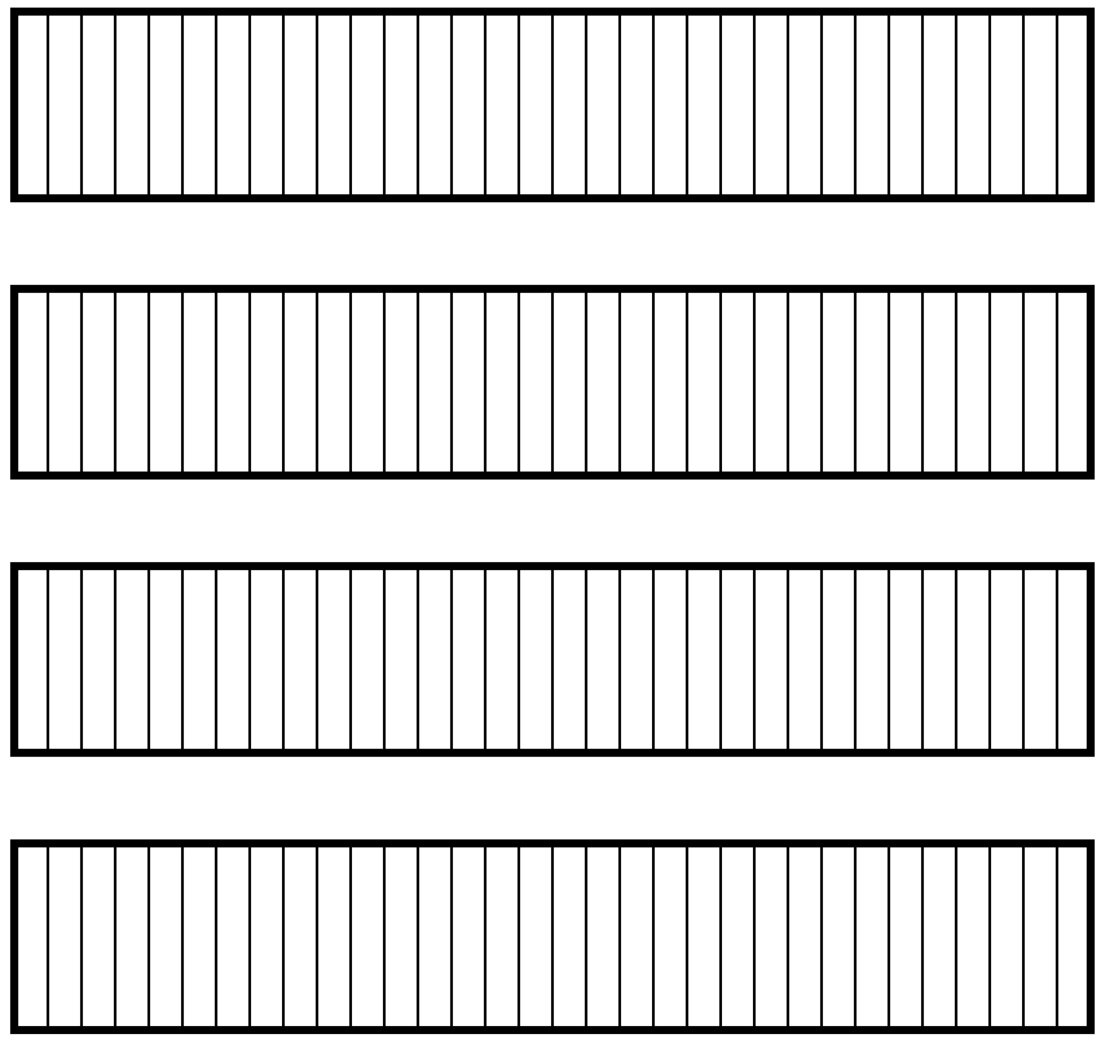
Structural observations

We call i**-identical** an internal state of Shadow in which i bundles are equal.



Structural observations

We call ***i*-identical** an internal state of Shadow in which i bundles are equal.



$$S(y^3+c) = S(y^3)+c'$$

$$S(y^2+c) = S(y^2)+c'$$

$$S(y^1+c) = S(y^1)+c'$$

$$S(y^0+c) = S(y^0)+c'$$

Structural observations

We call ***i*-identical** an internal state of Shadow in which i bundles are equal.

probabilities of preserving an i-identical state at step s

s	0	1	2	3	4
i=4	0	0	2^{-12}	2^{-8}	0

$$\boxed{\begin{array}{l} S(y^3+c) = S(y^3)+c' \\ S(y^2+c) = S(y^2)+c' \\ S(y^1+c) = S(y^1)+c' \\ S(y^0+c) = S(y^0)+c' \end{array}}$$

Structural observations

We call ***i*-identical** an internal state of Shadow in which i bundles are equal.

probabilities of preserving an i-identical state at step s

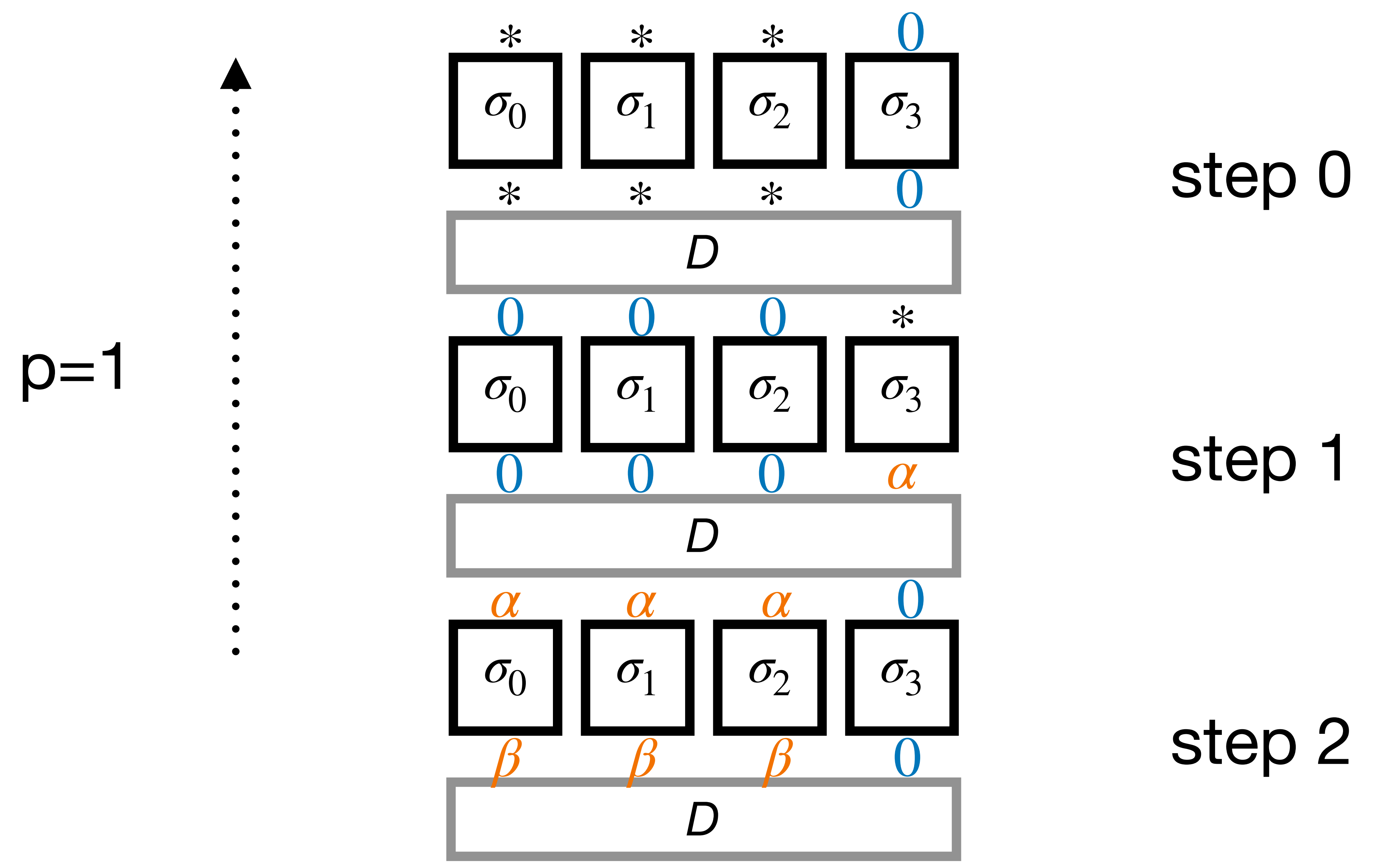
s	0	1	2	3	4
i=4	0	0	2^{-12}	2^{-8}	0
i=3	0	0	2^{-9}	2^{-6}	0
i=2	0	0	2^{-6}	2^{-4}	0

Distinguisher

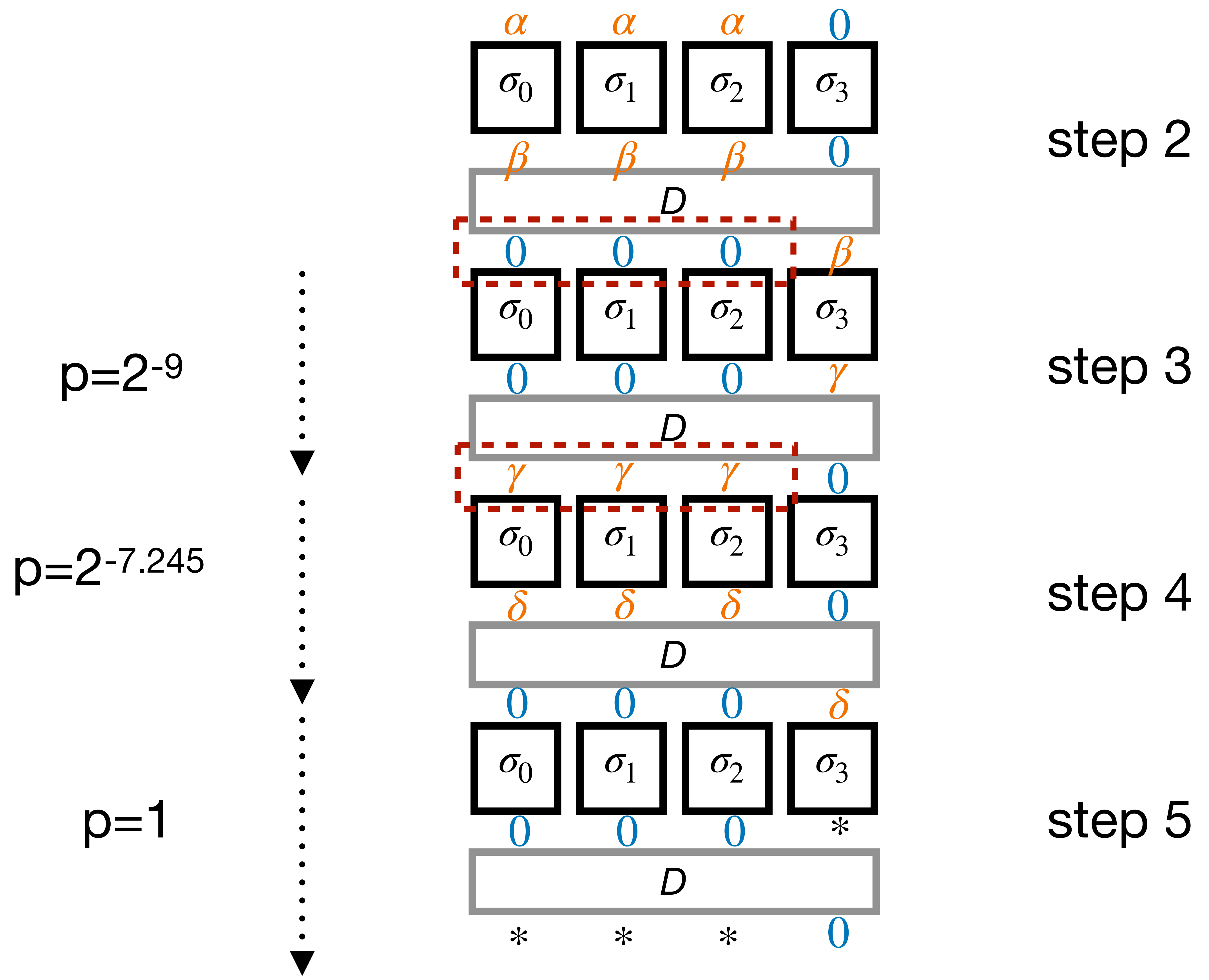
Distinguisher on 6 steps of Shadow-512

- $x \oplus x' = (*, *, *, 0)$ and shadow(x) $\oplus$ shadow(x') $= D(0, 0, 0, *)$
- Generic cost 2^{-64} vs $2^{-16.245}$ here

Distinguisher on 6 steps of Shadow-512



Distinguisher on 6 steps of Shadow-512



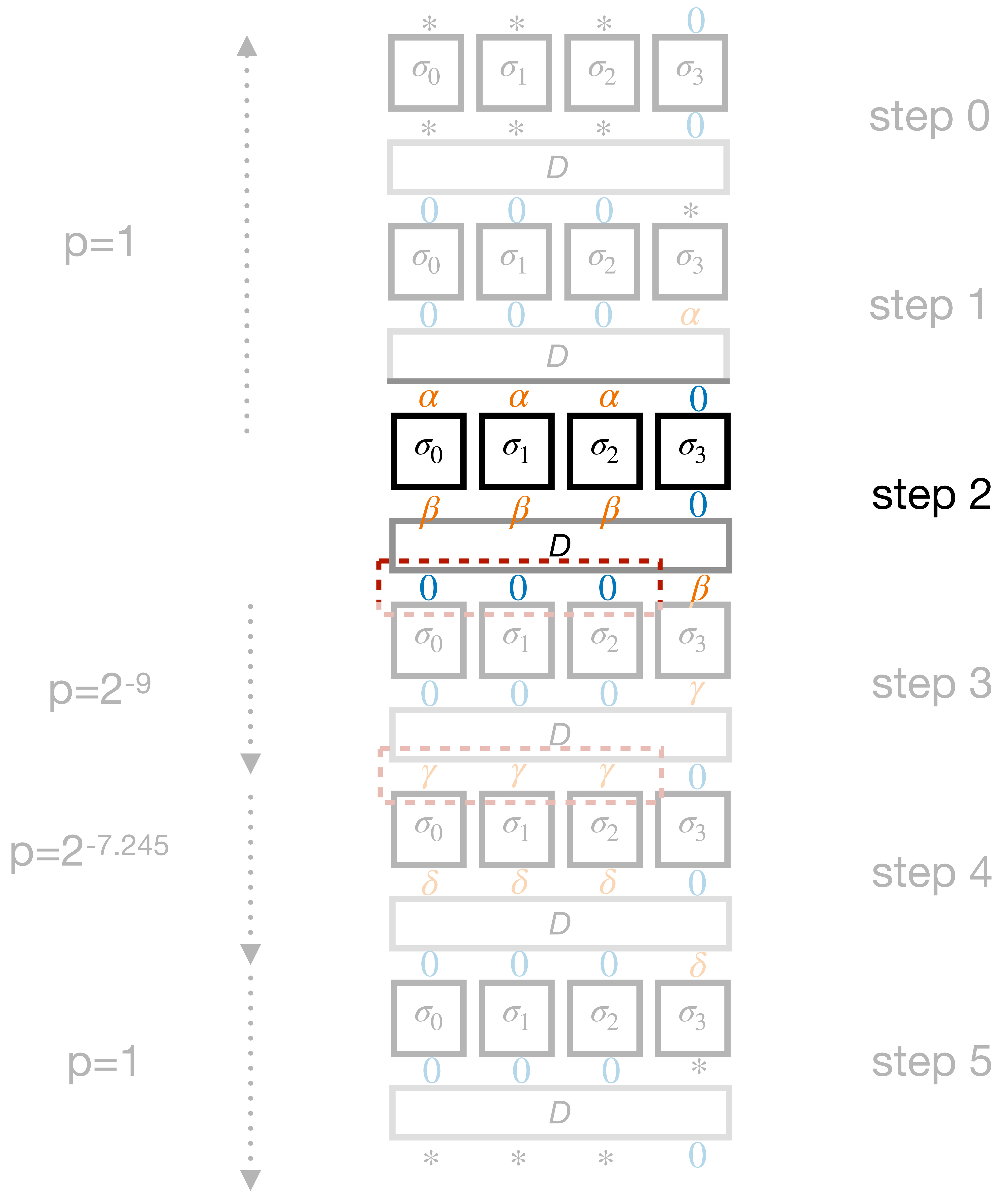
Some details

- Constructing a pair for **step 2**:
  - $\sigma_0(x) + \sigma_0(x+\alpha) = \beta$
    $\sigma_1(x+\epsilon) + \sigma_1(x+\epsilon+\alpha) = \beta$
    $\sigma_2(x+\epsilon') + \sigma_2(x+\epsilon'+\alpha) = \beta$

    and **3-identical state at the end of step 2**
  - Impact of the constant additions limited to the S-boxes with indices in {0,1,2,3}
  - Bits with indices **22** and **23** in each of the 4 input words of a Super S-box have **no influence** on the output bits with indices in {0,1,2,3}

    $$\nabla = \{a \times e_{22} + b \times e_{23}, a \in \mathbb{F}_2^4, b \in \mathbb{F}_2^4\}$$

    For all $\alpha \in \nabla$, all steps and all bundle index i,
    $\sigma_i(x) + \sigma_i(x+\alpha) = (*, *, \ldots, *, 0,0,0,0)$



Some details

- **Step 3**: probability of a **3-identical state** = **2^{-9}**
- **Step 4: difference of the form $(0,0,0,\delta)$ at the end of the step**

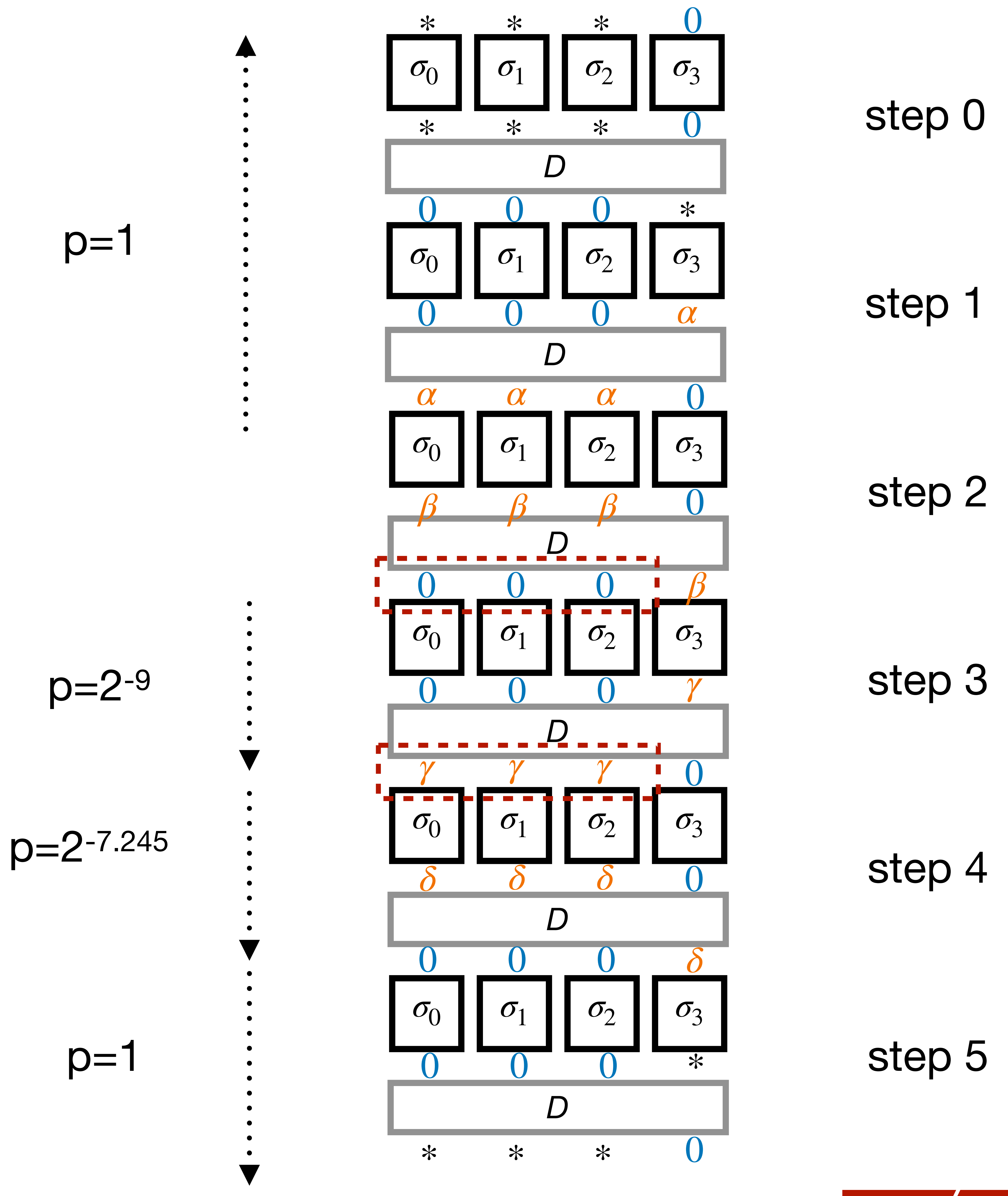
  Let (y, y, y, w) and (y', y', y', w) denote two messages after the application of S and L of step 4 then:

  $$S(y'^2) \oplus S(y'^2 \oplus c) = S(y^2) \oplus S(y^2 \oplus c)$$
  $$S(y'^1) \oplus S(y'^1 \oplus c) = S(y^1) \oplus S(y^1 \oplus c)$$
  $$S(y'^0) \oplus S(y'^0 \oplus c) = S(y^0) \oplus S(y^0 \oplus c)$$

  with $c = 0x5$, probability of **$2^{-2.415}$** for each equality

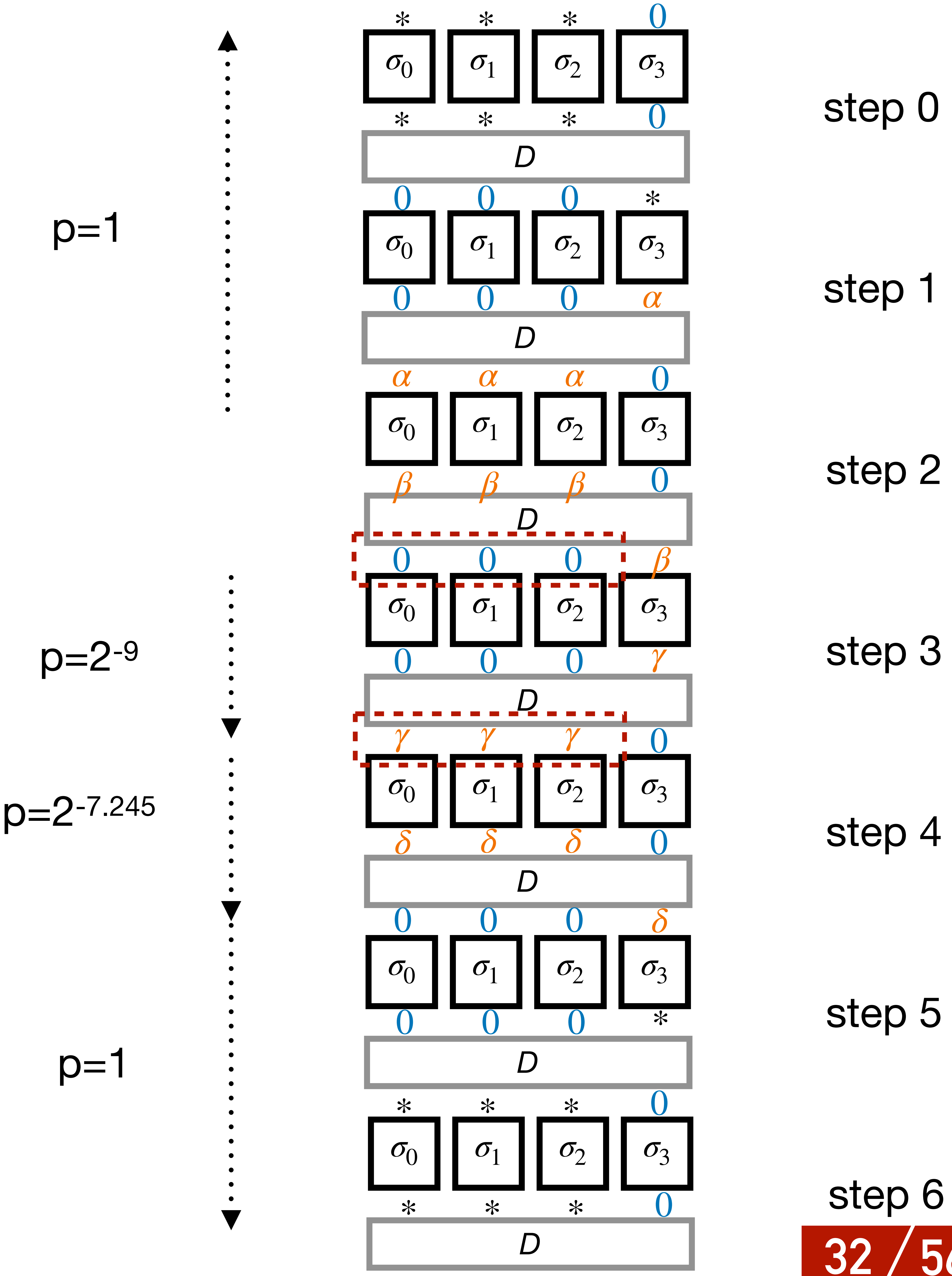
- **Step 5** has probability **1**

Total probability: $(2^{-2.415})^3$ x 2^{-9} = $2^{-16.245}$



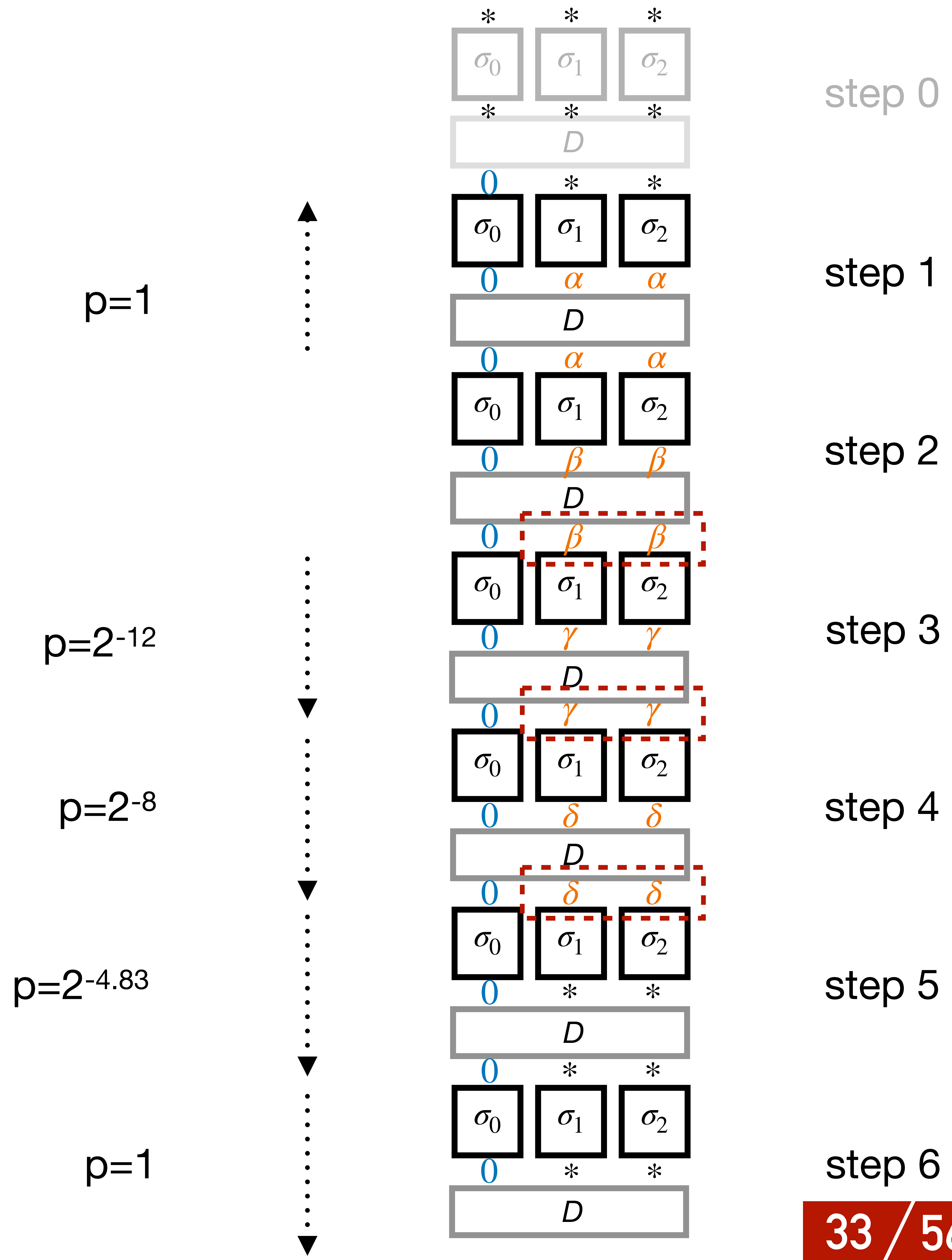
Extension to 7 steps

No extra cost.



The Shadow-384 case

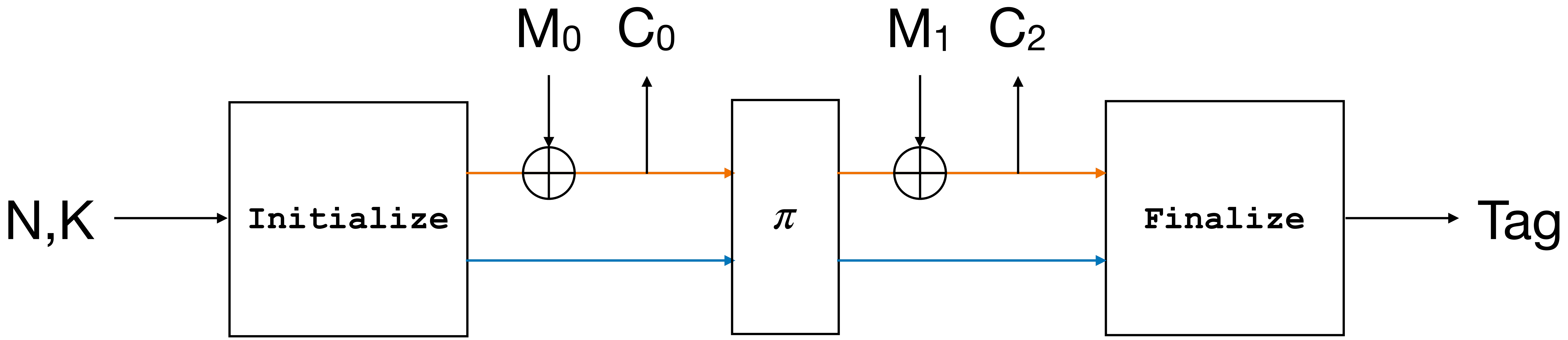
$$D(a,b,c) = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{pmatrix} \times \begin{pmatrix} a \\ b \\ c \end{pmatrix}$$



Forgery

Forgery

S1P mode in our attack setting

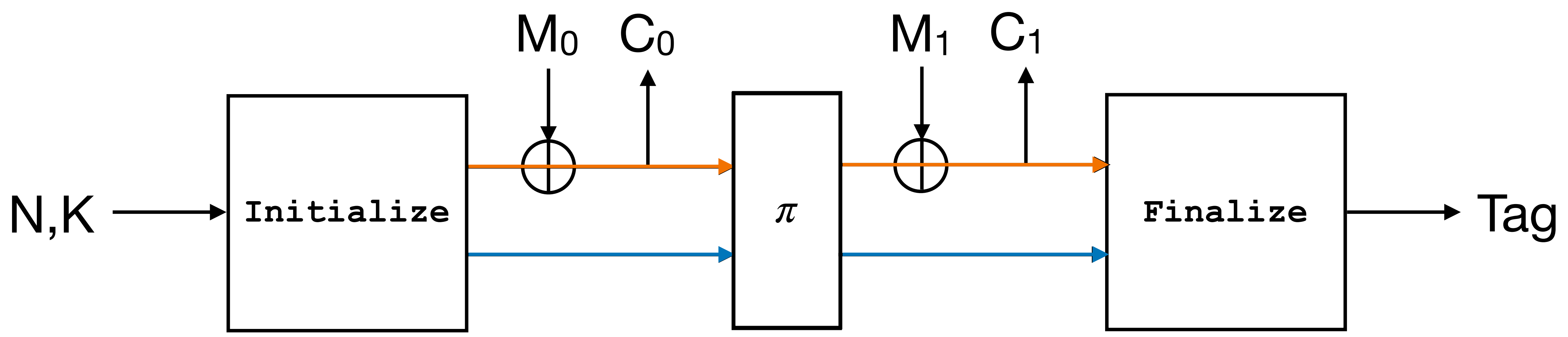


rate: bundle 0, 1

capacity: bundle 2, 3, **not visible**

Forgery

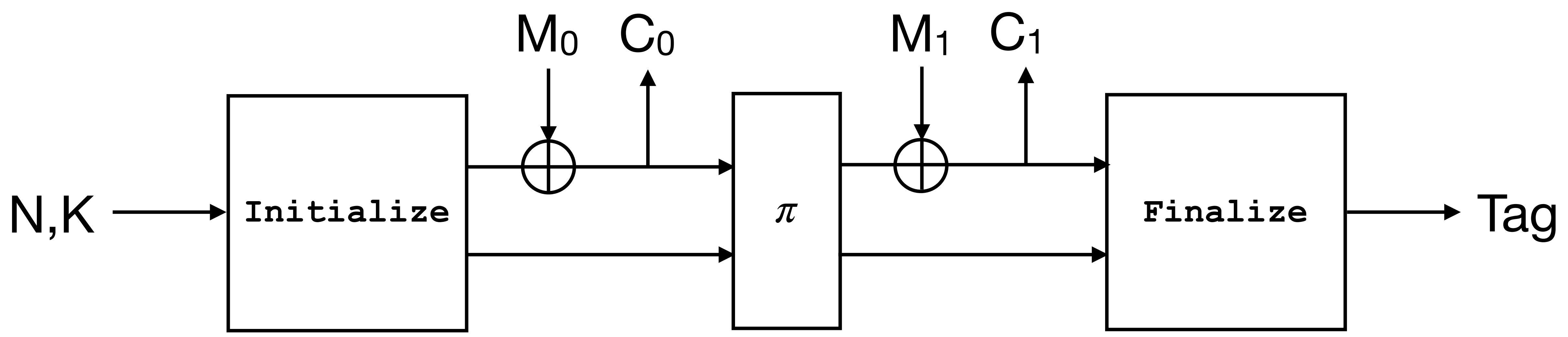
S1P mode in our attack setting

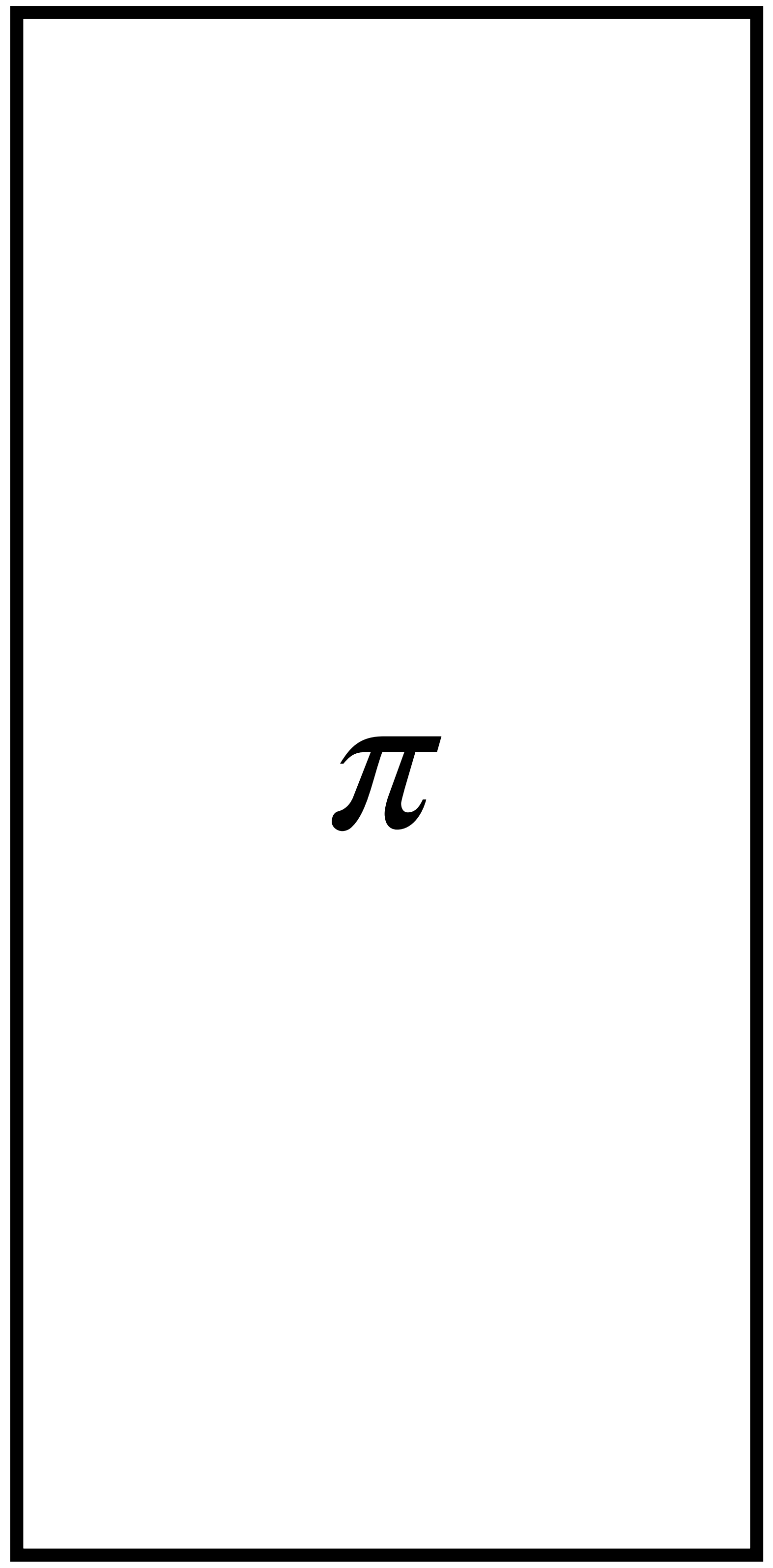


- "Aggressive parameters" (reduced version for cryptanalysis target): $\pi = 8$ rounds of Shadow-512
- Shifted version (step 2 to step 5)
- Same nonce used 3 times (**nonce misuse scenario**) to build collisions:

  2 different plaintexts that yield the **same tag**
- Probability of success of $\mathbf{2^{-24.83}}$

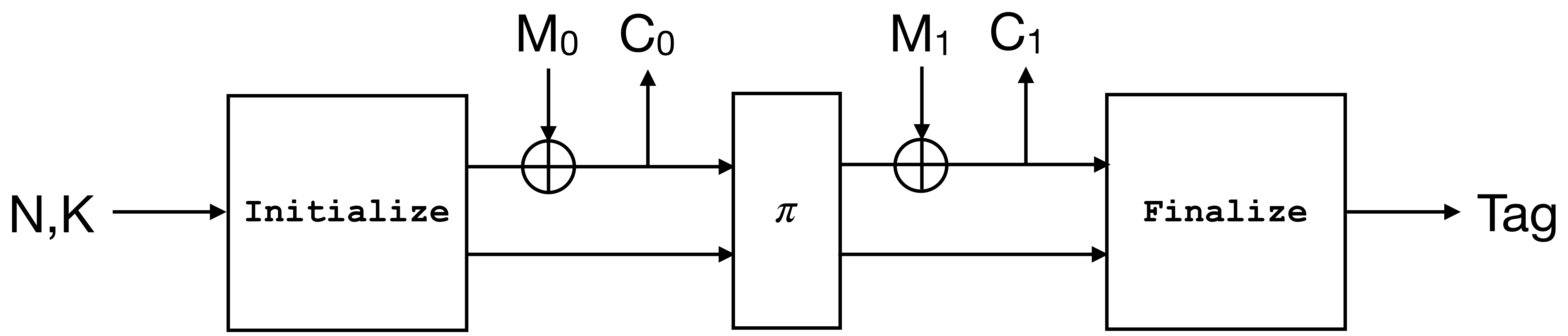
Forgery
Main ideas



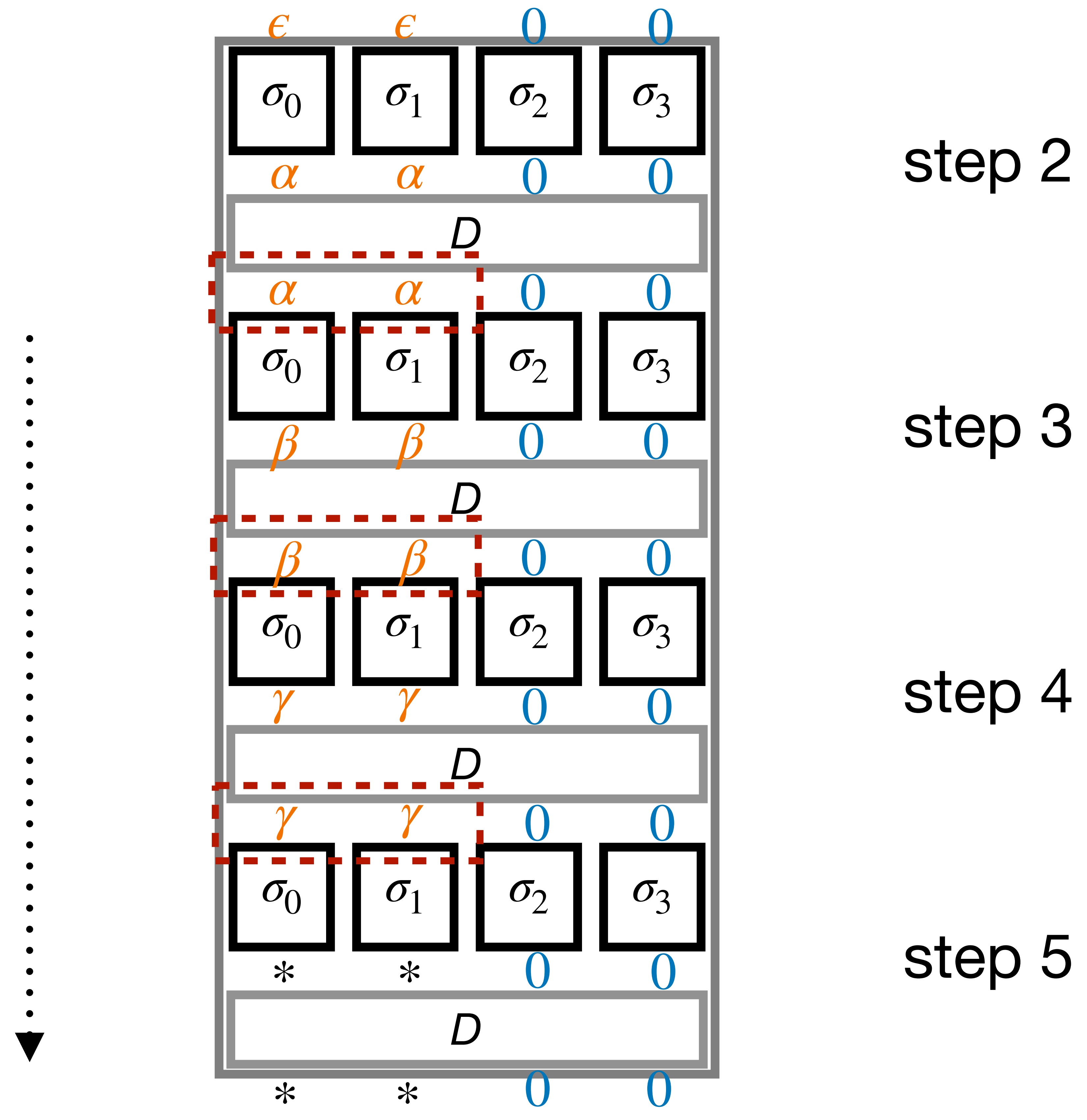


Forgery

Main ideas

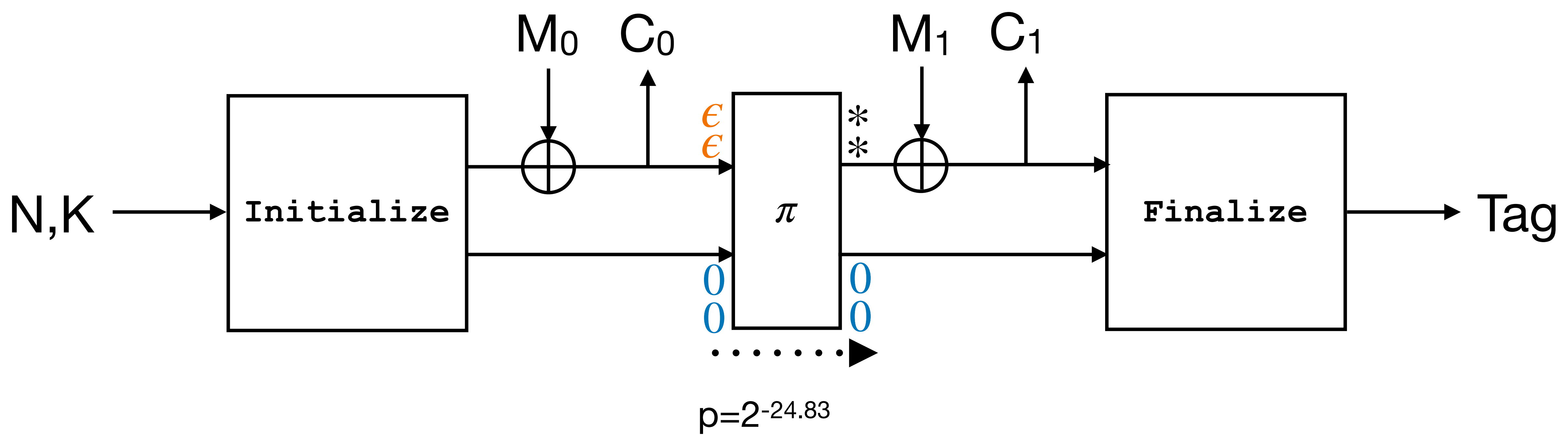


$p=2^{-24.83}$

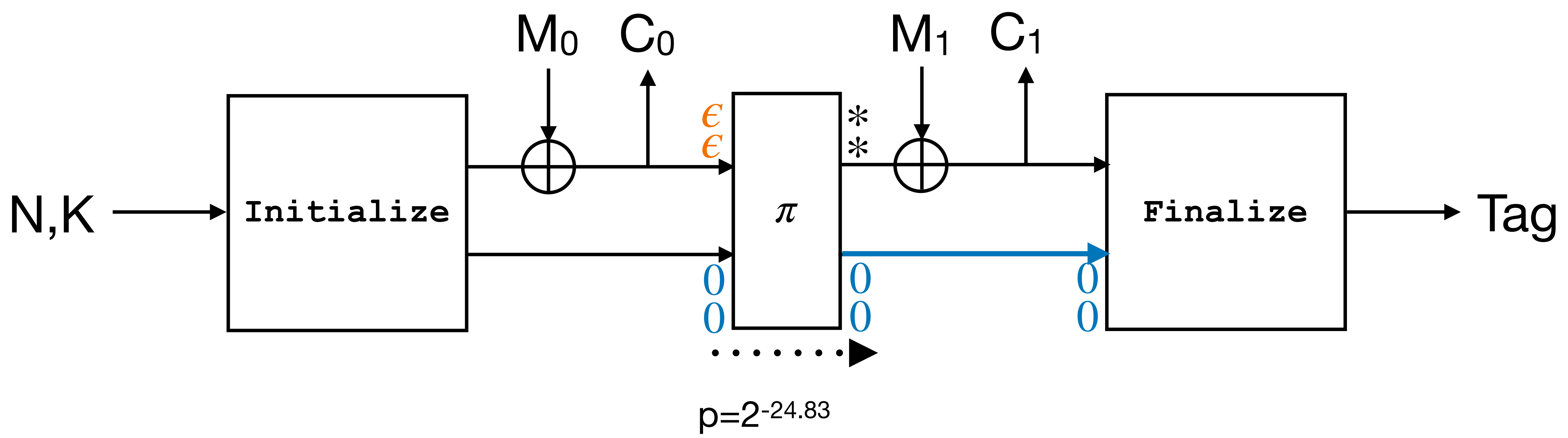


Forgery

Main ideas



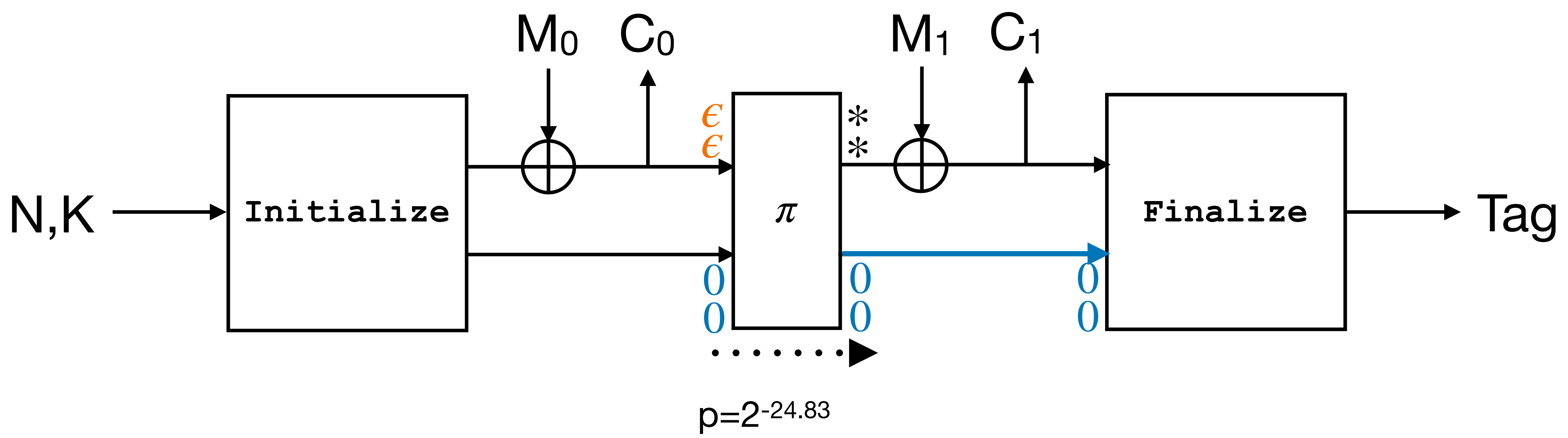
Forgery
Main ideas



Collision on the capacity part

Forgery

Main ideas

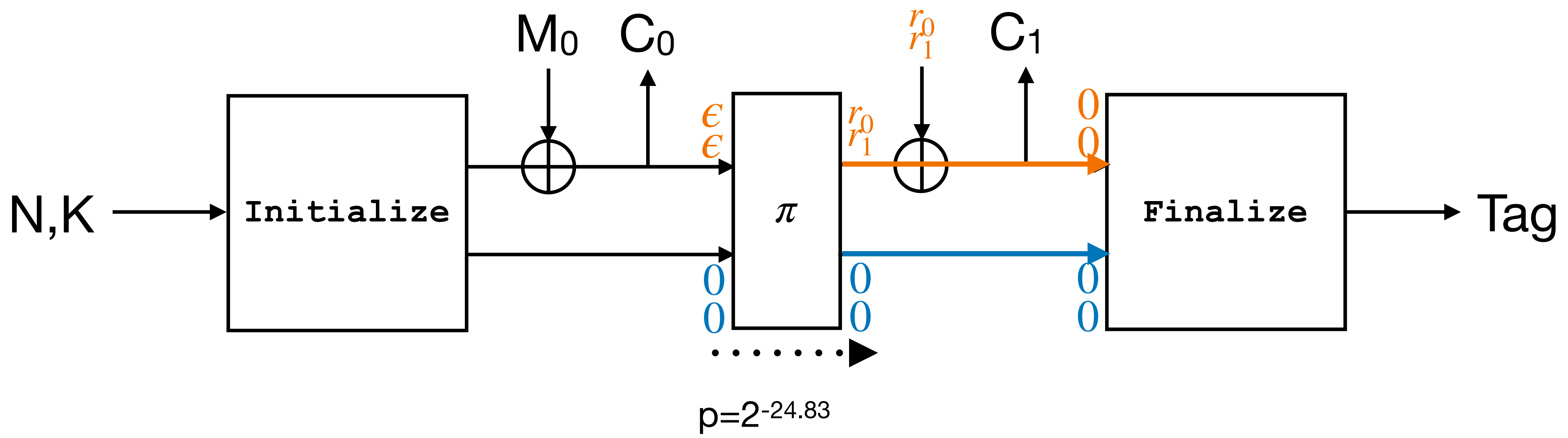


Collision on the rate part can be found using 3 queries

Collision on the capacity part

Forgery

Main ideas



Collision on the rate part can be found using 3 queries

Collision on the capacity part

Conclusion on Spook

- Summary of our work:
  - **Practical distinguishers** of the full 6-step version of Shadow-512 and Shadow-384 (shifted)
  - **Practical forgeries** with 4-step Shadow for the S1P mode of operation (nonce misuse scenario)
- After our results, the authors proposed **Spook v2** [ToSC special Issue]:
  - *D* matrix replaced with an efficient MDS matrix
  - modification of the round constants of Shadow for more efficiency
- New criterion for choosing round constants: prevent more than invariant subspaces attacks

Part III
Boomerang Attacks: the Feistel Case

Basic boomerang distinguisher

The Boomerang Attack
Wagner, *FSE 1999*

Variant of differential cryptanalysis that considers **quartets** of messages.

Basic boomerang distinguisher

The Boomerang Attack
Wagner, *FSE 1999*

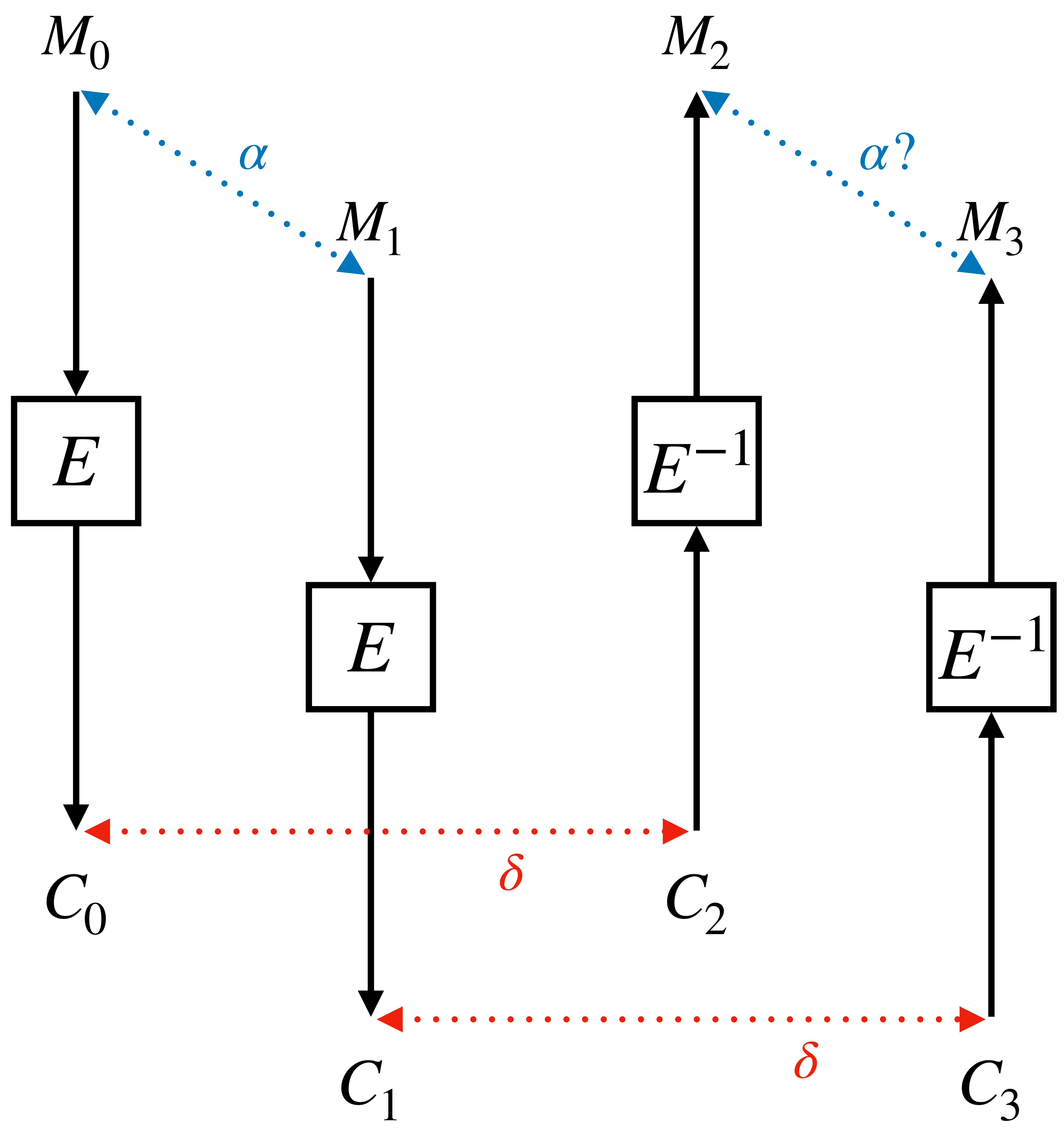
1. Pick M_0 at random, ask for its ciphertext C_0

2. Ask for C_1, the ciphertext of $M_1 = M_0 \oplus \alpha$

3. Compute $C_2 = C_0 \oplus \delta, C_3 = C_1 \oplus \delta$

4. Ask for their decryption (M_2, M_3)

5. Check if $M_2 \oplus M_3 = \alpha$



Basic boomerang distinguisher

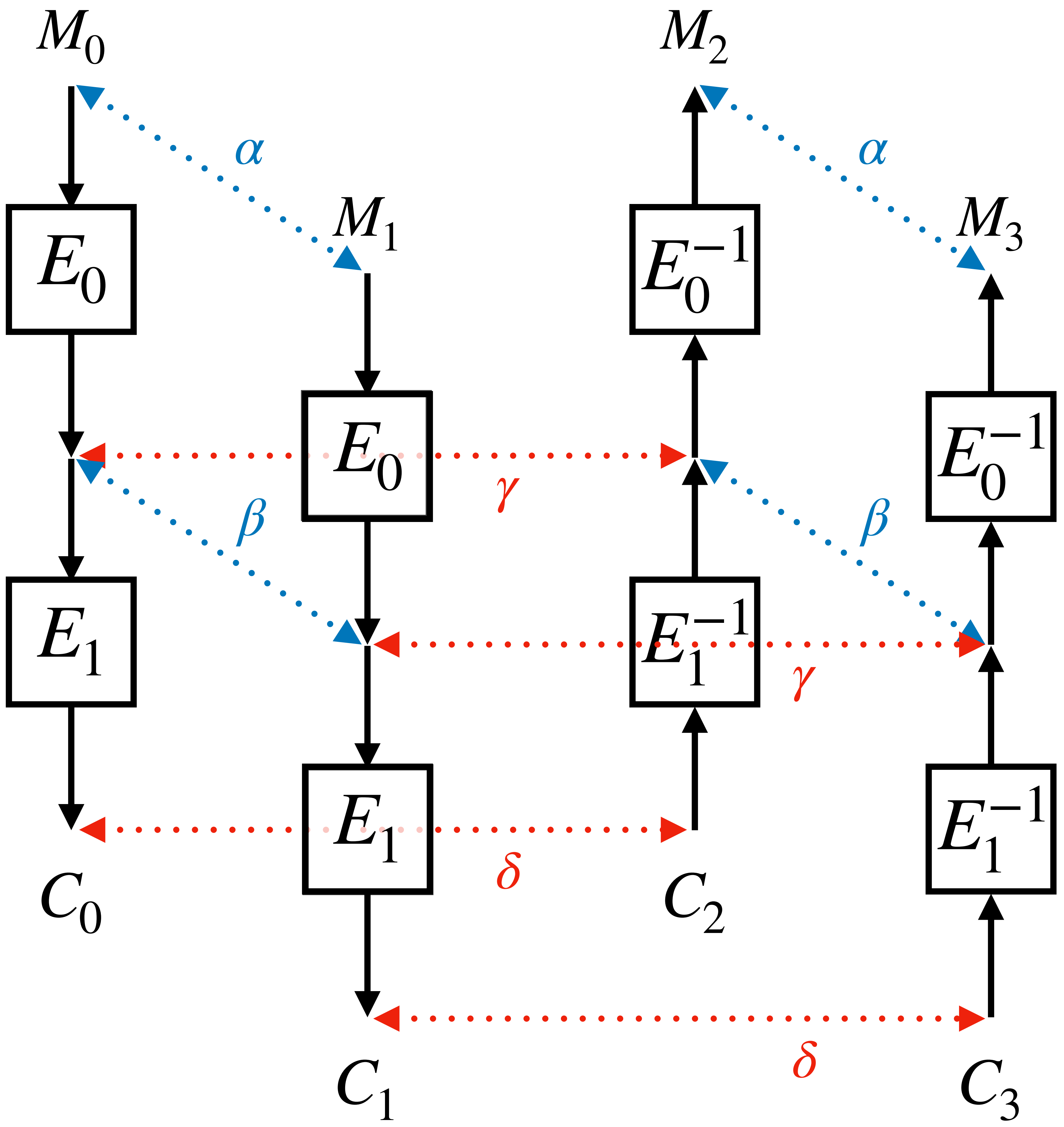
The Boomerang Attack
Wagner, *FSE 1999*

Rewrite $E = E_1 \circ E_0$

Find good differentials:

$$\mathbb{P}(\alpha \longrightarrow_{E_0} \beta) = p$$
$$\mathbb{P}(\gamma \longrightarrow_{E_1} \delta) = q$$

Expected probability of p^2q^2 if the two characteristics are “**independant**”.



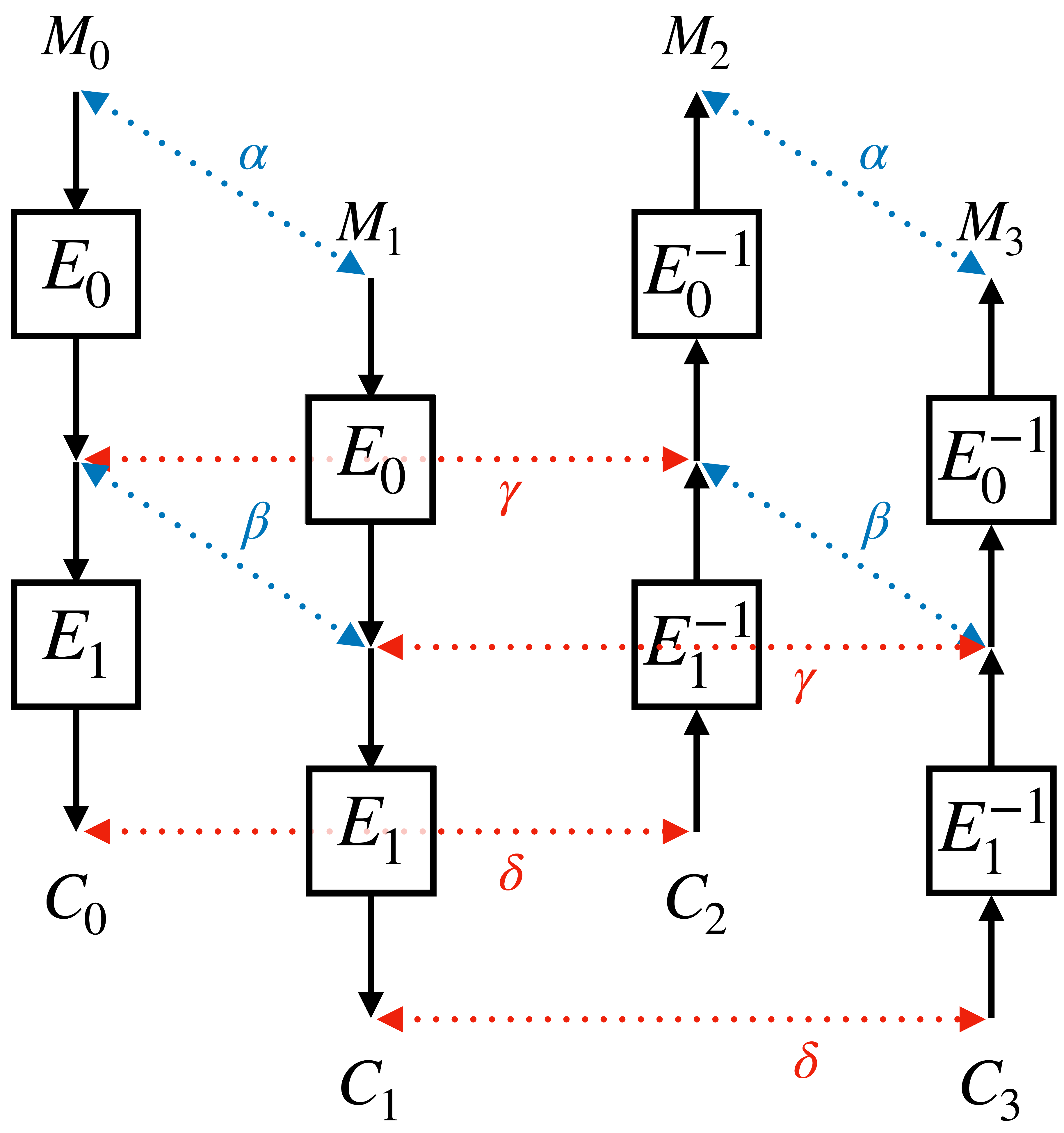
Basic boomerang distinguisher

Incompatibilities are discovered.

Related-key Cryptanalysis of the Full AES-192 and AES-256
Biryukov & Khovratovich, *ASIACRYPT 2009*

The Return of the Cryptographic Boomerang
Murphy, *IEEE Transactions on Information Theory 2011*

The problems come from interactions at the **junction** of the two trails.



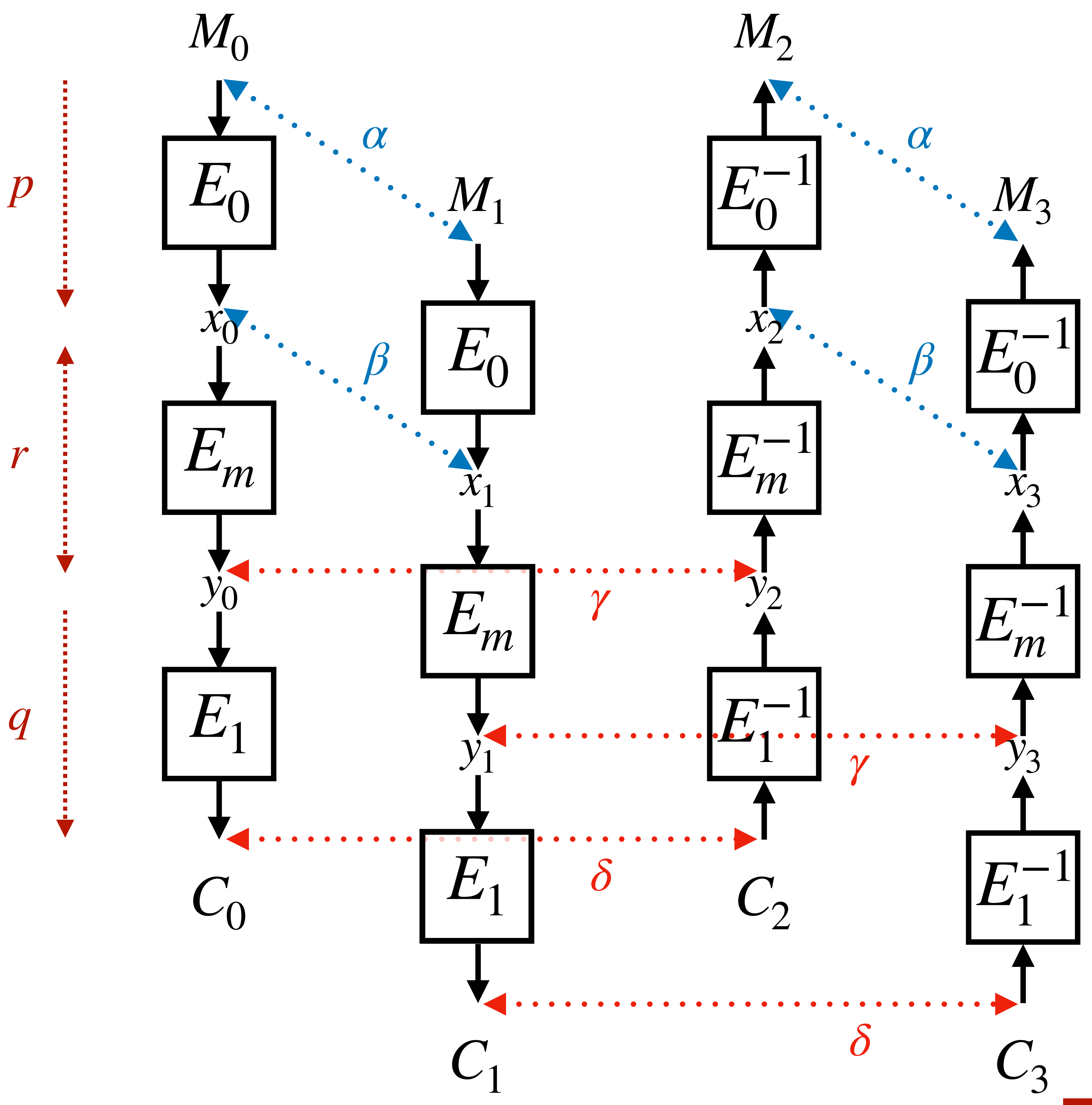
The sandwich attack

A Practical-time Related-key Attack on the KASUMI Cryptosystem Used in GSM and 3G Telephony
Dunkelman, Keller & Shamir, *CRYPTO 2010*

$E = E_1 \circ E_m \circ E_0$

E_m is 1 round (**boomerang switch**)

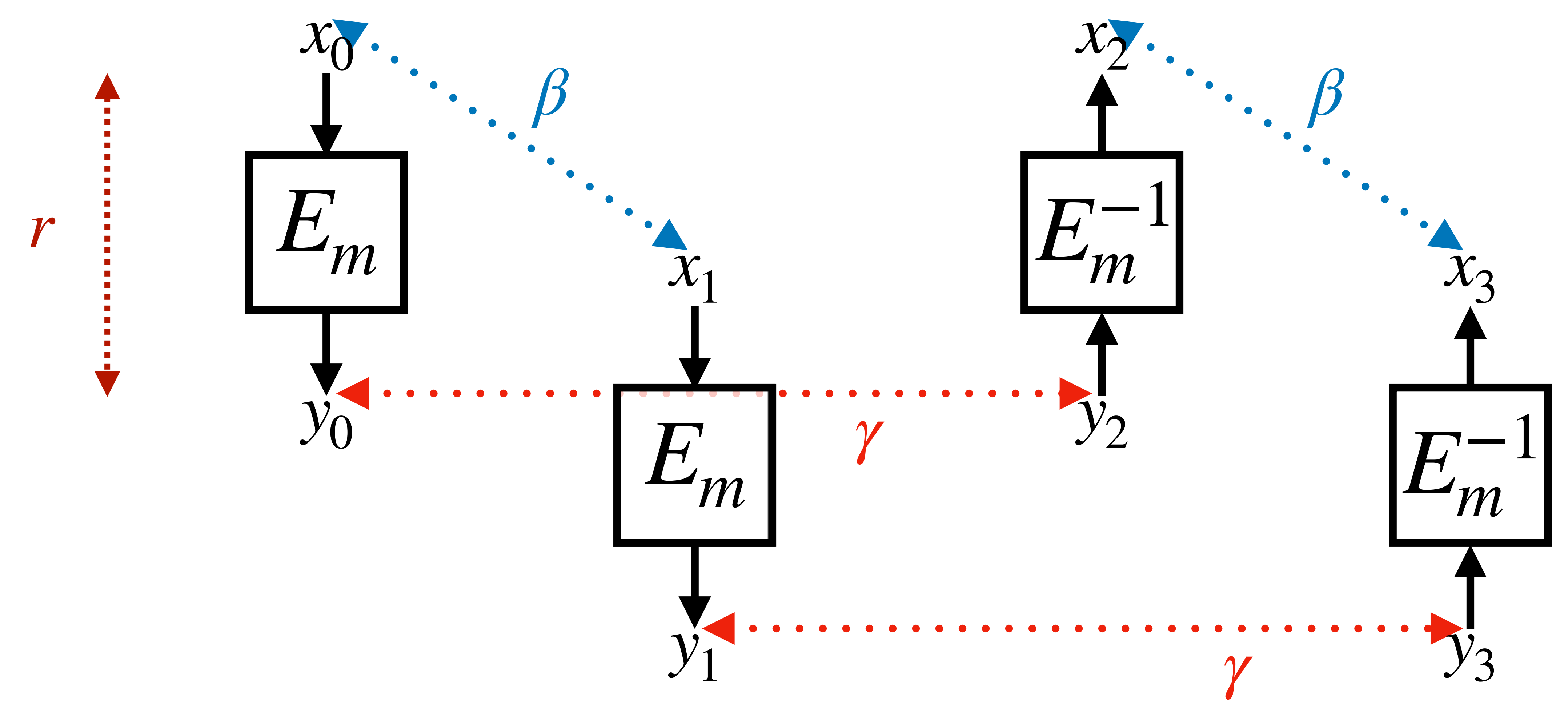
Expected probability of p^2q^2r



The sandwich attack

A Practical-time Related-key Attack on the KASUMI Cryptosystem Used in GSM and 3G Telephony
Dunkelman, Keller & Shamir, *CRYPTO 2010*

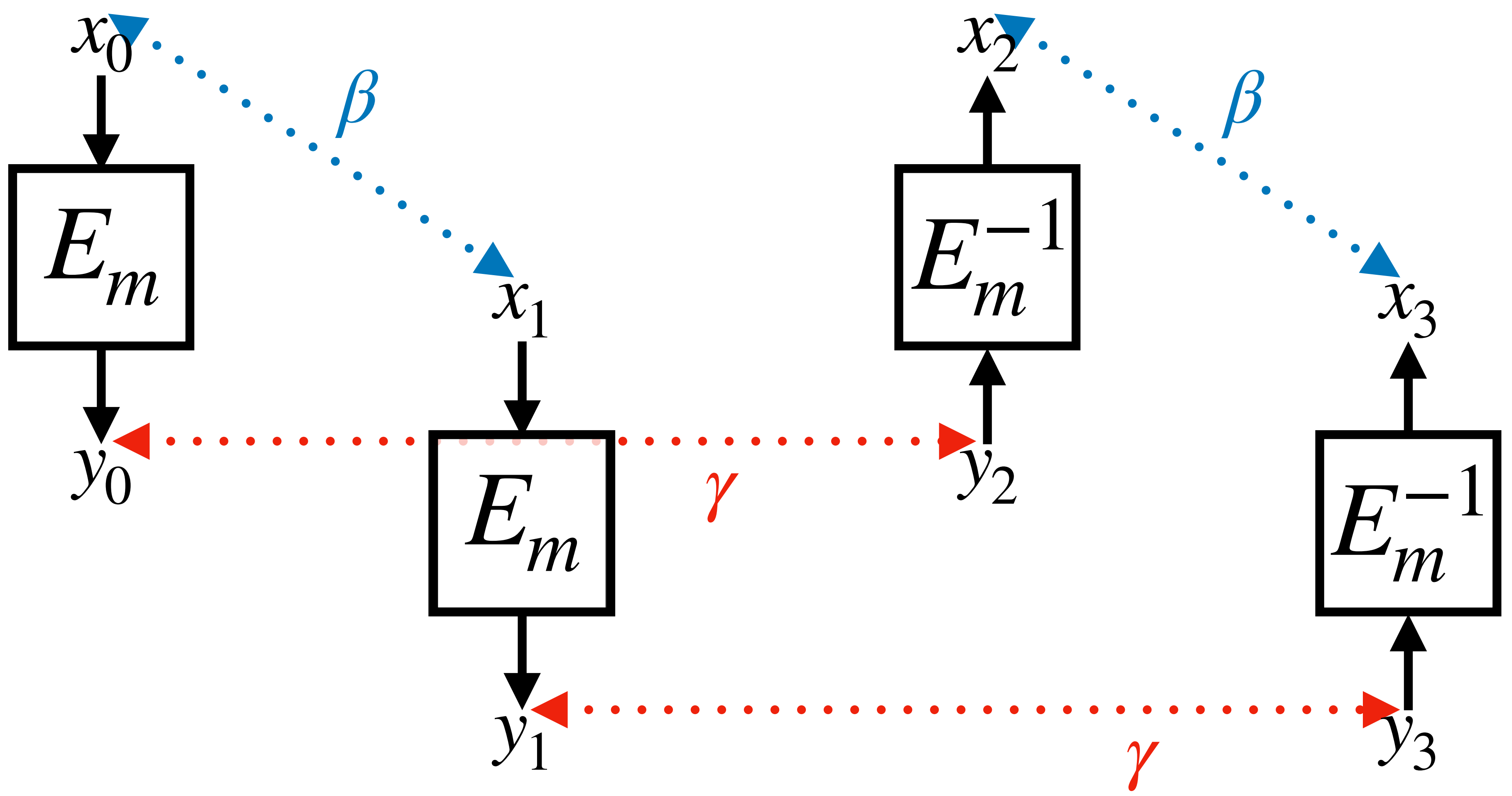
How to compute *r*?



The BCT: automated analysis for SPNs

Boomerang Connectivity Table: a New Cryptanalysis Tool
Cid, Huang, Peyrin, Sasaki & Song, *EUROCRYPT 2018*

	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16
1	16	0	8	0	4	4	4	4	0	0	2	2	0	0	2	2
2	16	0	8	0	4	4	4	4	0	0	2	2	0	0	2	2
3	16	16	0	0	0	0	0	0	8	8	0	0	8	8	0	0
4	16	8	0	2	0	0	0	2	4	4	2	0	4	4	2	0
5	16	8	0	2	0	0	0	2	4	4	0	2	4	4	0	2
6	16	8	0	2	0	0	0	2	4	4	0	2	4	4	0	2
7	16	8	0	2	0	0	0	2	4	4	2	0	4	4	2	0
8	16	0	0	0	0	0	0	0	2	2	2	2	2	2	2	2
9	16	0	8	0	4	4	4	4	0	0	2	2	0	0	2	2
a	16	0	8	0	4	4	4	4	2	2	0	0	2	2	0	0
b	16	0	16	0	8	8	8	8	0	0	0	0	0	0	0	0
c	16	0	0	2	2	2	2	0	2	2	2	0	0	0	0	2
d	16	0	0	2	2	2	2	0	0	0	2	0	2	2	0	2
e	16	0	0	2	2	2	2	0	2	2	0	2	0	0	2	0
f	16	0	0	2	2	2	2	0	0	0	0	2	2	2	2	0

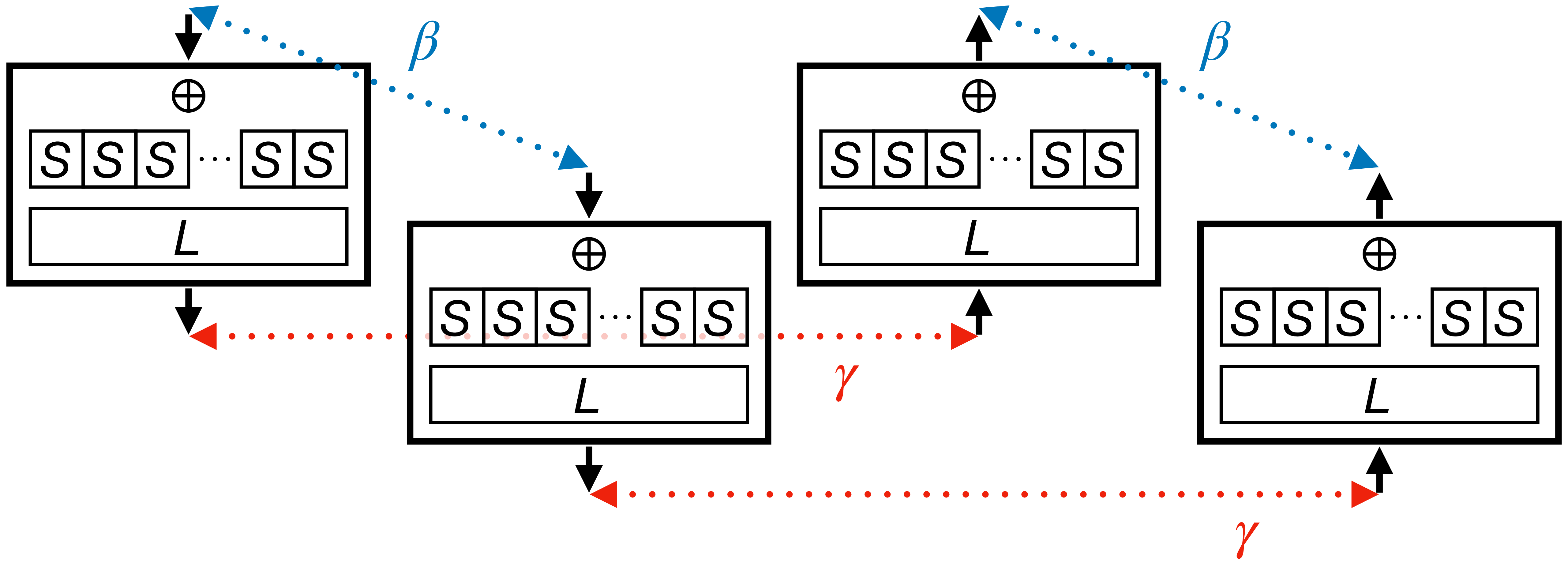


$$E_m^{-1}(E_m(X) \oplus \gamma) \oplus E_m^{-1}(E_m(X \oplus \beta) \oplus \gamma) = \beta$$

The BCT: automated analysis for SPNs

Boomerang Connectivity Table: a New Cryptanalysis Tool
Cid, Huang, Peyrin, Sasaki & Song, *EUROCRYPT 2018*

	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16
1	16	0	8	0	4	4	4	4	0	0	2	2	0	0	2	2
2	16	0	8	0	4	4	4	4	0	0	2	2	0	0	2	2
3	16	16	0	0	0	0	0	0	8	8	0	0	8	8	0	0
4	16	8	0	2	0	0	0	2	4	4	2	0	4	4	2	0
5	16	8	0	2	0	0	0	2	4	4	0	2	4	4	0	2
6	16	8	0	2	0	0	0	2	4	4	0	2	4	4	0	2
7	16	8	0	2	0	0	0	2	4	4	2	0	4	4	2	0
8	16	0	0	0	0	0	0	0	2	2	2	2	2	2	2	2
9	16	0	8	0	4	4	4	4	0	0	2	2	0	0	2	2
a	16	0	8	0	4	4	4	4	2	2	0	0	2	2	0	0
b	16	0	16	0	8	8	8	8	0	0	0	0	0	0	0	0
c	16	0	0	2	2	2	2	0	2	2	2	0	0	0	0	2
d	16	0	0	2	2	2	2	0	0	0	2	0	2	2	0	2
e	16	0	0	2	2	2	2	0	2	2	0	2	0	0	2	0
f	16	0	0	2	2	2	2	0	0	0	0	2	2	2	2	0

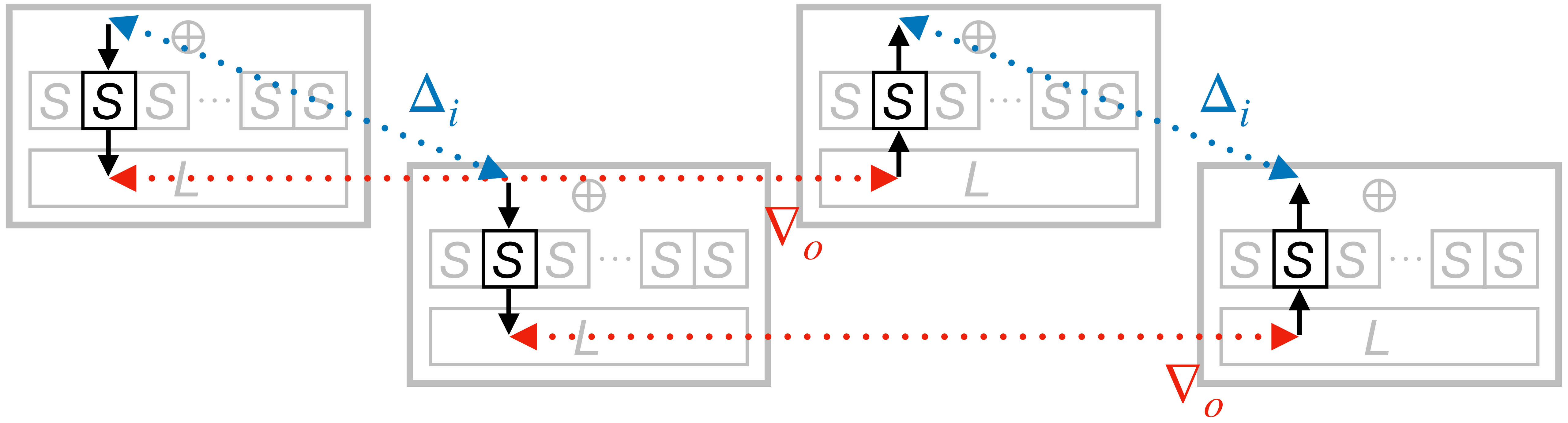


$$E_m^{-1}(E_m(X) \oplus \gamma) \oplus E_m^{-1}(E_m(X \oplus \beta) \oplus \gamma) = \beta$$

The BCT: automated analysis for SPNs

Boomerang Connectivity Table: a New Cryptanalysis Tool
Cid, Huang, Peyrin, Sasaki & Song, *EUROCRYPT 2018*

	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16
1	16	0	8	0	4	4	4	4	0	0	2	2	0	0	2	2
2	16	0	8	0	4	4	4	4	0	0	2	2	0	0	2	2
3	16	16	0	0	0	0	0	0	8	8	0	0	8	8	0	0
4	16	8	0	2	0	0	0	2	4	4	2	0	4	4	2	0
5	16	8	0	2	0	0	0	2	4	4	0	2	4	4	0	2
6	16	8	0	2	0	0	0	2	4	4	0	2	4	4	0	2
7	16	8	0	2	0	0	0	2	4	4	2	0	4	4	2	0
8	16	0	0	0	0	0	0	0	2	2	2	2	2	2	2	2
9	16	0	8	0	4	4	4	4	0	0	2	2	0	0	2	2
a	16	0	8	0	4	4	4	4	2	2	0	0	2	2	0	0
b	16	0	16	0	8	8	8	8	0	0	0	0	0	0	0	0
c	16	0	0	2	2	2	2	0	2	2	2	0	0	0	0	2
d	16	0	0	2	2	2	2	0	0	0	2	0	2	2	0	2
e	16	0	0	2	2	2	2	0	2	2	0	2	0	0	2	0
f	16	0	0	2	2	2	2	0	0	0	0	2	2	2	2	0

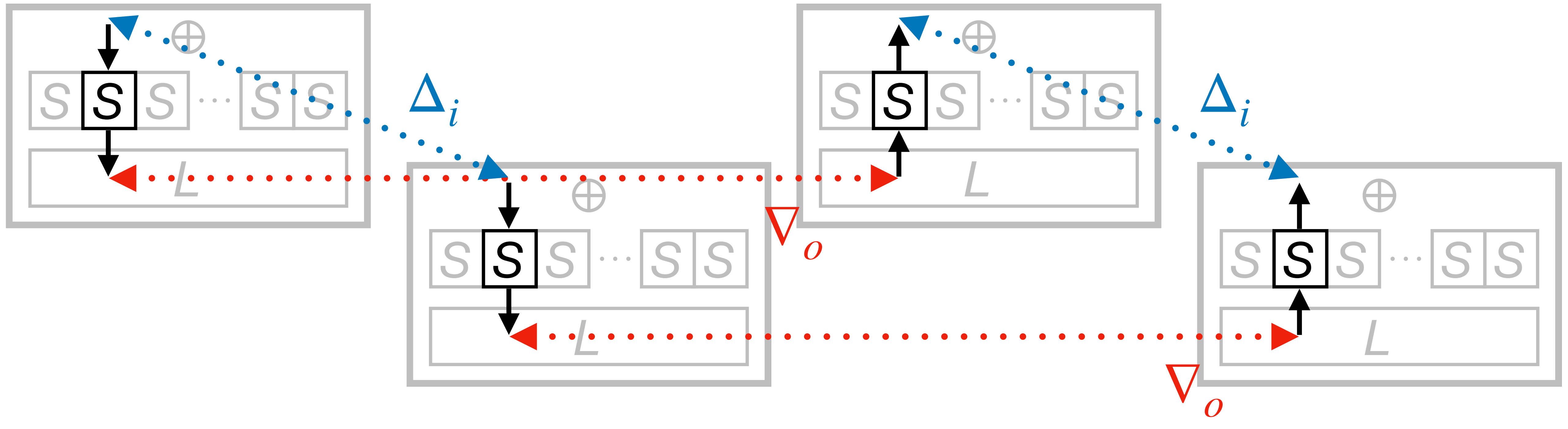


$$S^{-1}(S(x) \oplus \nabla_o) \oplus S^{-1}(S(x \oplus \Delta_i) \oplus \nabla_o) = \Delta_i$$

The BCT: automated analysis for SPNs

Boomerang Connectivity Table: a New Cryptanalysis Tool
Cid, Huang, Peyrin, Sasaki & Song, *EUROCRYPT 2018*

	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16
1	16	0	8	0	4	4	4	4	0	0	2	2	0	0	2	2
2	16	0	8	0	4	4	4	4	0	0	2	2	0	0	2	2
3	16	16	0	0	0	0	0	0	8	8	0	0	8	8	0	0
4	16	8	0	2	0	0	0	2	4	4	2	0	4	4	2	0
5	16	8	0	2	0	0	0	2	4	4	0	2	4	4	0	2
6	16	8	0	2	0	0	0	2	4	4	0	2	4	4	0	2
7	16	8	0	2	0	0	0	2	4	4	2	0	4	4	2	0
8	16	0	0	0	0	0	0	0	2	2	2	2	2	2	2	2
9	16	0	8	0	4	4	4	4	0	0	2	2	0	0	2	2
a	16	0	8	0	4	4	4	4	2	2	0	0	2	2	0	0
b	16	0	16	0	8	8	8	8	0	0	0	0	0	0	0	0
c	16	0	0	2	2	2	2	0	2	2	2	0	0	0	0	2
d	16	0	0	2	2	2	2	0	0	0	2	0	2	2	0	2
e	16	0	0	2	2	2	2	0	2	2	0	2	0	0	2	0
f	16	0	0	2	2	2	2	0	0	0	0	2	2	2	2	0



$$\mathrm{BCT}(\Delta_i, \nabla_o) = \#\{x \mid S^{-1}(S(x) \oplus \nabla_o) \oplus S^{-1}(S(x \oplus \Delta_i) \oplus \nabla_o) = \Delta_i\}$$

The BCT: automated analysis for SPNs

Boomerang Connectivity Table: a New Cryptanalysis Tool
Cid, Huang, Peyrin, Sasaki & Song, *EUROCRYPT 2018*

	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16
1	16	0	8	0	4	4	4	4	0	0	2	2	0	0	2	2
2	16	0	8	0	4	4	4	4	0	0	2	2	0	0	2	2
3	16	16	0	0	0	0	0	0	8	8	0	0	8	8	0	0
4	16	8	0	2	0	0	0	2	4	4	2	0	4	4	2	0
5	16	8	0	2	0	0	0	2	4	4	0	2	4	4	0	2
6	16	8	0	2	0	0	0	2	4	4	0	2	4	4	0	2
7	16	8	0	2	0	0	0	2	4	4	2	0	4	4	2	0
8	16	0	0	0	0	0	0	0	2	2	2	2	2	2	2	2
9	16	0	8	0	4	4	4	4	0	0	2	2	0	0	2	2
a	16	0	8	0	4	4	4	4	2	2	0	0	2	2	0	0
b	16	0	16	0	8	8	8	8	0	0	0	0	0	0	0	0
c	16	0	0	2	2	2	2	0	2	2	2	0	0	0	0	2
d	16	0	0	2	2	2	2	0	0	0	2	0	2	2	0	2
e	16	0	0	2	2	2	2	0	2	2	0	2	0	0	2	0
f	16	0	0	2	2	2	2	0	0	0	0	2	2	2	2	0

Probability over 1 round of SPN

↓

Probability over each S-box

Easily gives incompatibility, **Ladder switch**

New criteria for the choice of S-boxes

$$\boxed{\mathrm{BCT}(\Delta_i, \nabla_o) = \#\{x \mid S^{-1}(S(x) \oplus \nabla_o) \oplus S^{-1}(S(x \oplus \Delta_i) \oplus \nabla_o) = \Delta_i\}}$$

The BCT: automated analysis for SPNs

Boomerang Connectivity Table: a New Cryptanalysis Tool
Cid, Huang, Peyrin, Sasaki & Song, *EUROCRYPT 2018*

	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16
1	16	0	8	0	4	4	4	4	0	0	2	2	0	0	2	2
2	16	0	8	0	4	4	4	4	0	0	2	2	0	0	2	2
3	16	16	0	0	0	0	0	0	8	8	0	0	8	8	0	0
4	16	8	0	2	0	0	0	2	4	4	2	0	4	4	2	0
5	16	8	0	2	0	0	0	2	4	4	0	2	4	4	0	2
6	16	8	0	2	0	0	0	2	4	4	0	2	4	4	0	2
7	16	8	0	2	0	0	0	2	4	4	2	0	4	4	2	0
8	16	0	0	0	0	0	0	0	2	2	2	2	2	2	2	2
9	16	0	8	0	4	4	4	4	0	0	2	2	0	0	2	2
a	16	0	8	0	4	4	4	4	2	2	0	0	2	2	0	0
b	16	0	16	0	8	8	8	8	0	0	0	0	0	0	0	0
c	16	0	0	2	2	2	2	0	2	2	2	0	0	0	0	2
d	16	0	0	2	2	2	2	0	0	0	2	0	2	2	0	2
e	16	0	0	2	2	2	2	0	2	2	0	2	0	0	2	0
f	16	0	0	2	2	2	2	0	0	0	0	2	2	2	2	0

What about Feistel ciphers ?

$$\boxed{\mathrm{BCT}(\Delta_i, \nabla_o) = \#\{x \mid S^{-1}(S(x) \oplus \nabla_o) \oplus S^{-1}(S(x \oplus \Delta_i) \oplus \nabla_o) = \Delta_i\}}$$

The FBCT: the Feistel case

Boomerang Connectivity Table: a New Cryptanalysis Tool
Cid, Huang, Peyrin, Sasaki & Song, *EUROCRYPT 2018*

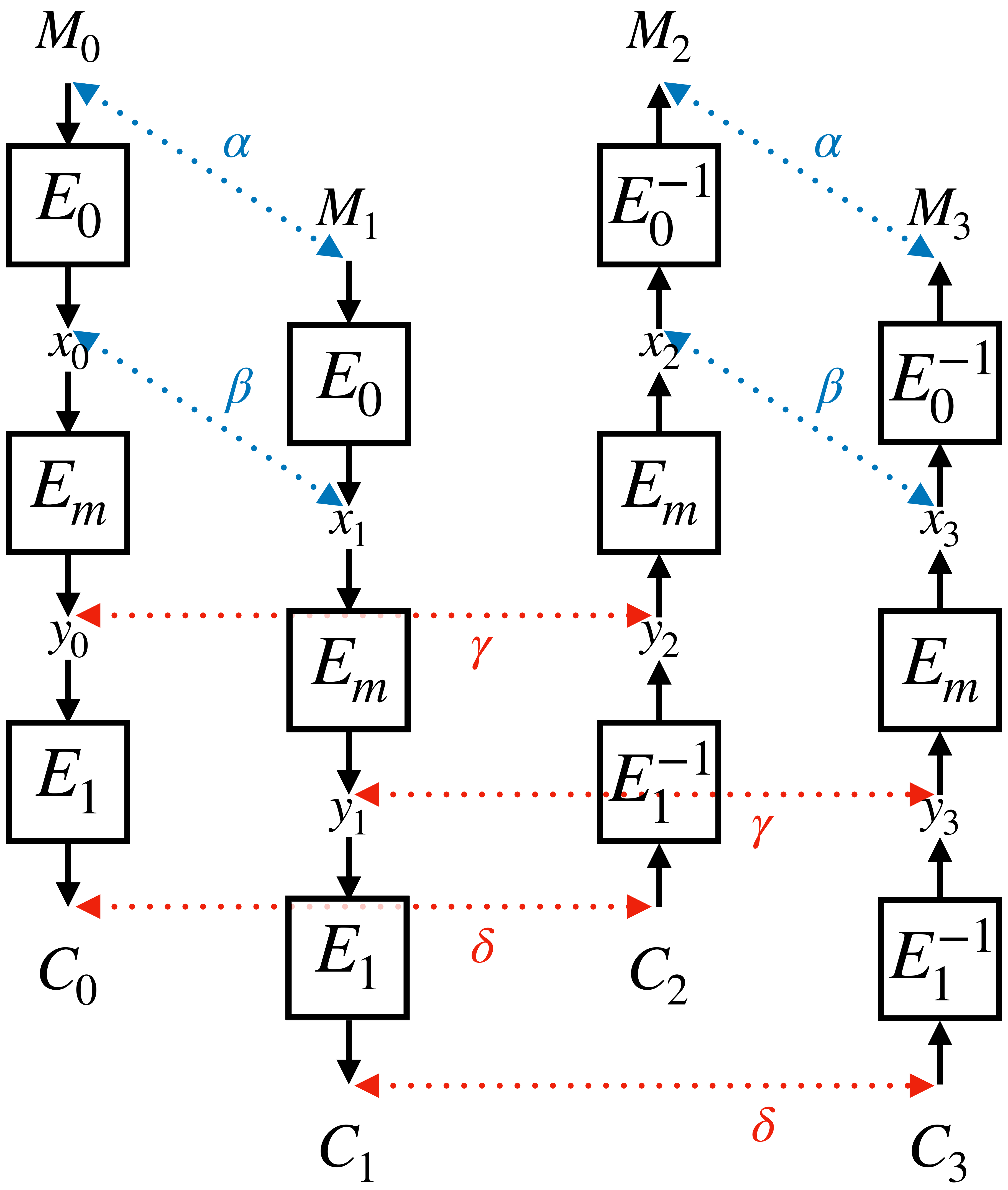
	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16
1	16	0	8	0	4	4	4	4	0	0	2	2	0	0	2	2
2	16	0	8	0	4	4	4	4	0	0	2	2	0	0	2	2
3	16	16	0	0	0	0	0	0	8	8	0	0	8	8	0	0
4	16	8	0	2	0	0	0	2	4	4	2	0	4	4	2	0
5	16	8	0	2	0	0	0	2	4	4	0	2	4	4	0	2
6	16	8	0	2	0	0	0	2	4	4	0	2	4	4	0	2
7	16	8	0	2	0	0	0	2	4	4	2	0	4	4	2	0
8	16	0	0	0	0	0	0	0	2	2	2	2	2	2	2	2
9	16	0	8	0	4	4	4	4	0	0	2	2	0	0	2	2
a	16	0	8	0	4	4	4	4	2	2	0	0	2	2	0	0
b	16	0	16	0	8	8	8	8	0	0	0	0	0	0	0	0
c	16	0	0	2	2	2	2	0	2	2	2	0	0	0	0	2
d	16	0	0	2	2	2	2	0	0	0	2	0	2	2	0	2
e	16	0	0	2	2	2	2	0	2	2	0	2	0	0	2	0
f	16	0	0	2	2	2	2	0	0	0	0	2	2	2	2	0

This work

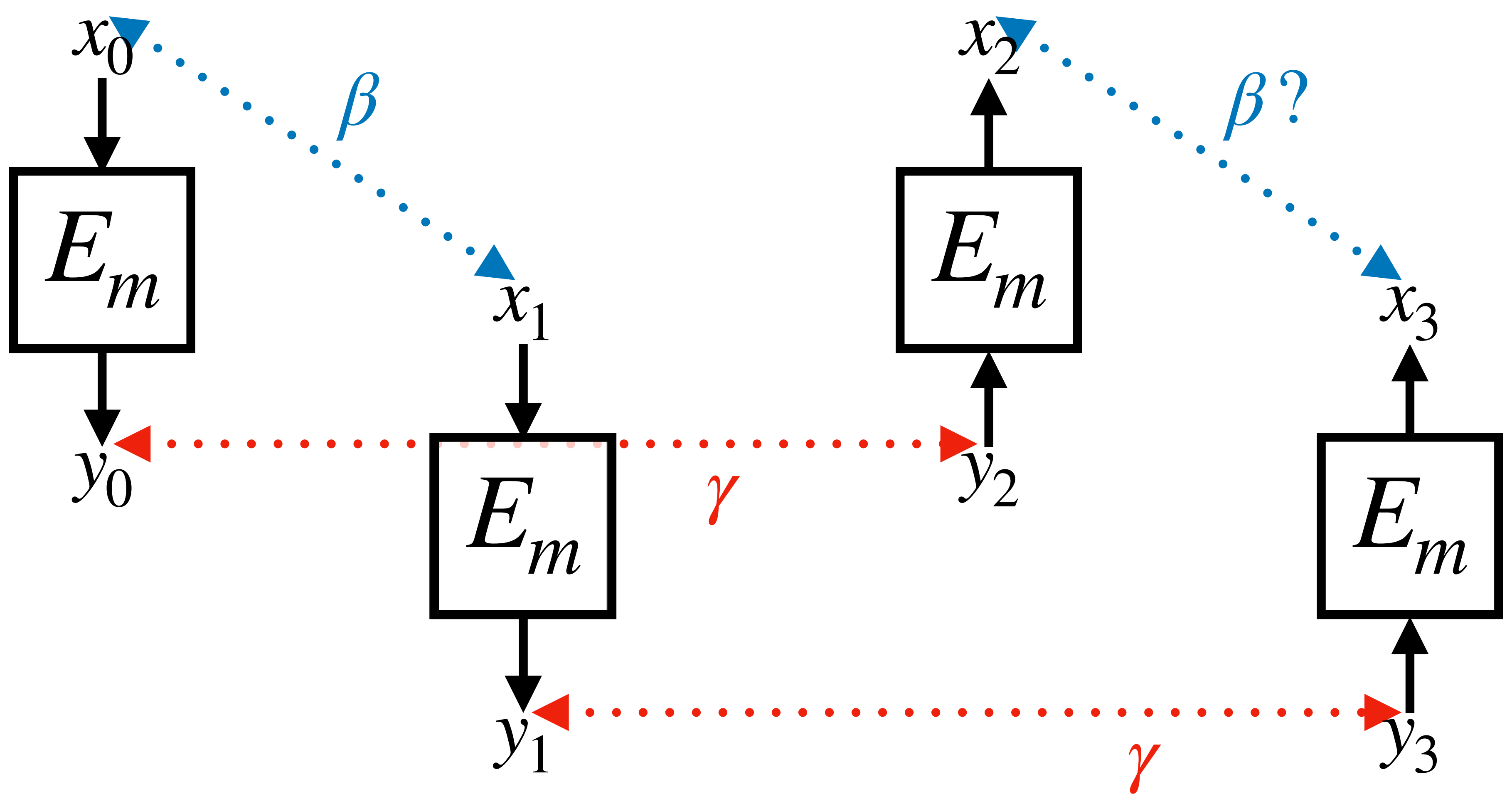
	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16
1	16	16	0	0	0	0	0	0	0	0	8	8	0	0	0	0
2	16	0	16	0	0	0	0	0	0	0	0	8	0	0	0	0
3	16	0	0	16	8	8	8	8	0	0	0	0	0	0	0	0
4	16	0	0	8	16	0	0	8	0	0	0	0	0	0	0	0
5	16	0	0	8	0	16	8	0	0	0	0	0	0	0	0	0
6	16	0	0	8	0	8	16	0	0	0	0	0	0	0	0	0
7	16	0	0	8	8	0	0	16	0	0	0	0	0	0	0	0
8	16	0	0	0	0	0	0	0	16	0	0	0	0	0	0	0
9	16	0	8	0	0	0	0	0	0	16	0	8	0	0	0	0
a	16	8	0	0	0	0	0	0	0	0	16	8	0	0	0	0
b	16	8	8	0	0	0	0	0	0	8	8	16	0	0	0	0
c	16	0	0	0	0	0	0	0	0	0	0	0	16	0	0	0
d	16	0	0	0	0	0	0	0	0	0	0	0	0	16	0	0
e	16	0	0	0	0	0	0	0	0	0	0	0	0	0	16	0
f	16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	16

$$\boxed{\mathrm{FBCT}(\Delta_i, \nabla_o) = \#\{x \,|\, S(x) \oplus S(x \oplus \Delta_i) \oplus S(x \oplus \nabla_o) \oplus S(x \oplus \Delta_i \oplus \nabla_o) = 0\}}$$

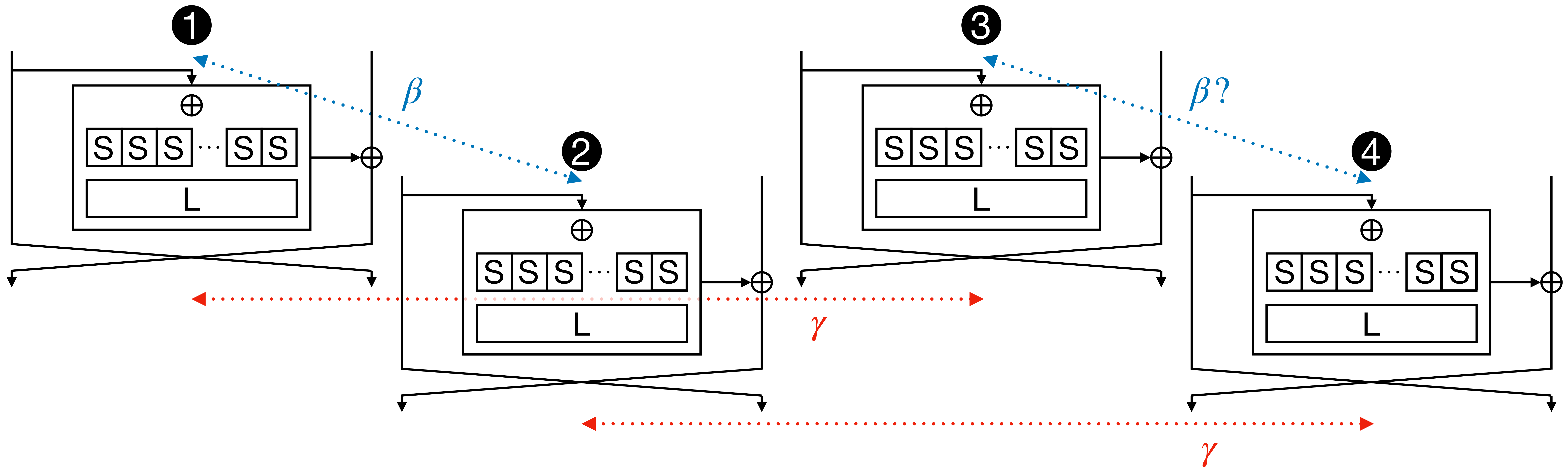
The Feistel counterpart of the BCT



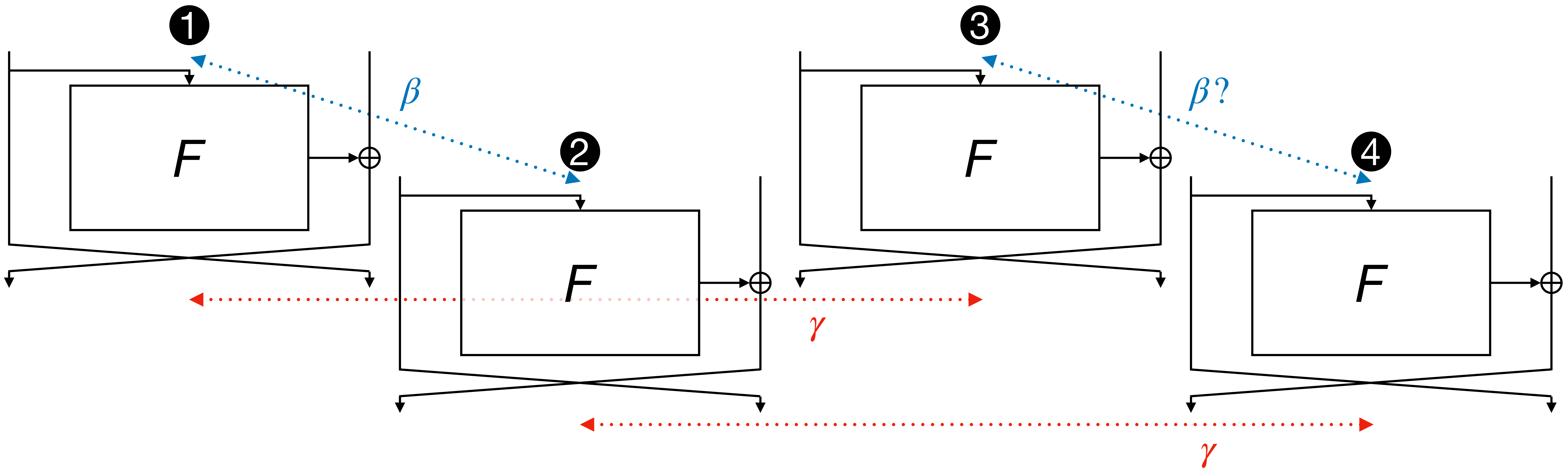
The Feistel counterpart of the BCT



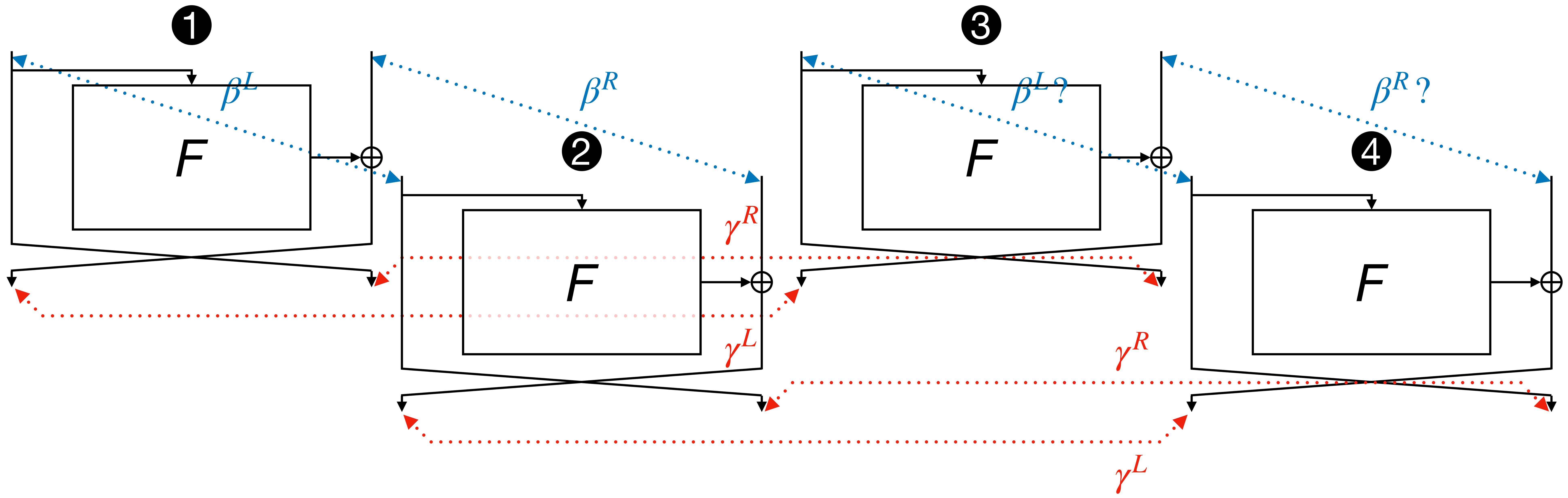
The FBCT



The FBCT

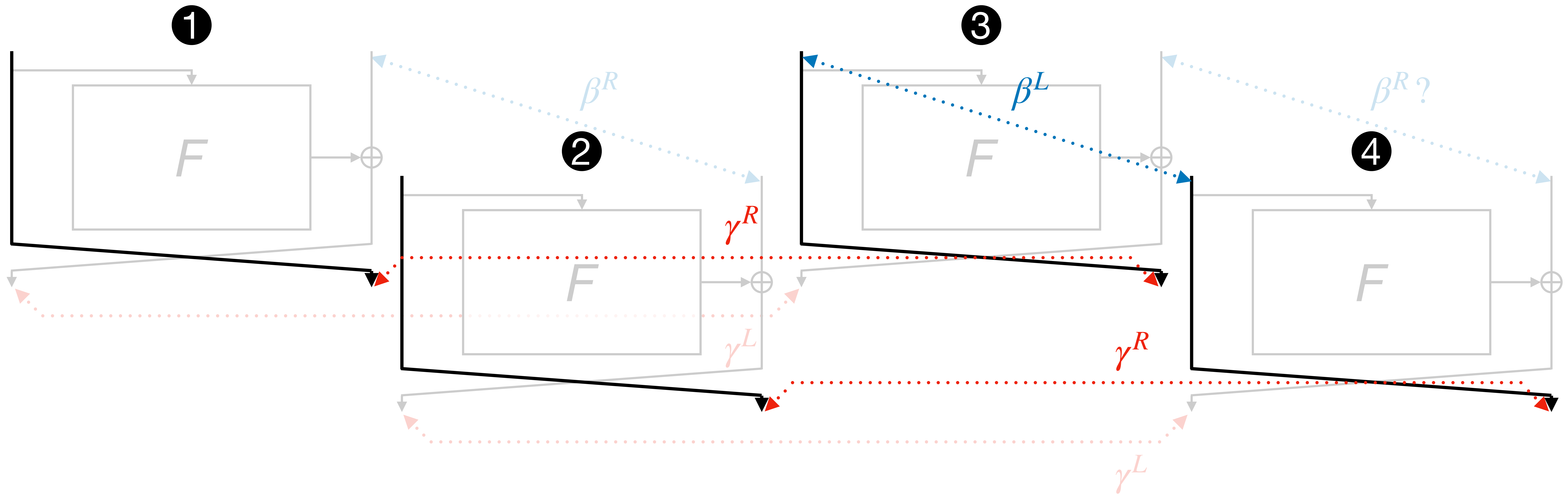


The FBCT



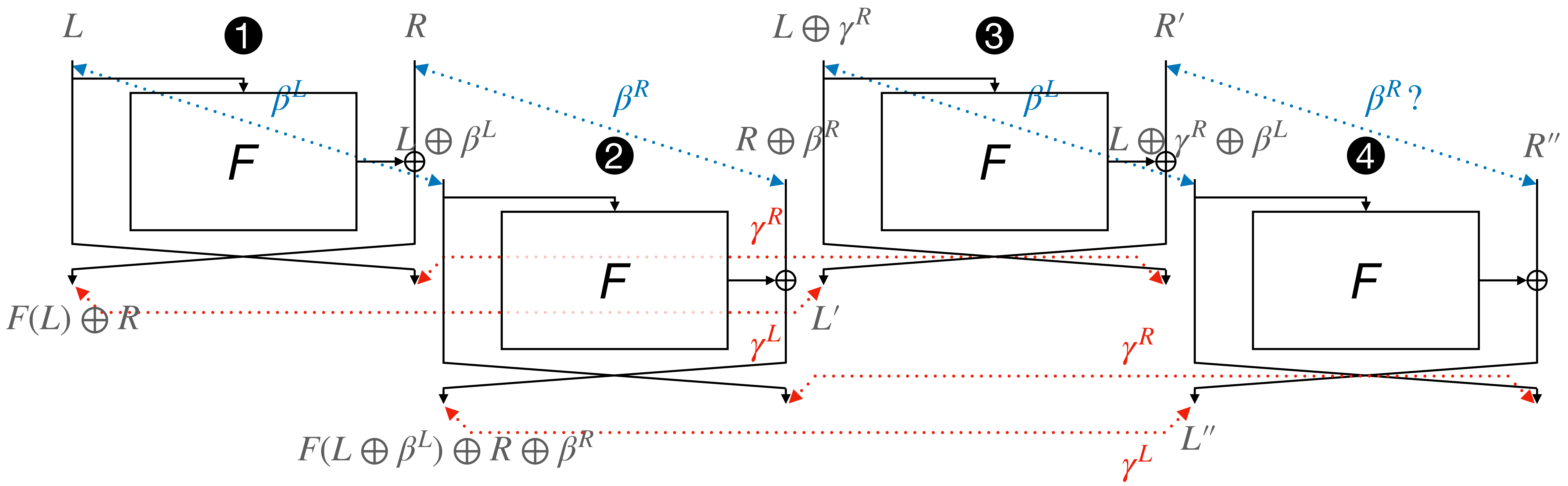
❶
❷
❸
❹
F
F
F
F
β^L
β^R
β^L ?
β^R ?
γ^R
γ^L
γ^R
γ^L

The FBCT (left part)



The left part of the difference comes for free.

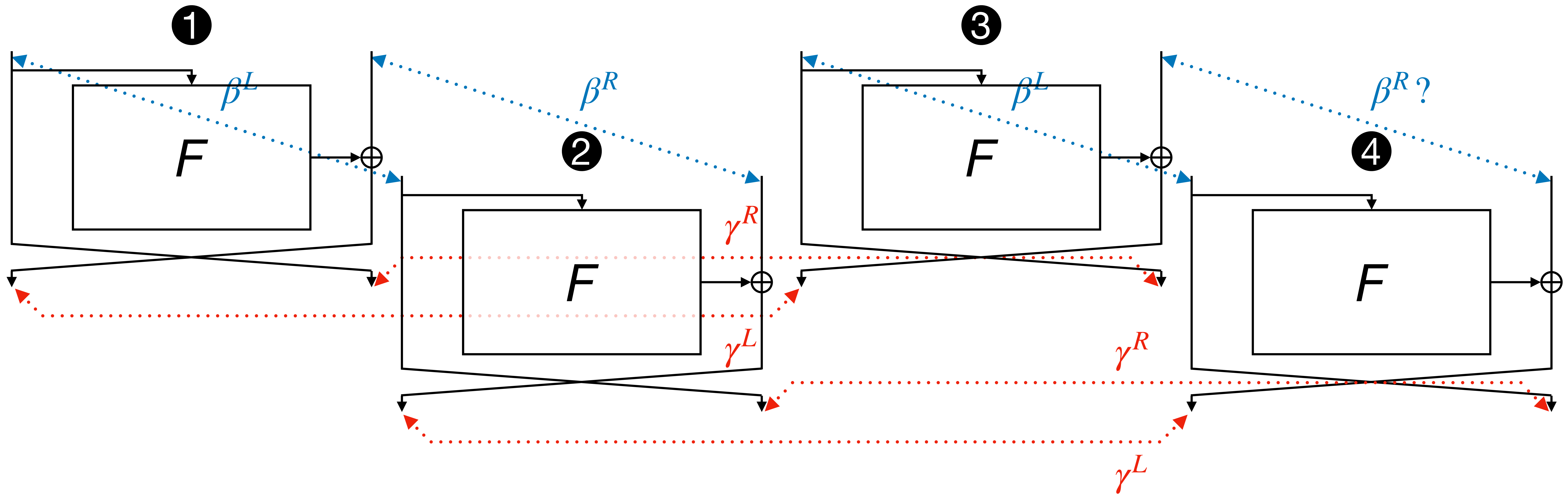
The FBCT (right part)



We want that $R' \oplus R'' = \beta^R$

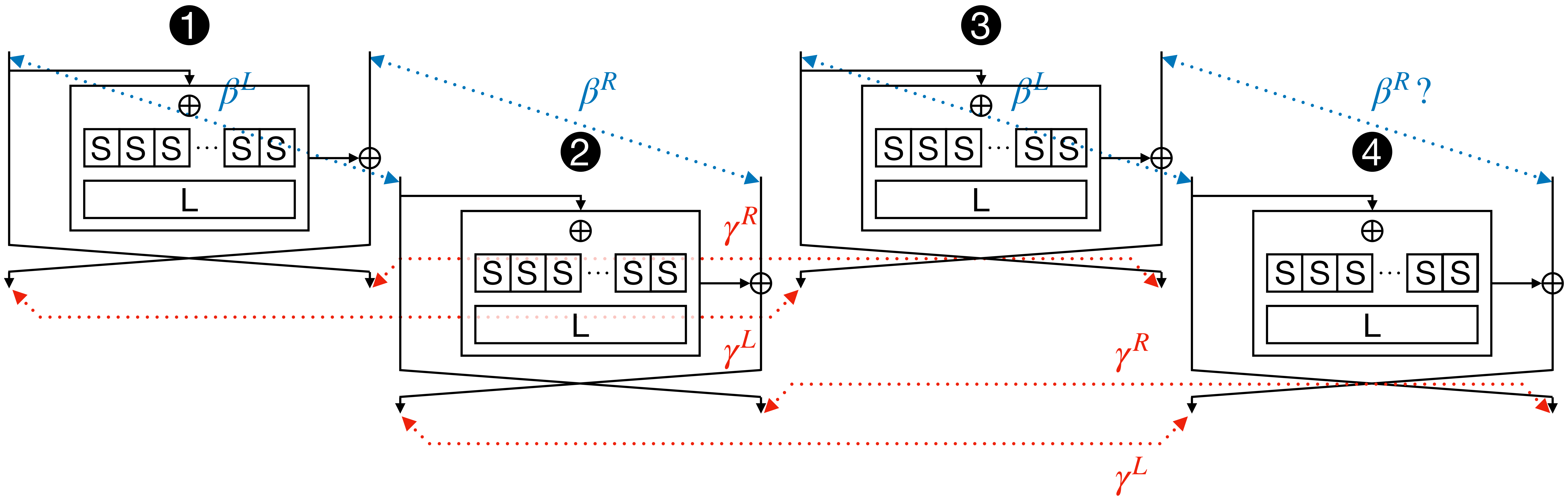
$$R' \oplus R'' = \underbrace{F(L \oplus \gamma^R) \oplus F(L) \oplus F(L \oplus \gamma^R \oplus \beta^L) \oplus F(L \oplus \beta^L)}_{0} \oplus \beta^R$$

The FBCT (right part)



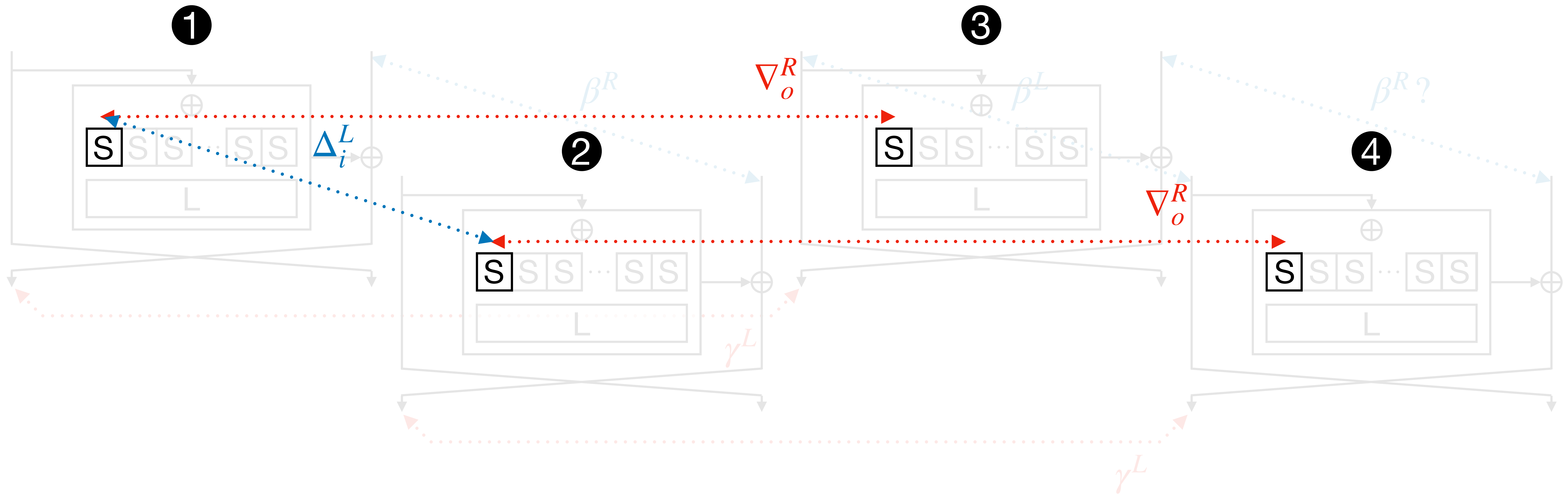
$$F(L \oplus \gamma^R) \oplus F(L) \oplus F(L \oplus \gamma^R \oplus \beta^L) \oplus F(L \oplus \beta^L) = 0$$

The FBCT (right part)



$$F(L \oplus \gamma^R) \oplus F(L) \oplus F(L \oplus \gamma^R \oplus \beta^L) \oplus F(L \oplus \beta^L) = 0$$

The FBCT (right part)



$$F(L \oplus \gamma^R) \oplus F(L) \oplus F(L \oplus \gamma^R \oplus \beta^L) \oplus F(L \oplus \beta^L) = 0$$

$$S(x \oplus \nabla_o^R) \oplus S(x) \oplus S(x \oplus \nabla_o^R \oplus \Delta_i^L) \oplus S(x \oplus \Delta_i^L) = 0$$

second derivative canceling out

Some properties of the FBCT

Symmetry: $\mathrm{FBCT}(\Delta_i, \nabla_o) = \mathrm{FBCT}(\nabla_o, \Delta_i)$

Diagonal: $\mathrm{FBCT}(\Delta_i, \Delta_i) = 2^n$

Multiplicity: $\mathrm{FBCT}(\Delta_i, \nabla_o) \equiv 0 \pmod 4$

Equalities: $\mathrm{FBCT}(\Delta_i, \nabla_o) = \mathrm{FBCT}(\Delta_i, \Delta_i \oplus \nabla_o)$

	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16
1	16	16	0	0	0	0	0	0	0	0	8	8	0	0	0	0
2	16	0	16	0	0	0	0	0	0	0	0	8	0	0	0	0
3	16	0	0	16	8	8	8	8	0	0	0	0	0	0	0	0
4	16	0	0	8	16	0	0	8	0	0	0	0	0	0	0	0
5	16	0	0	8	0	16	8	0	0	0	0	0	0	0	0	0
6	16	0	0	8	0	8	16	0	0	0	0	0	0	0	0	0
7	16	0	0	8	8	0	0	16	0	0	0	0	0	0	0	0
8	16	0	0	0	0	0	0	0	16	0	0	0	0	0	0	0
9	16	0	8	0	0	0	0	0	0	16	0	8	0	0	0	0
a	16	8	0	0	0	0	0	0	0	0	16	8	0	0	0	0
b	16	8	8	0	0	0	0	0	0	8	8	16	0	0	0	0
c	16	0	0	0	0	0	0	0	0	0	0	0	16	0	0	0
d	16	0	0	0	0	0	0	0	0	0	0	0	0	16	0	0
e	16	0	0	0	0	0	0	0	0	0	0	0	0	0	16	0
f	16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	16

$$\boxed{\mathrm{FBCT}(\Delta_i, \nabla_o) = \#\{x \mid S(x) \oplus S(x \oplus \Delta_i) \oplus S(x \oplus \nabla_o) \oplus S(x \oplus \Delta_i \oplus \nabla_o) = 0\}}$$

Properties of the FBCT

Theorem
S is APN if and only if its FBCT verifies $\mathrm{FBCT}(\Delta_i, \nabla_o) = 0 \ \forall\ 1 \leq \Delta_i \neq \nabla_o \leq 2^n - 1$

e.g. S = [1, 3, 6, 5, 2, 4, 7, 0]

8	0	0	0	0	0	0	0
0	0	2	2	0	0	2	2
0	0	0	0	2	2	2	2
0	0	2	2	2	2	0	0
0	2	0	2	0	2	0	2
0	2	2	0	0	2	2	0
0	2	0	2	2	0	2	0
0	2	0	0	2	0	0	2

DDT

8	8	8	8	8	8	8	8
8	8	0	0	0	0	0	0
8	0	8	0	0	0	0	0
8	0	0	8	0	0	0	0
8	0	0	0	8	0	0	0
8	0	0	0	0	8	0	0
8	0	0	0	0	0	8	0
8	0	0	0	0	0	0	8

FBCT

8	8	8	8	8	8	8	8
8	0	2	2	0	0	2	2
8	0	0	0	2	2	2	2
8	0	2	2	2	2	0	0
8	2	0	2	0	2	0	2
8	2	2	0	0	2	2	0
8	2	0	2	2	0	2	0
8	2	0	0	2	0	0	2

BCT

Comparing the BCT and the FBCT

Boomerang uniformity for the **SPN** case:
$\max_{\Delta_i \neq 0, \nabla_o \neq 0} \mathrm{BCT}(\Delta_i, \nabla_o)$

Boomerang uniformity for the **Feistel** case:
$\max_{\Delta_i \neq 0, \nabla_o \neq 0, \Delta_i \neq \nabla_o} \mathrm{FBCT}(\Delta_i, \nabla_o)$

Boomerang uniformity preserved under	BCT	FBCT
Affine equivalence	✓	✓
Extended-affine equivalence	✗	✓
CCZ equivalence	✗	✗
Inversion (if *S* is invertible)	✓	✗

S-box behavior can be different regarding boomerang switches when used in an SPN vs in a Feistel

Switches over more rounds

1-round switch

FBCT, counterpart of the BCT from

Boomerang Connectivity Table: a New Cryptanalysis Tool
Cid, Huang, Peyrin, Sasaki & Song, *EUROCRYPT 2018*

$$\mathrm{FBCT}(\Delta_i, \nabla_o) = \#\{x \in \mathbb{F}_2^n \mid S(x) \oplus S(x \oplus \Delta_i) \oplus S(x \oplus \nabla_o) \oplus S(x \oplus \Delta_i \oplus \nabla_o) = 0\}$$

$$\boxed{2^{-tn} \times \mathrm{FBCT}(\Delta_i, \delta, \nabla_o)}$$

Switches over more rounds

1-round switch

FBCT, counterpart of the BCT from

Boomerang Connectivity Table: a New Cryptanalysis Tool
Cid, Huang, Peyrin, Sasaki & Song, *EUROCRYPT 2018*

2-round switch

FBDT, counterpart of the BDT from[1]

Boomerang switch in multiple rounds.
Wang & Peyrin, *ToSC 2019*

$$\mathrm{FBDT}(\Delta_i, \delta, \nabla_o) = \#\{x \in \mathbb{F}_2^n \mid S(x) \oplus S(x \oplus \Delta_i) \oplus S(x \oplus \nabla_o) \oplus S(x \oplus \Delta_i \oplus \nabla_o) = 0, \\ S(x) \oplus S(x \oplus \Delta_i) = \delta\}$$

$$\boxed{2^{-2tn} \times \sum_{0 \le \delta, \alpha < 2^n} \mathrm{FBDT}(\Delta_i, \delta, \nabla'_o) \times \mathrm{FBDT}(\nabla_o, \alpha, \Delta'_i)}$$

[1] also studied in Boomerang Connectivity Table Revisited. Application to SKINNY and AES
Song, Qin & Hu, *ToSC 2019*

Switches over more rounds

1-round switch

FBCT, counterpart of the BCT from

Boomerang Connectivity Table: a New Cryptanalysis Tool
Cid, Huang, Peyrin, Sasaki & Song, *EUROCRYPT 2018*

2-round switch

FBDT, counterpart of the BDT from

Boomerang switch in multiple rounds.
Wang & Peyrin, *ToSC 2019*

3 rounds and more…

FBET

$$\text{FBET}(\Delta_i, \delta, \nabla_o, \alpha) = \#\{x \in \mathbb{F}_2^n \mid S(x) \oplus S(x \oplus \Delta_i) \oplus S(x \oplus \nabla_o) \oplus S(x \oplus \Delta_i \oplus \nabla_o) = 0, \\ S(x) \oplus S(x \oplus \Delta_i) = \delta, \\ S(x \oplus \Delta_i) \oplus S(x \oplus \Delta_i \oplus \nabla_o) = \alpha\}$$

$$\boxed{2^{-3tn} \sum_{0 \le \delta, \alpha, \delta', \alpha', \delta'', \alpha'' < 2^n} \text{FBET}(\Delta_i, \delta, \nabla_o, \alpha) \times \text{FBET}\Delta_i', \delta', \nabla_o', \alpha') \times \text{FBET}(\Delta_i'', \delta'', \nabla_o'', \alpha'')}$$

Conclusion on the FBCT

- Introduction of the **FBCT**, a new tool that:
  - easily evaluates the probability of a 1-round boomerang switch
  - gives a new criterion when choosing an S-box for a Feistel cipher
- Proposal of a **generic formula** for a switch over many rounds:
  - evaluation is computationally expensive if E_m covers many rounds with many active S-boxes
  - might be preferable to experimentally evaluate it

Conclusion
& Perspectives

General Conclusion

- This thesis explored several aspects of lightweight cryptography, from both design and analysis aspects.
- Many design strategies.
- Finding the right balance between performance/cost & security is hard.
- Third-party analysis is instrumental.
- Such analysis can be improved using automated tools (MILP/CP).

New Directions

- How small can we go ?
- Can we automate everything ?

Bibliography

The Boomerang Attack
Wagner,
FSE 1999

A Practical-time Related-key Attack on the KASUMI Cryptosystem Used in GSM and 3G Telephony
Dunkelman, Keller & Shamir,
CRYPTO 2010

Boomerang Connectivity Table: a New Cryptanalysis Tool
Cid, Huang, Peyrin, Sasaki & Song,
EUROCRYPT 2018

Boomerang Connectivity Table Revisited. Application to SKINNY and AES
Song, Qin & Hu,
ToSC 2019

Spook: Sponge-Based Leakage-Resistant Authenticated Encryption with a Masked Tweakable Block Cipher
Bellizia, Berti, Bronchain, Cassiers,Duval, Guo, Leander, Leurent, Levi, Momin, Pereira, Peters, Standaert, Udvarhelyi & Wiemer,
ToSC 2020

Appendix
Overview of other contributions + additional details

Lilliput-AE

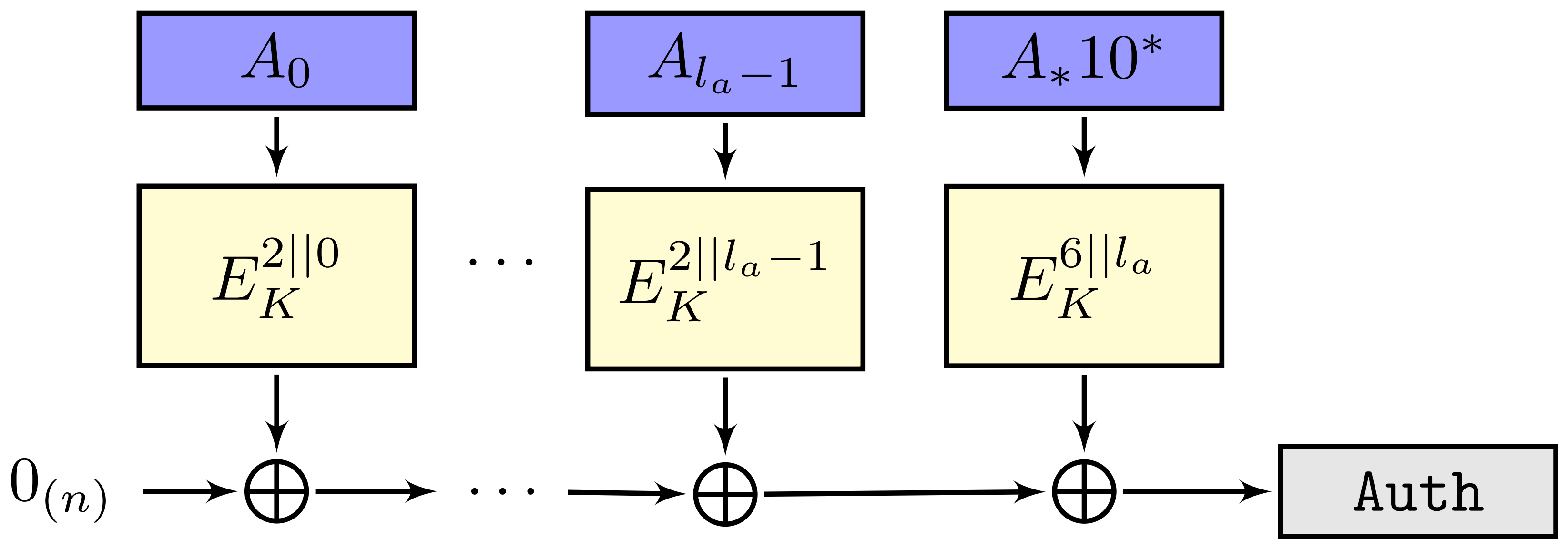
Recommended Parameters

Two Authenticated Encryption modes:

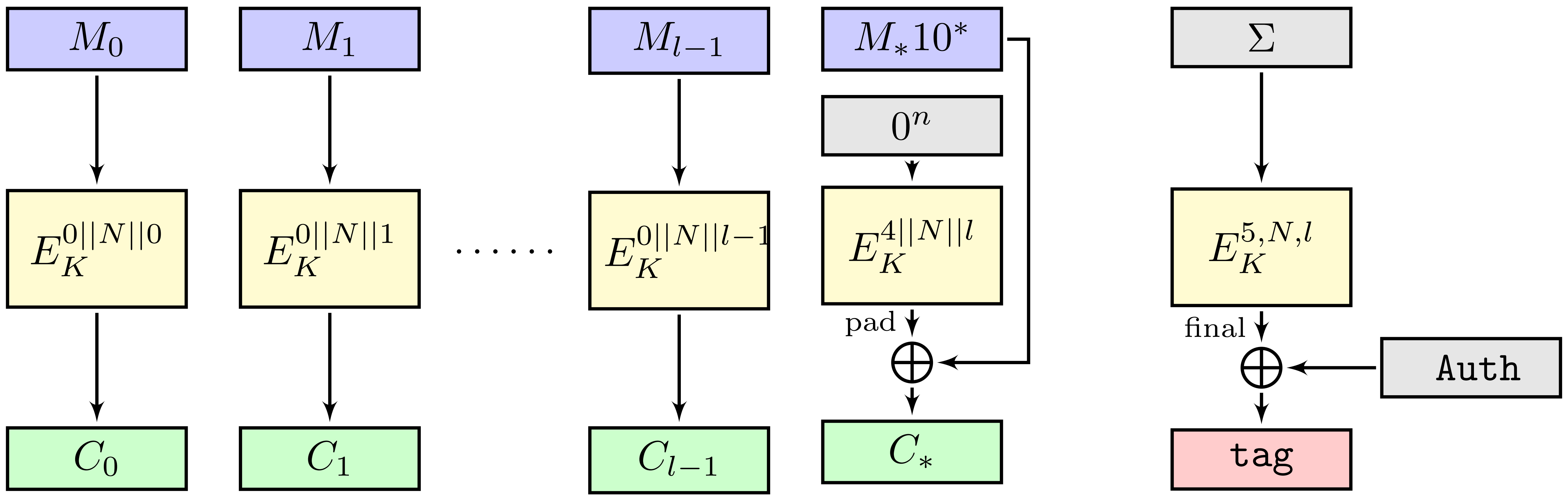
- **Lilliput-I**, nonce-respecting mode **ΘCB3** [Krovetz & Rogaway, 11]
- **Lilliput-II**, nonce-misuse resistant mode **SCT-2** [Peyrin & Seurin, 16]

Name	k	t	n	τ
Lilliput-I-128	128	192	128	120
Lilliput-I-192	192	192	128	120
Lilliput-I-256	256	192	128	120
Lilliput-II-128	**128**	**128**	**128**	**120**
Lilliput-II-192	192	128	128	120
Lilliput-II-256	256	128	128	120

Lilliput-I: Nonce-respecting Mode

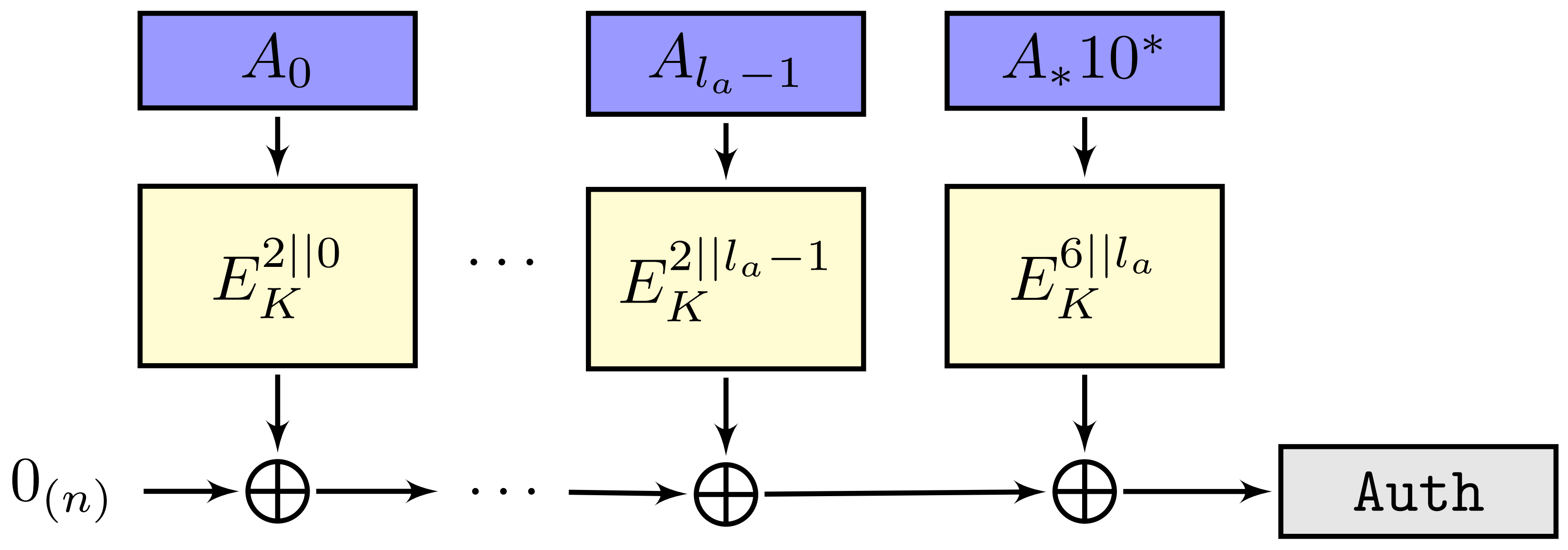


Handling of Associated Data.

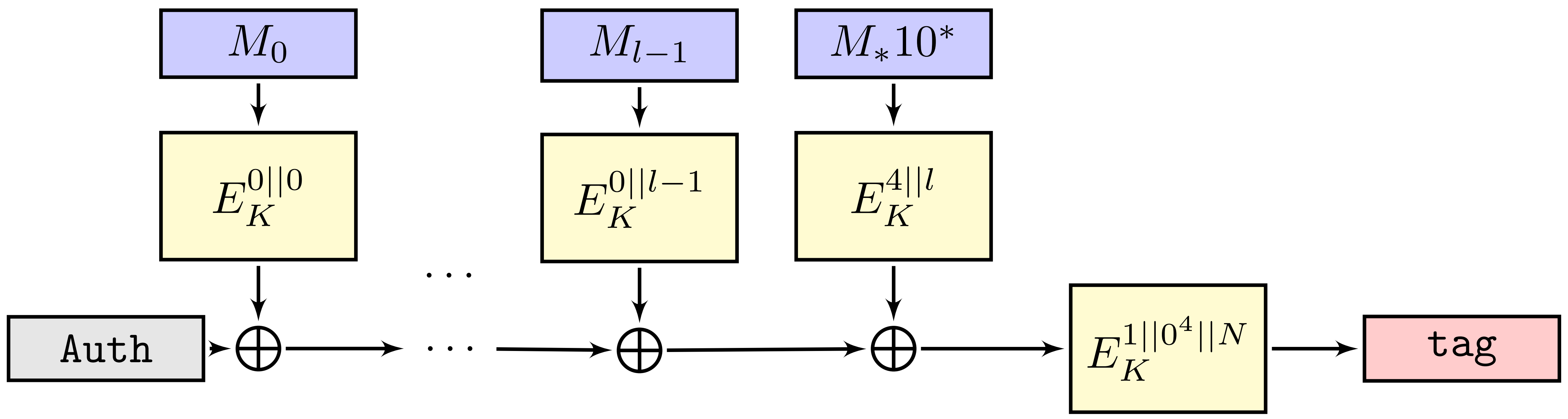


Message processing.

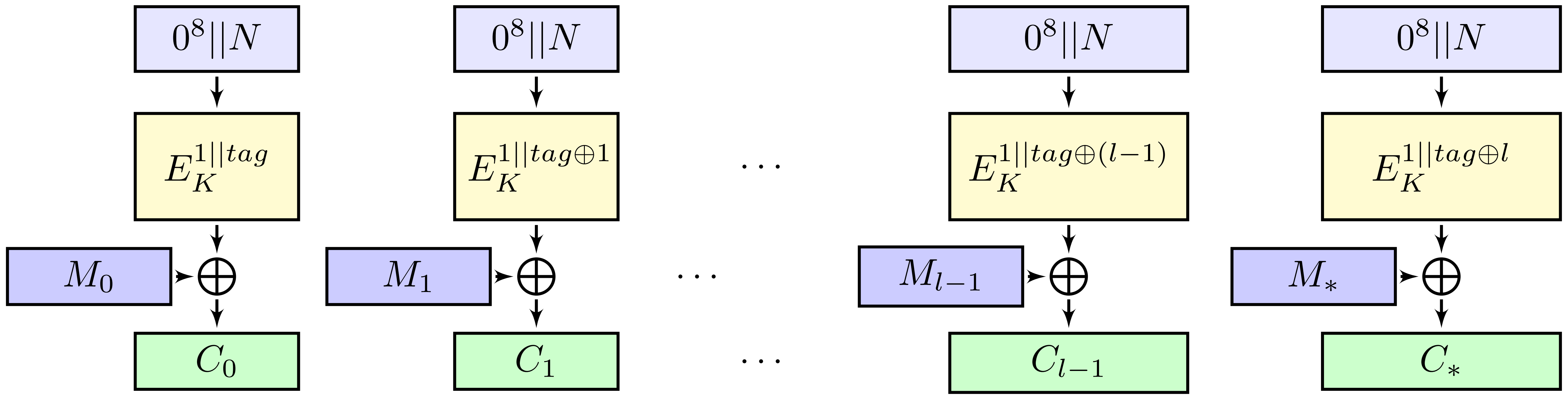
Lilliput-II: Nonce-misuse Resistant Mode



Handling of Associated Data.



Message Processing for Authentication.



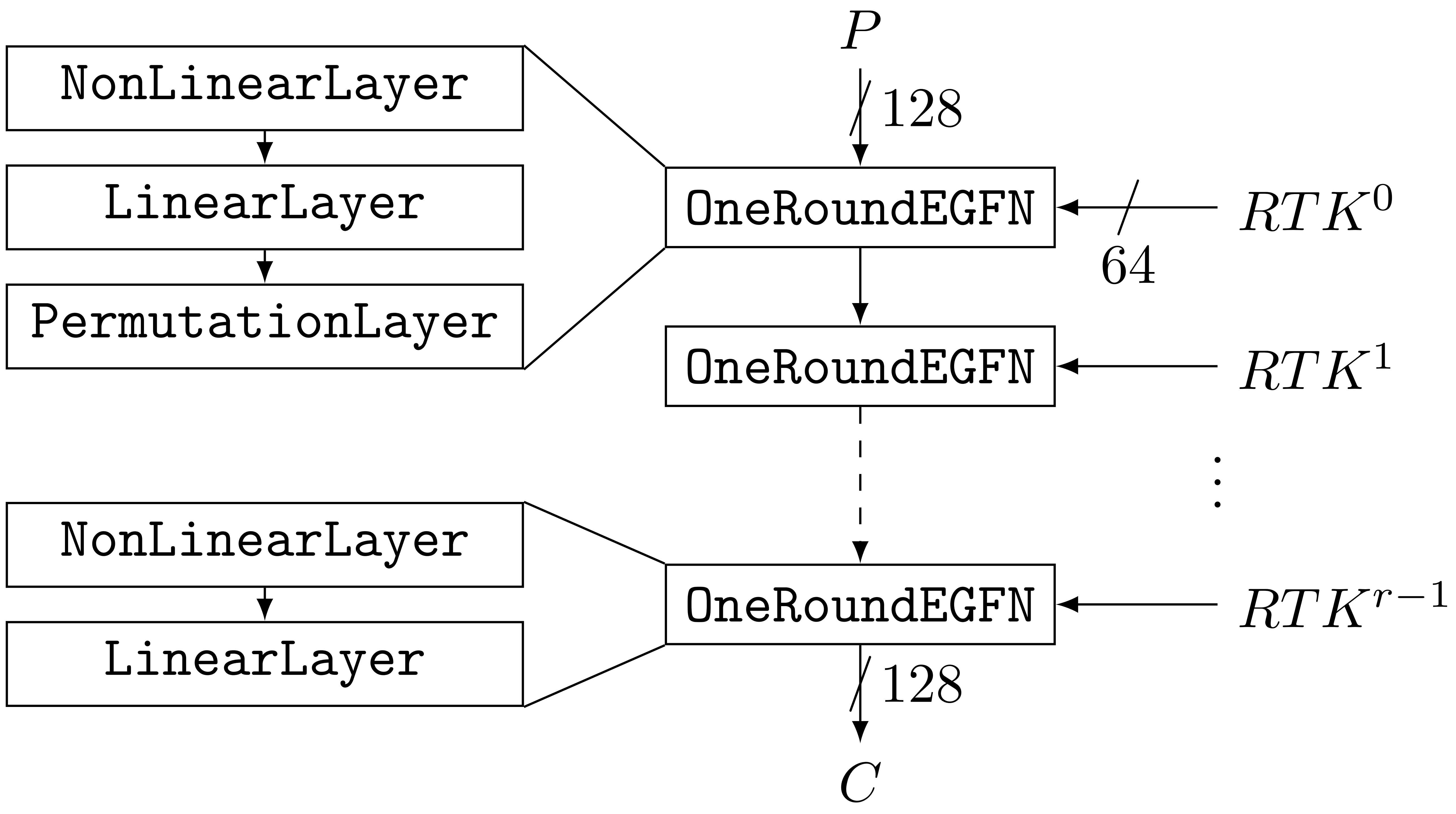
Message Processing for Encryption.

The Lilliput-TBC Tweakable Block Cipher

Based on **Lilliput** [Berger, Francq, Minier &Thomas, 15]

Name	*k*	*t*	*nb rounds*
Lilliput-TBC-I-128	128	192	32
Lilliput-TBC-I-192	192	192	36
Lilliput-TBC-I-256	256	192	42
Lilliput-TBC-II-128	128	128	32
Lilliput-TBC-II-192	192	128	36
Lilliput-TBC-II-256	256	128	42

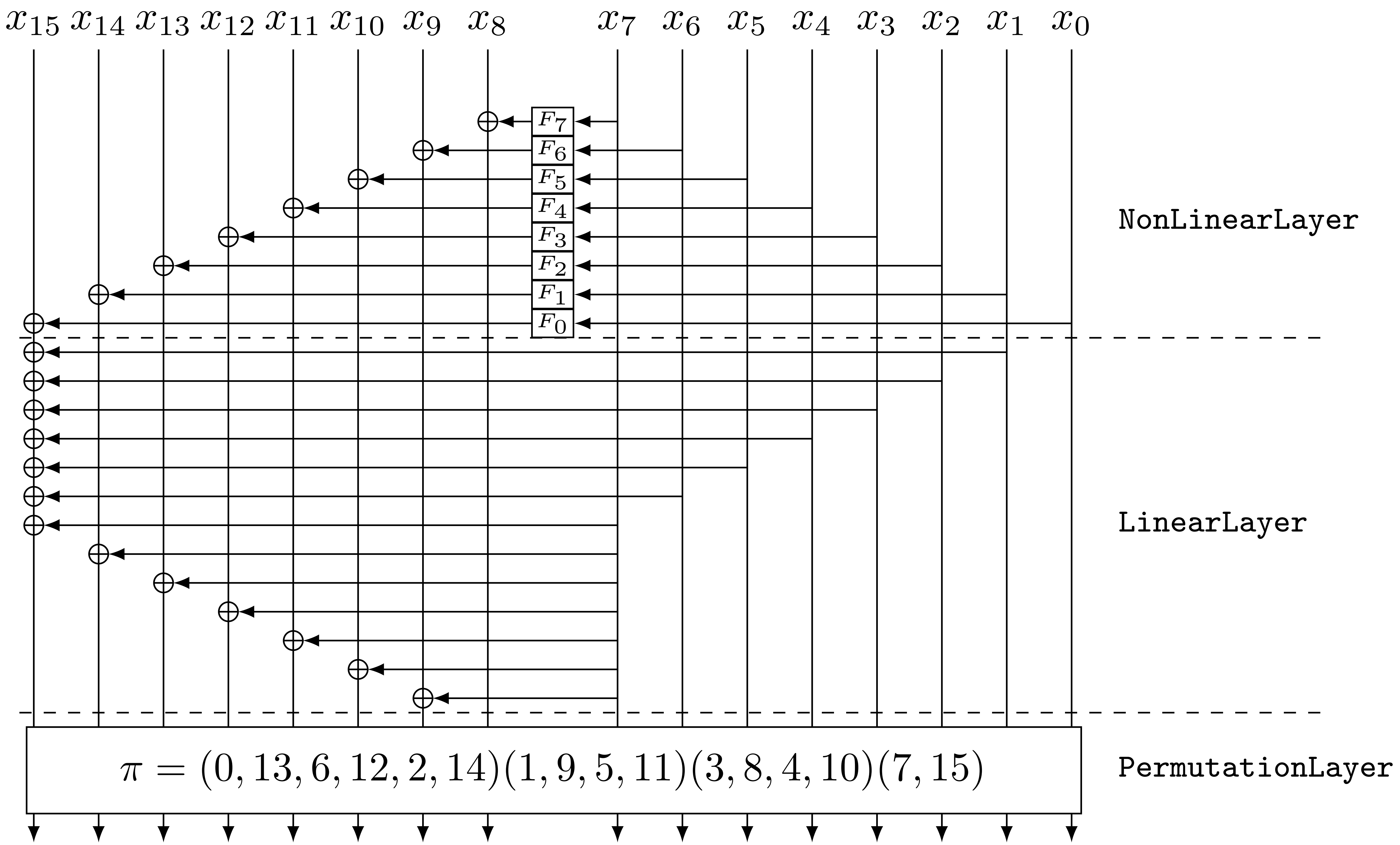
Lilliput-TBC Encryption Process



Decryption analogous to encryption (inverted block permutation layer and reverted subkeys order)

Lilliput-TBC Round Function

Based on **Lilliput** [Berger, Francq, Minier &Thomas, 15]



Lilliput-TBC S-Box

- Differential uniformity $\delta = 8$
- Linearity $L = 64$
- Algebraic degree $deg = 6$
- No fixed point

	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F
00	20	00	B2	85	3B	35	A6	A4	30	E4	6A	2C	FF	59	E2	0E
10	F8	1E	7A	80	15	BD	3E	B1	E8	F3	A2	C2	DA	51	2A	10
20	21	01	23	78	5C	24	27	B5	37	C7	2B	1F	AE	0A	77	5F
30	6F	09	9D	81	04	5A	29	DC	39	9C	05	57	97	74	79	17
40	44	C6	E6	E9	DD	41	F2	8A	54	CA	6E	4A	E1	AD	B6	88
50	1C	98	7E	CE	63	49	3A	5D	0C	EF	F6	34	56	25	2E	D6
60	67	75	55	76	B8	D2	61	D9	71	8B	CD	0B	72	6C	31	4B
70	69	FD	7B	6D	60	3C	2F	62	3F	22	73	13	C9	82	7F	53
80	32	12	A0	7C	02	87	84	86	93	4E	68	46	8D	C3	DB	EC
90	9B	B7	89	92	A7	BE	3D	D8	EA	50	91	F1	33	38	E0	A9
A0	A3	83	A1	1B	CF	06	95	07	9E	ED	B9	F5	4C	C0	F4	2D
B0	16	FA	B4	03	26	B3	90	4F	AB	65	FC	FE	14	F7	E3	94
C0	EE	AC	8C	1A	DE	CB	28	40	7D	C8	C4	48	6B	DF	A5	52
D0	E5	FB	D7	64	F9	F0	D3	5E	66	96	8F	1D	45	36	CC	C5
E0	4D	9F	BF	0F	D1	08	EB	43	42	19	E7	99	A8	8E	58	C1
F0	9A	D4	18	47	AA	AF	BC	5B	D5	11	D0	B0	70	BB	0D	BA

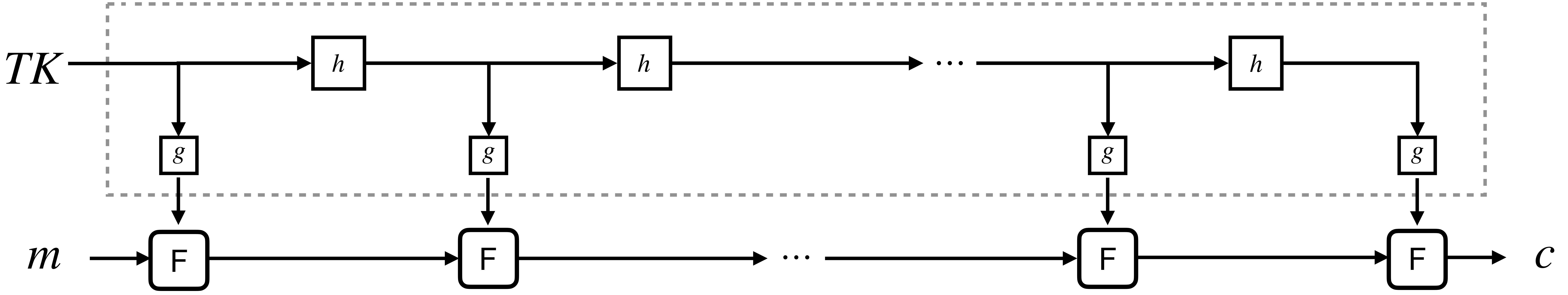
Tweakey Schedule: Parameters

- An adapted version of the TWEAKEY framework: the key and the tweak inputs are handled almost the same way
- The tweakey schedule produces the 64-bit subtweakeys RTK^0 to RTK^{r-1} from the master key K and the tweak T divided into $p = (t + k)/64$ lanes that we denote TK_j^i

Name	*k*	*t*	*p*	*nb rounds*
Lilliput-TBC-I-128	128	192	5	32
Lilliput-TBC-I-192	192	192	6	36
Lilliput-TBC-I-256	256	192	7	42
Lilliput-TBC-II-128	128	128	4	32
Lilliput-TBC-II-192	192	128	5	36
Lilliput-TBC-II-256	256	128	6	42

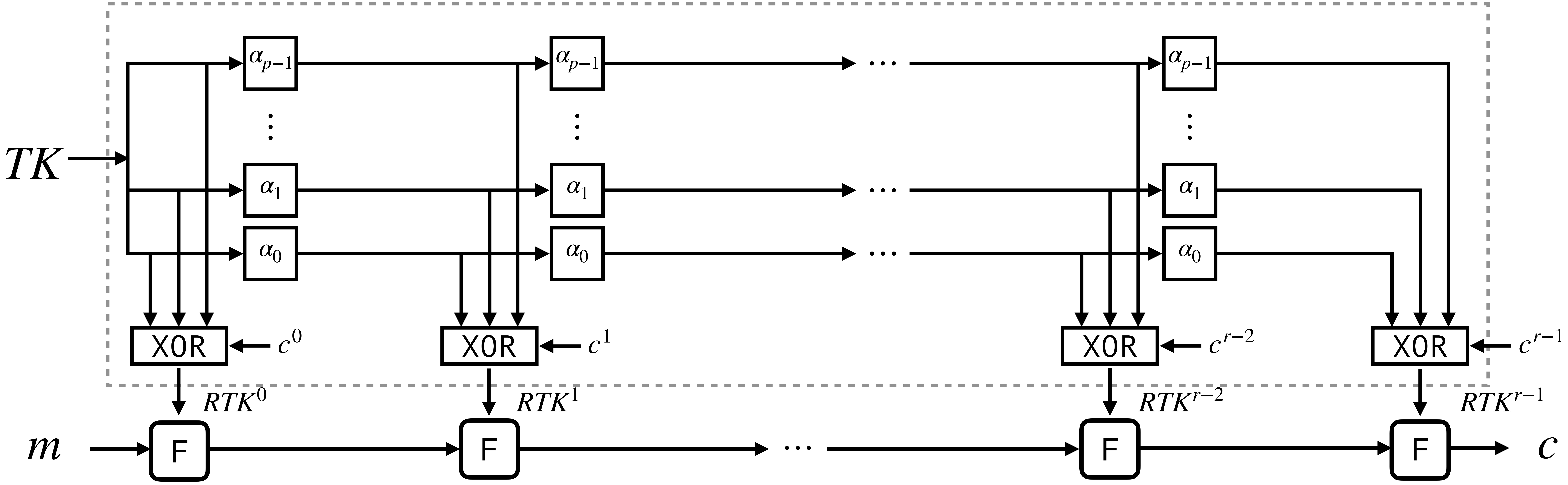
Lilliput-TBC Tweakey Schedule

TWEAKEY framework [Jean, Nikolić & Peyrin, 2014]



Lilliput-TBC Tweakey Schedule

$\alpha_0, \cdots, \alpha_{p-1}$ produced by word-ring-LFSRs to improve software and hardware performances



Design Rationale

- Based on **Lilliput**, a well studied block cipher without any known weaknesses

- Underlying **EGFN** structure chosen for its good diffusion properties
  - Permutation layer chosen to maximize the resistance against linear/differential cryptanalysis

- Tweakey schedule based on the TWEAKEY construction ensuring that the number of cancellations on r+1 subtweakeys is at most (p-1)

Design Rationale: the S-Box

- Chosen for its good cryptographic properties (resistance against linear/differential cryptanalysis, high algebraic degree, etc.)
- Built from 4-bit S-boxes
- Chosen for its low cost in terms of hardware implementation and of threshold implementation

	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F
00	20	00	B2	85	3B	35	A6	A4	30	E4	6A	2C	FF	59	E2	0E
10	F8	1E	7A	80	15	BD	3E	B1	E8	F3	A2	C2	DA	51	2A	10
20	21	01	23	78	5C	24	27	B5	37	C7	2B	1F	AE	0A	77	5F
30	6F	09	9D	81	04	5A	29	DC	39	9C	05	57	97	74	79	17
40	44	C6	E6	E9	DD	41	F2	8A	54	CA	6E	4A	E1	AD	B6	88
50	1C	98	7E	CE	63	49	3A	5D	0C	EF	F6	34	56	25	2E	D6
60	67	75	55	76	B8	D2	61	D9	71	8B	CD	0B	72	6C	31	4B
70	69	FD	7B	6D	60	3C	2F	62	3F	22	73	13	C9	82	7F	53
80	32	12	A0	7C	02	87	84	86	93	4E	68	46	8D	C3	DB	EC
90	9B	B7	89	92	A7	BE	3D	D8	EA	50	91	F1	33	38	E0	A9
A0	A3	83	A1	1B	CF	06	95	07	9E	ED	B9	F5	4C	C0	F4	2D
B0	16	FA	B4	03	26	B3	90	4F	AB	65	FC	FE	14	F7	E3	94
C0	EE	AC	8C	1A	DE	CB	28	40	7D	C8	C4	48	6B	DF	A5	52
D0	E5	FB	D7	64	F9	F0	D3	5E	66	96	8F	1D	45	36	CC	C5
E0	4D	9F	BF	0F	D1	08	EB	43	42	19	E7	99	A8	8E	58	C1
F0	9A	D4	18	47	AA	AF	BC	5B	D5	11	D0	B0	70	BB	0D	BA

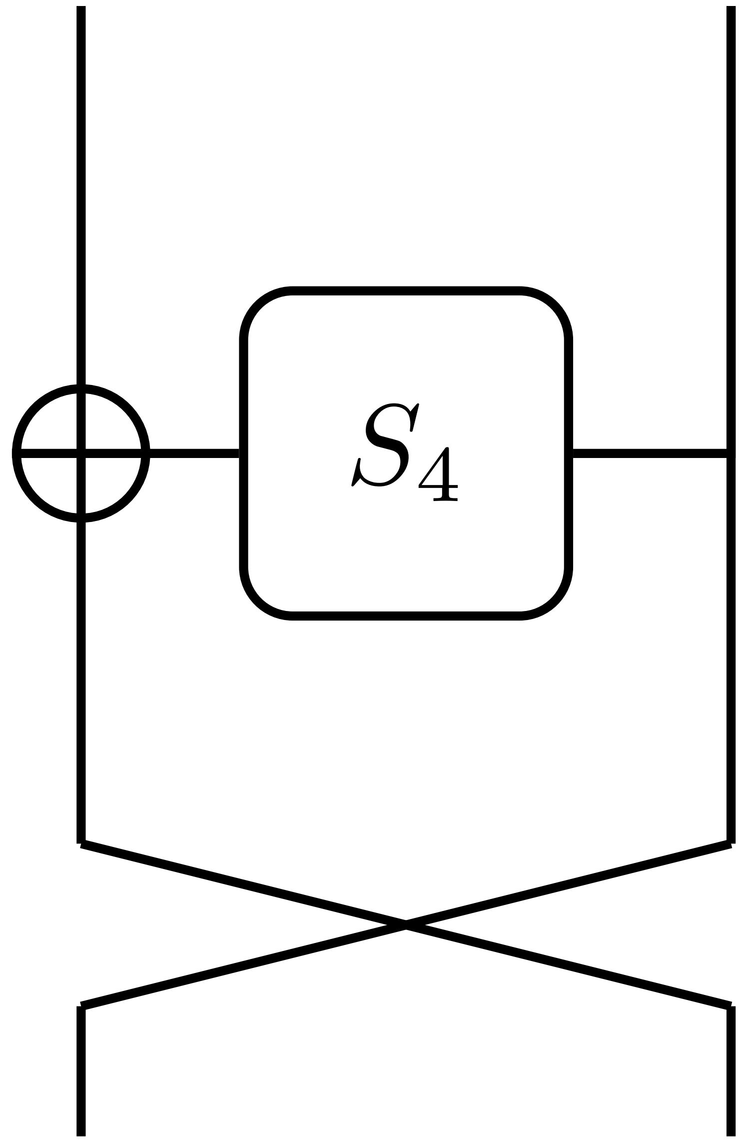
Design Rationale: the S-Box

- Chosen for its good cryptographic properties (resistance against linear/differential cryptanalysis, high algebraic degree, etc.)

- Built from 4-bit S-boxes

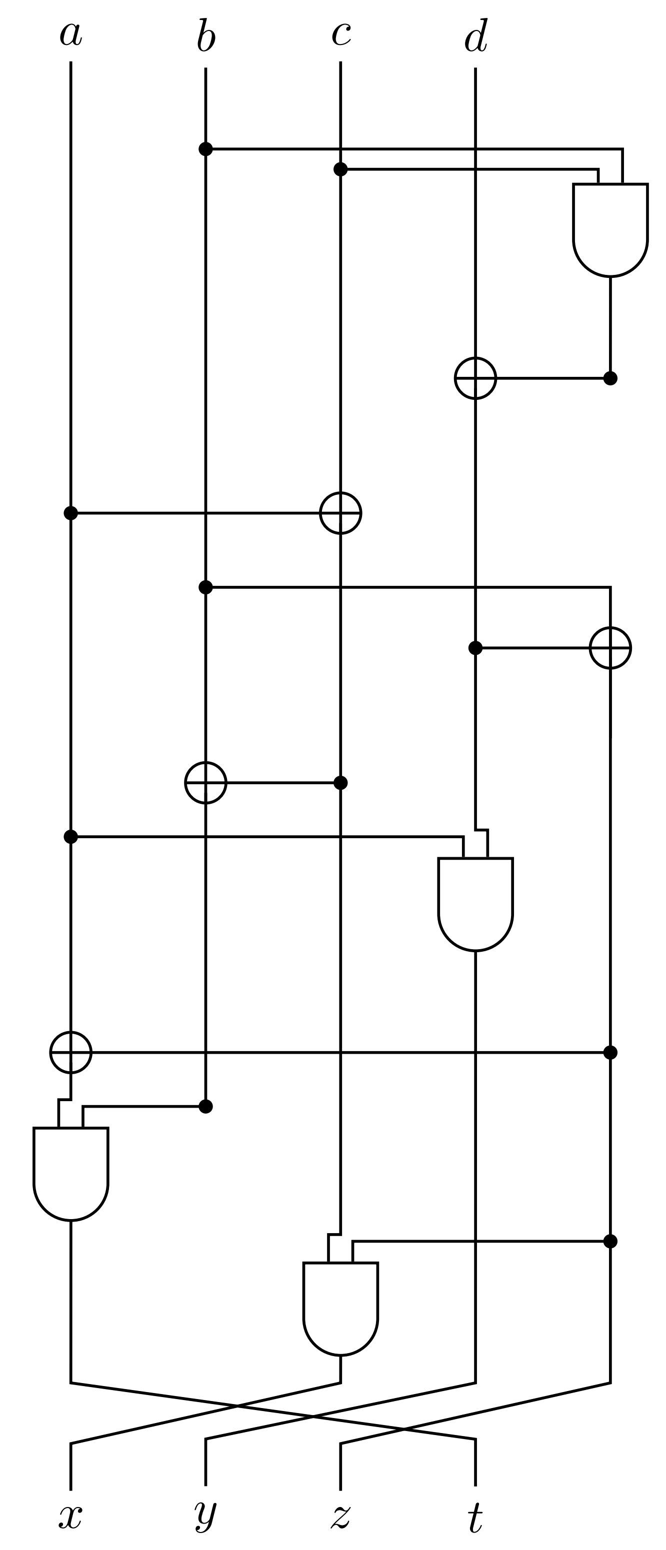
    - Based on a 3-round Feistel scheme with two APN functions and a 4-bit S-box in the middle round

- Chosen for its low cost in terms of hardware implementation and of threshold implementation

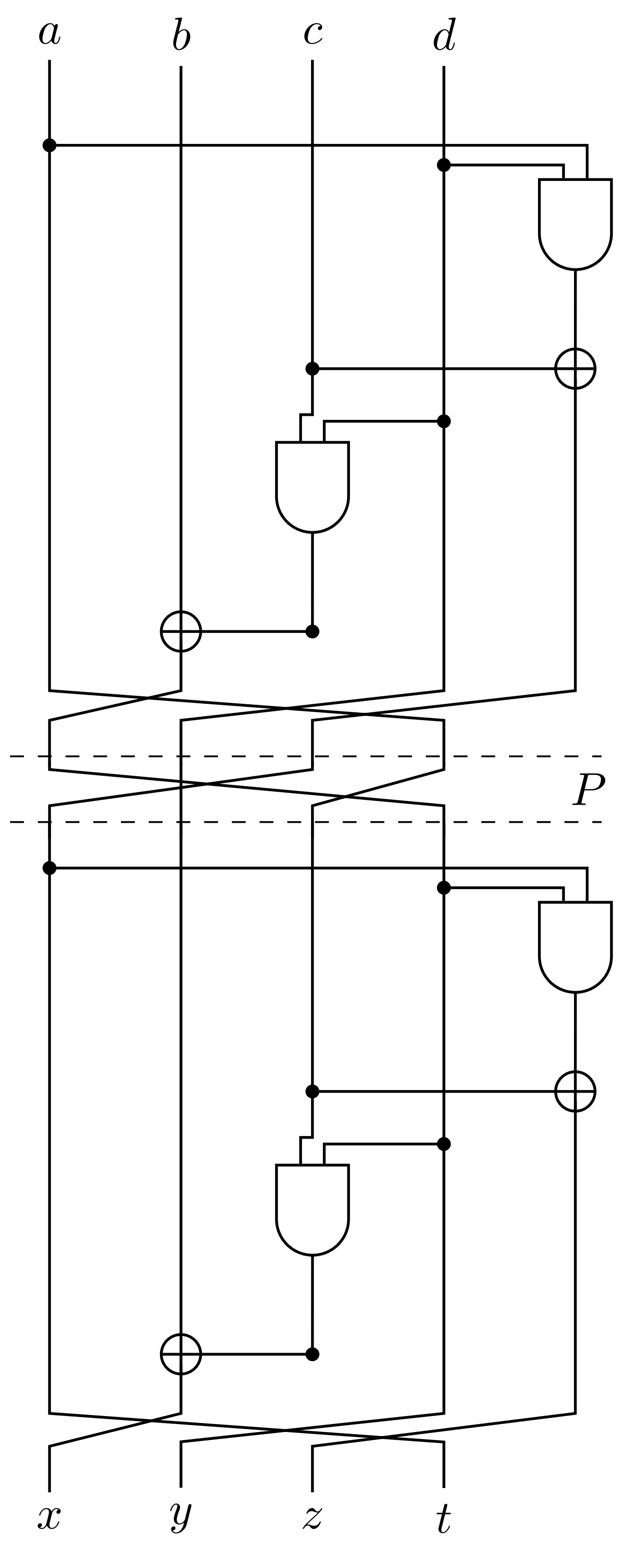


Design Rationale: the S-Box

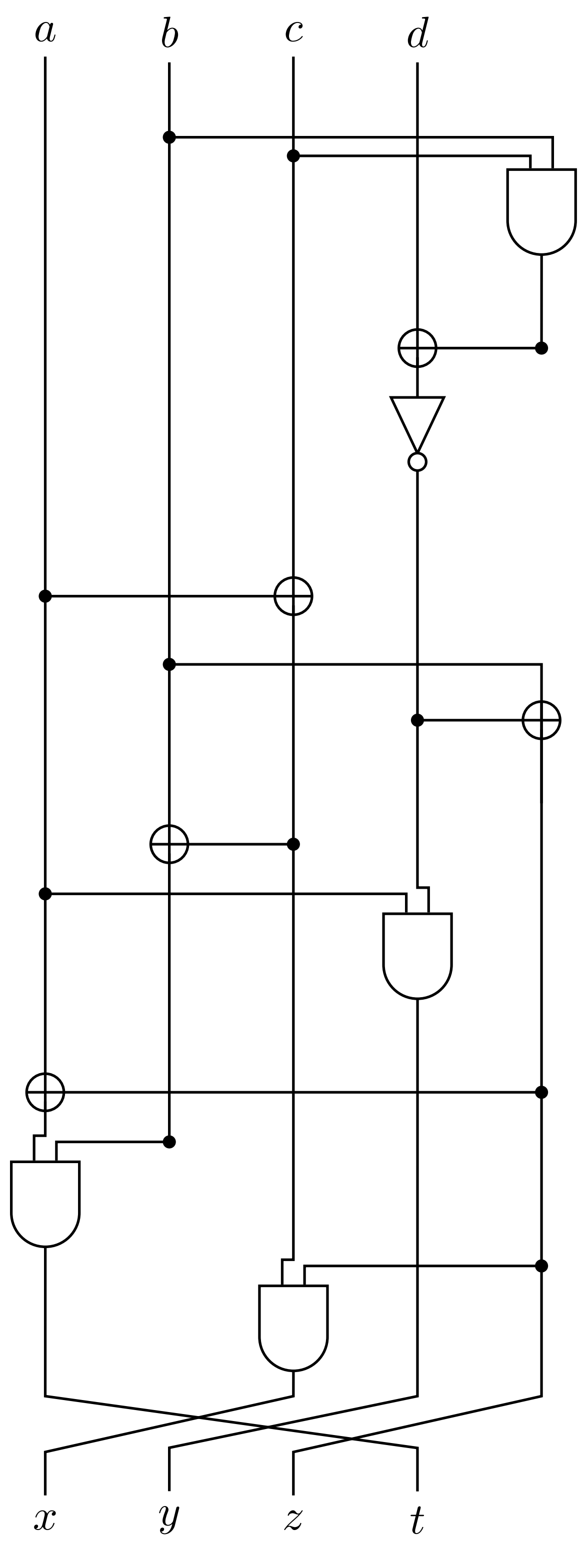
- Chosen for its good cryptographic properties (resistance against linear/differential cryptanalysis, high algebraic degree, etc.)
- Built from 4-bit S-boxes
    - Based on a 3-round Feistel scheme with two APN functions and a 4-bit S-box in the middle round
- Chosen for its low cost in terms of hardware implementation and of threshold implementation
    - number of TI shares limited by using quadratic bijections $S = F \circ G$, with affine functions A_1, A_2 such that $F = A_1 \circ Q \circ A_2$



$S_4^1 = F \circ G$
020b300a1e06a452



$\bar{S}_4^2 = Q \circ P \circ Q$
081f4c792b36e5d



$S_4^3 = F \circ (\bigoplus 1) \circ G$
20b003a0e1604a25

Security Analysis

	STKM			RTMK				Nb rounds (r)	Security Margin (in rounds)
	Diff.	Lin.	Struct.	Diff.	Lin.	RTKB	Struct.		
Lilliput-TBC-I-128	21	24	18	27	24	28	23	32	**4**
Lilliput-TBC-I-192	25	31	18	32	31	32	24	36	**4**
Lilliput-TBC-I-256	32	38	18	40	38	36	25	42	**2**
Lilliput-TBC-II-128	21	24	18	26	24	26	22	32	**6**
Lilliput-TBC-II-192	25	31	18	31	31	30	23	36	**5**
Lilliput-TBC-II-256	32	38	18	39	38	34	24	42	**3**

Security Evaluation summary (“paranoid” case). STKM means “Single Tweakey Model”, RTKM means “Related Tweakey Model” and RTKB means “Related Tweakey Boomerang attack”.

Software Implementations (1/4)

	Version	`CFLAGS`	Code size (B)	RAM (B)	Execution time (cycles)
ACORN-128	`8bitfast`	`-O3`	3700	263	287991
Ascon-128	`ref`	`-O3`	6140	268	191049
Ascon-128a	`ref`	`-O3`	6832	300	163320
Lilliput-I-128	`ref`	`-O3`	8188	563	174332
Lilliput-I-192	`ref`	`-O3`	8318	611	225200
Lilliput-I-256	`ref`	`-O3`	8466	675	298223
Lilliput-II-128	`ref`	`-O3`	7500	544	178436
Lilliput-II-192	`ref`	`-O3`	7478	592	260600
Lilliput-II-256	`ref`	`-O3`	7600	656	349372
ACORN-128	`8bitfast`	`-Os`	2850	240	335934
Ascon-128	`ref`	`-Os`	4322	323	254913
Ascon-128a	`ref`	`-Os`	4340	339	216080
Lilliput-I-128	`ref`	`-Os`	3252	523	221161
Lilliput-I-192	`ref`	`-Os`	3394	571	278344
Lilliput-I-256	`ref`	`-Os`	3564	637	362194
Lilliput-II-128	`ref`	`-Os`	3252	493	259277
Lilliput-II-192	`ref`	`-Os`	3360	541	328421
Lilliput-II-256	`ref`	`-Os`	3492	605	429541

Performance results on AVR ATmega128.

Software Implementations (2/4)

	Version	`CFLAGS`	Code size (B)	RAM (B)	Execution time (cycles)
ACORN-128	`8bitfast`	`-O3`	3276	274	391983
Ascon-128	`ref`	`-O3`	8358	290	544075
Ascon-128a	`ref`	`-O3`	8620	306	457998
Lilliput-I-128	`ref`	`-O3`	8300	624	153294
Lilliput-I-192	`ref`	`-O3`	8494	672	199212
Lilliput-I-256	`ref`	`-O3`	8720	738	268425
Lilliput-II-128	`ref`	`-O3`	6336	592	172179
Lilliput-II-192	`ref`	`-O3`	6406	644	227943
Lilliput-II-256	`ref`	`-O3`	6600	708	307751
ACORN-128	`8bitfast`	`-Os`	2326	218	381698
Ascon-128	`ref`	`-Os`	3686	372	567110
Ascon-128a	`ref`	`-Os`	3672	382	475176
Lilliput-I-128	`ref`	`-Os`	2582	546	263997
Lilliput-I-192	`ref`	`-Os`	2712	594	333411
Lilliput-I-256	`ref`	`-Os`	2874	660	436140
Lilliput-II-128	`ref`	`-Os`	2574	514	299282
Lilliput-II-192	`ref`	`-Os`	2660	564	384122
Lilliput-II-256	`ref`	`-Os`	2790	628	506170

Performance results on MSP430F1611.

Software Implementations (3/4)

	Version	`CFLAGS`	Code size (B)	RAM (B)	Execution time (cycles)
ACORN-128	`8bitfast`	`-O3`	2568	472	158059
Ascon-128	`ref`	`-O3`	4080	600	32350
Ascon-128a	`ref`	`-O3`	4424	608	27683
Lilliput-I-128	`ref`	`-O3`	6400	748	104988
Lilliput-I-192	`ref`	`-O3`	6484	796	132691
Lilliput-I-256	`ref`	`-O3`	6580	860	175955
Lilliput-II-128	`ref`	`-O3`	5336	724	114004
Lilliput-II-192	`ref`	`-O3`	5220	772	157405
Lilliput-II-256	`ref`	`-O3`	5304	836	206440
ACORN-128	`8bitfast`	`-Os`	1584	320	166370
Ascon-128	`ref`	`-Os`	1426	472	49636
Ascon-128a	`ref`	`-Os`	1408	480	41113
Lilliput-I-128	`ref`	`-Os`	1800	584	197463
Lilliput-I-192	`ref`	`-Os`	1874	632	238539
Lilliput-I-256	`ref`	`-Os`	1958	696	289026
Lilliput-II-128	`ref`	`-Os`	1854	552	212443
Lilliput-II-192	`ref`	`-Os`	1908	600	318290
Lilliput-II-256	`ref`	`-Os`	1980	664	340500

Performance results on ARM Cortex-M3.

Software Implementations (4/4)

	Version	`CFLAGS`	Code size (B)	RAM (B)	Execution time (cycles)
ACORN-128	`8bitfast`	`-O3`	3592	2048	19795
Ascon-128	`ref`	`-O3`	2236	2048	6929
Ascon-128a	`ref`	`-O3`	2102	2048	6538
Lilliput-I-128	`ref`	`-O3`	8578	2048	12248
Lilliput-I-192	`ref`	`-O3`	8756	2056	15313
Lilliput-I-256	`ref`	`-O3`	8979	2064	19688
Lilliput-II-128	`ref`	`-O3`	7421	2048	13584
Lilliput-II-192	`ref`	`-O3`	7583	2056	17350
Lilliput-II-256	`ref`	`-O3`	7761	2064	22556
ACORN-128	`8bitfast`	`-Os`	2409	2048	31612
Ascon-128	`ref`	`-Os`	1486	2048	3900
Ascon-128a	`ref`	`-Os`	1466	2048	3587
Lilliput-I-128	`ref`	`-Os`	2872	2048	19182
Lilliput-I-192	`ref`	`-Os`	3009	2056	22483
Lilliput-I-256	`ref`	`-Os`	3142	2064	28780
Lilliput-II-128	`ref`	`-Os`	2850	2048	21905
Lilliput-II-192	`ref`	`-Os`	2932	2056	27267
Lilliput-II-256	`ref`	`-Os`	3060	2064	33567

Performance results on PC.

Hardware Implementations: Estimations

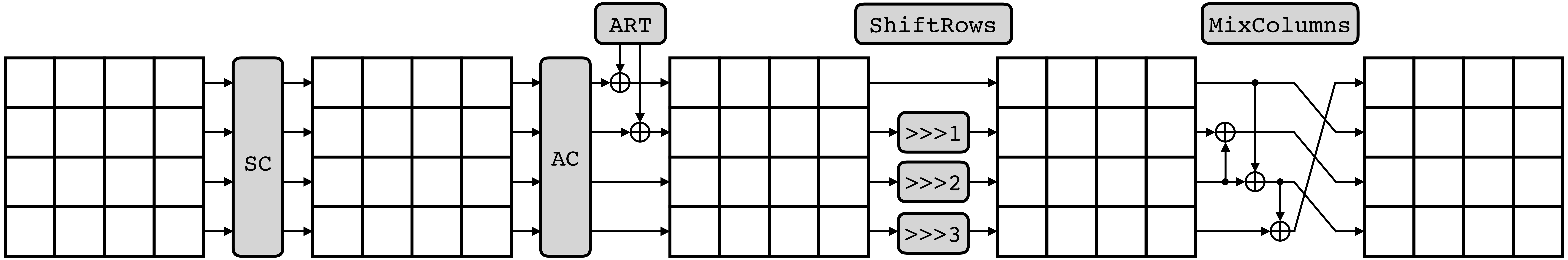
Nb. Lanes	Registers	Round Function	Tweakey Schedule	Total	Relative Perf.
4	384	8 SBoxes + 29 x 8 XORs	176 XORs	4057 GEs	1
5	448	8 SBoxes + 29 x 8 XORs	200 XORs	4230 GEs	1.04
6	512	8 SBoxes + 29 x 8 XORs	256 XORs	4721 GEs	1.16
7	576	8 SBoxes + 29 x 8 XORs	354 XORs	4983 GEs	1.22

Differential Cryptanalysis of Skinny

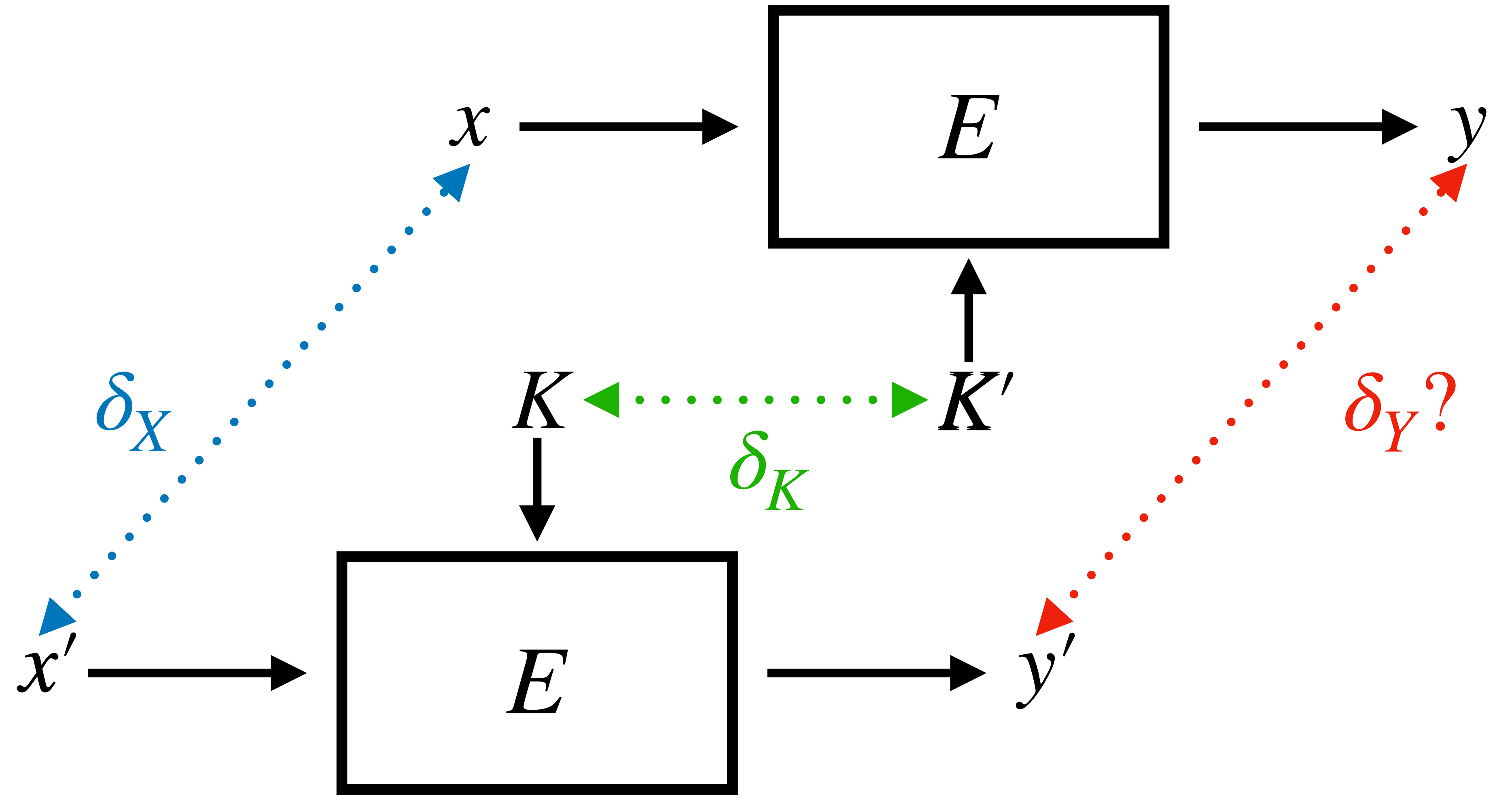
Skinny

[Beirle *et al*. 2016]

- AES-like lightweight tweakeable block cipher
- State size n = 64 / 128 bits
- Tweakey size = n / 2n / 3n
- From 32 to 56 rounds



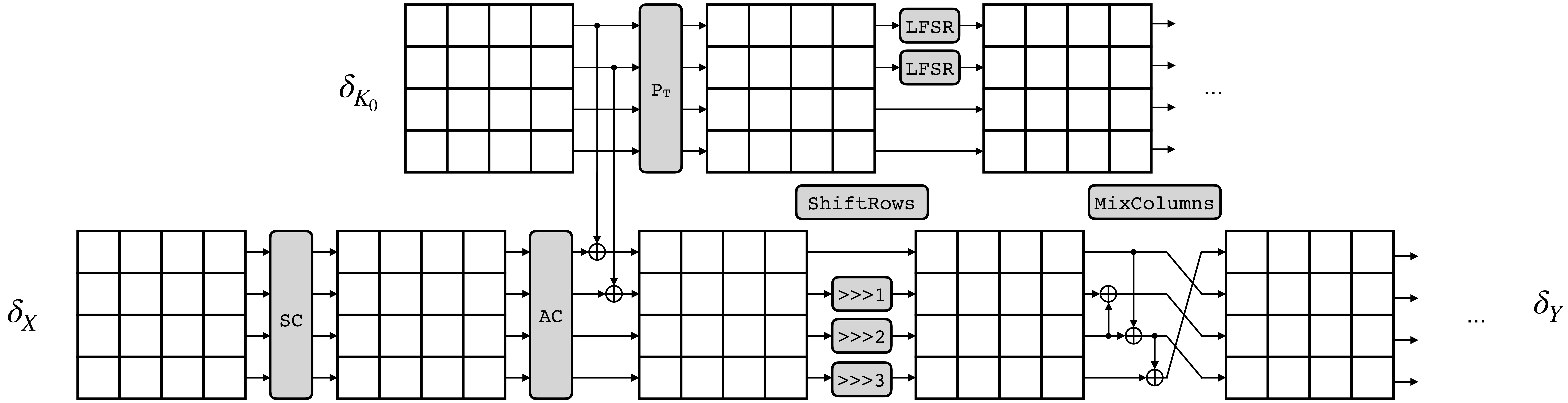
Related-Key Differential Analysis



Differences between plaintexts **and keys**

E is **weak** if there exists a differential $\exists \delta_X, \delta_K,$ and δ_Y such that $\Pr[\delta Y \mid \delta X, \delta K] \gg 2^{-|K|}$.

Related-Key Analysis of Skinny



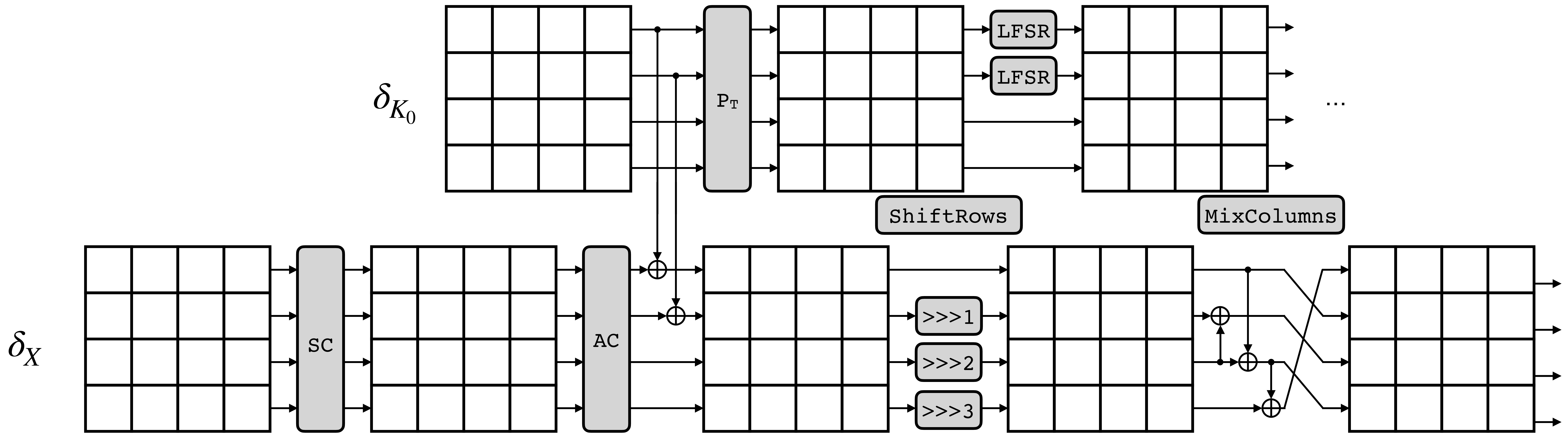
Goal: find $\delta_X, \delta_{K_0},$ and δ_Y that maximizes $\Pr[\delta Y \mid \delta X, \delta K_0]$

Two-step Solving Process

[Biryukov *et al.*, 10][Fouque *et al.*, 13]

Step 1: Abstract differential bytes $\delta B = B \oplus B'$ to booleans ΔB

Step 2: Concretize booleans to differential bytes



Two-step Solving Process

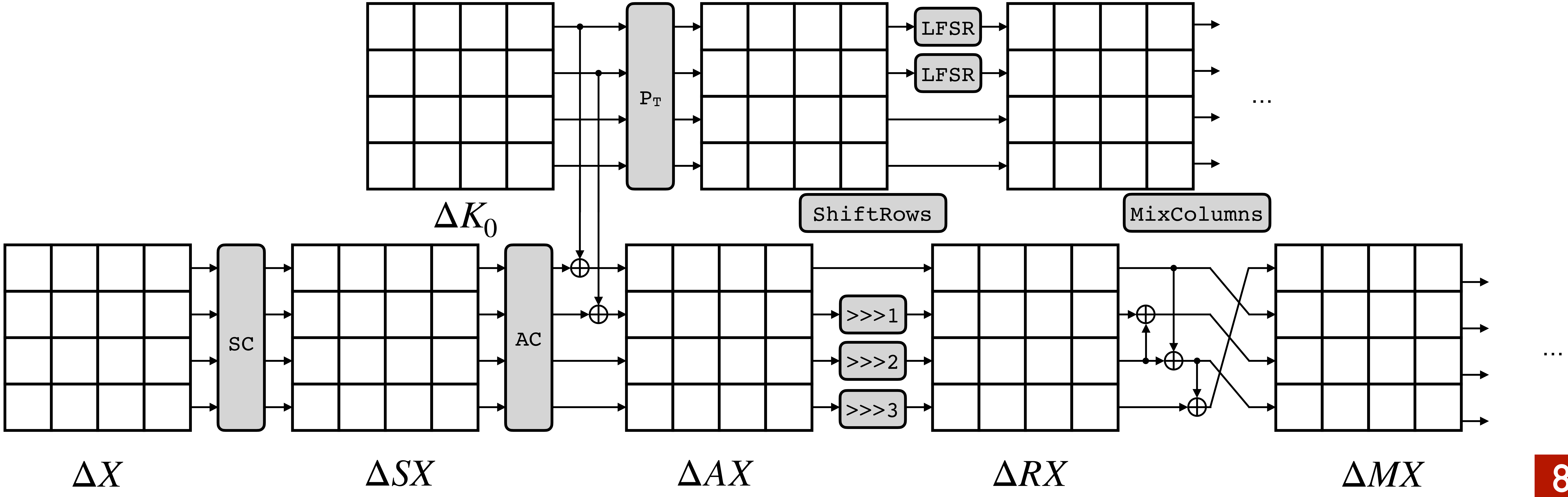
[Biryukov *et al.*, 10][Fouque *et al.*, 13]

Step 1: Abstract differential bytes $\delta B = B \oplus B'$ to booleans ΔB

For each differential byte δB: $\Delta B = 0$ if $\delta B = 0$; $\Delta B = 1$ otherwise

Minimize number of active S-Boxes (ΔSX = 1)

Step 2: Concretize booleans to differential bytes



Two-step Solving Process

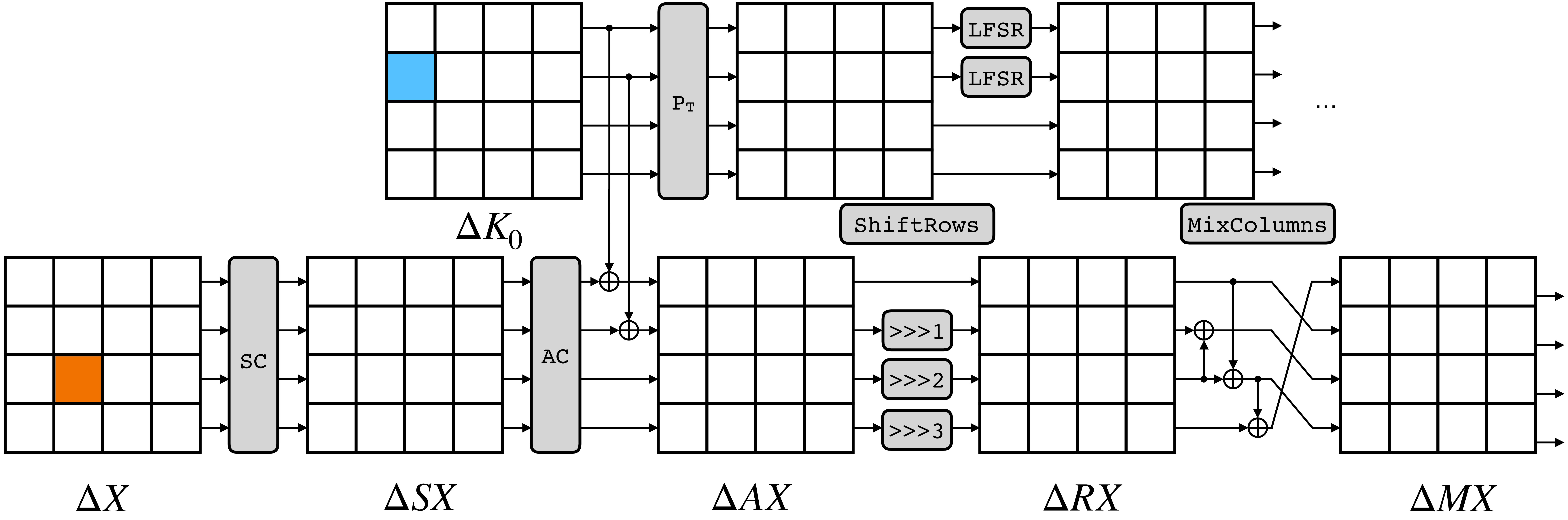
[Biryukov *et al.*, 10][Fouque *et al.*, 13]

Step 1: Abstract differential bytes $\delta B = B \oplus B'$ to booleans ΔB

For each differential byte δB: $\Delta B = 0$ if $\delta B = 0$; $\Delta B = 1$ otherwise

Minimize number of active S-Boxes (ΔSX = 1)

Step 2: Concretize booleans to differential bytes

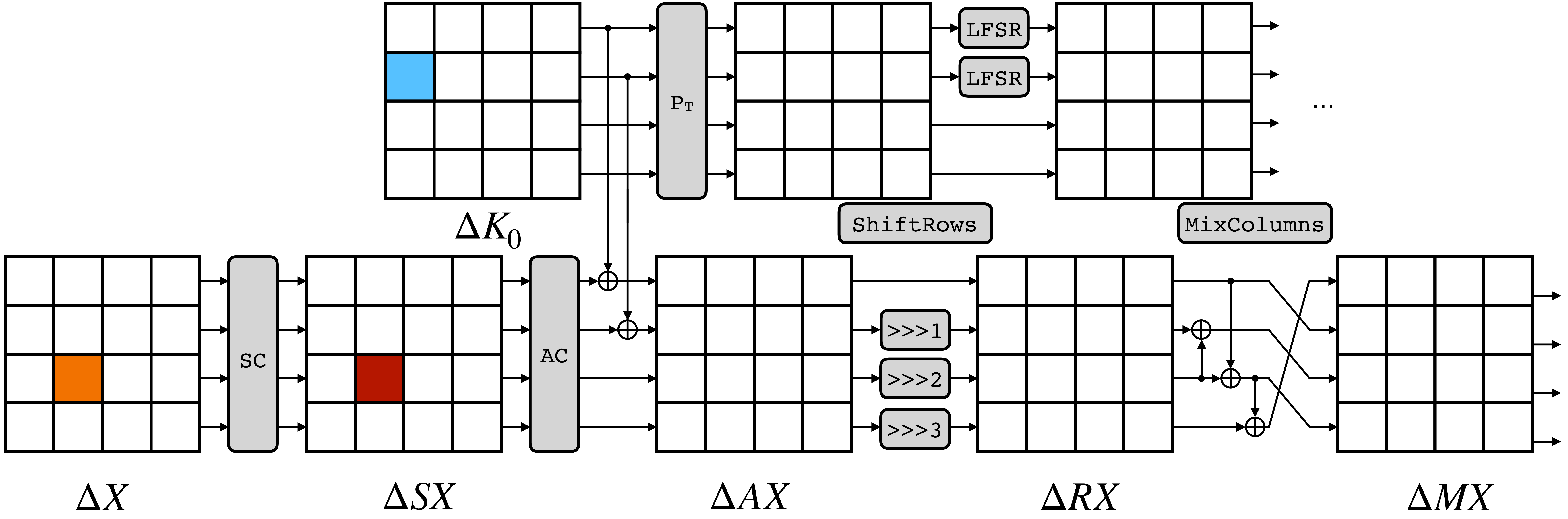


Two-step Solving Process

[Biryukov *et al.*, 10][Fouque *et al.*, 13]

Step 1: Abstract differential bytes $\delta B = B \oplus B'$ to booleans ΔB
For each differential byte δB: $\Delta B = 0$ if $\delta B = 0$; $\Delta B = 1$ otherwise
Minimize number of active S-Boxes (ΔSX = 1)

Step 2: Concretize booleans to differential bytes



Two-step Solving Process

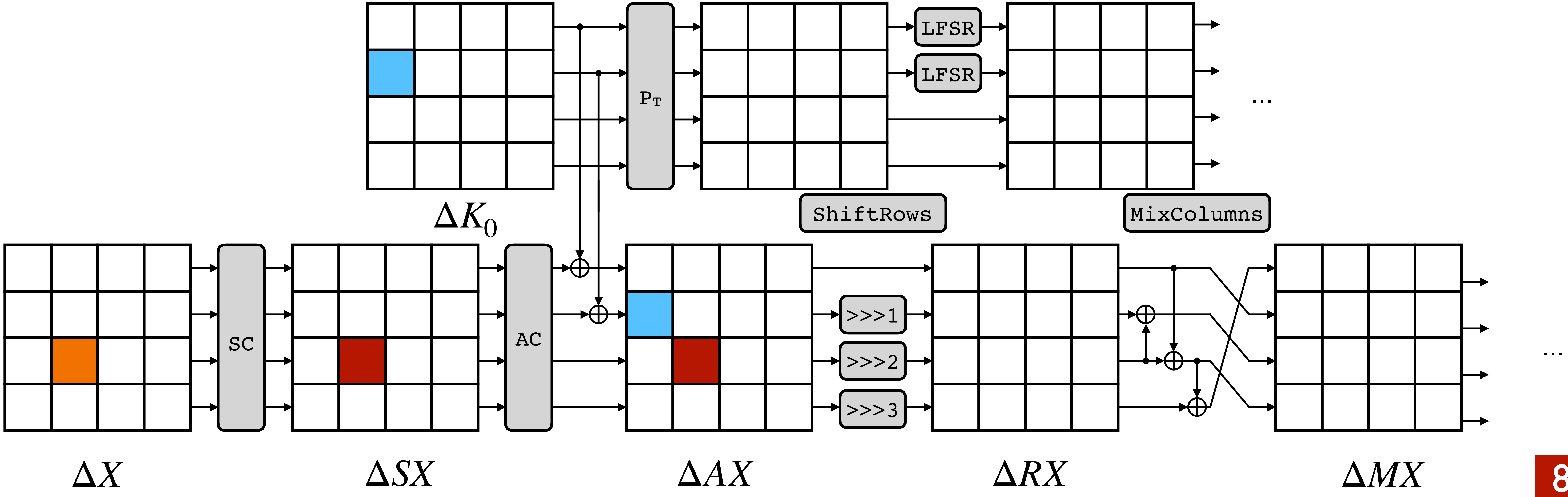
[Biryukov *et al.*, 10][Fouque *et al.*, 13]

Step 1: Abstract differential bytes $\delta B = B \oplus B'$ to booleans ΔB

For each differential byte δB: $\Delta B = 0$ if $\delta B = 0$; $\Delta B = 1$ otherwise

Minimize number of active S-Boxes (ΔSX = 1)

Step 2: Concretize booleans to differential bytes

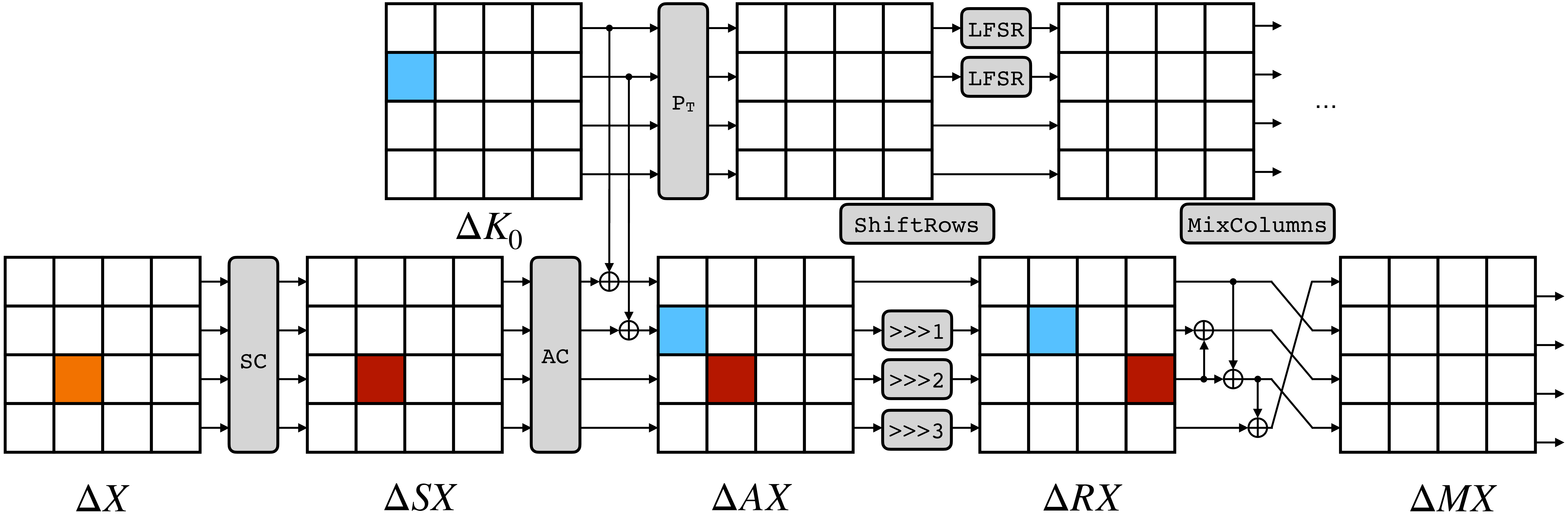


Two-step Solving Process

[Biryukov *et al.*, 10][Fouque *et al.*, 13]

Step 1: Abstract differential bytes $\delta B = B \oplus B'$ to booleans ΔB
For each differential byte δB: $\Delta B = 0$ if $\delta B = 0$; $\Delta B = 1$ otherwise
Minimize number of active S-Boxes ($\Delta SX = 1$)

Step 2: Concretize booleans to differential bytes



Two-step Solving Process

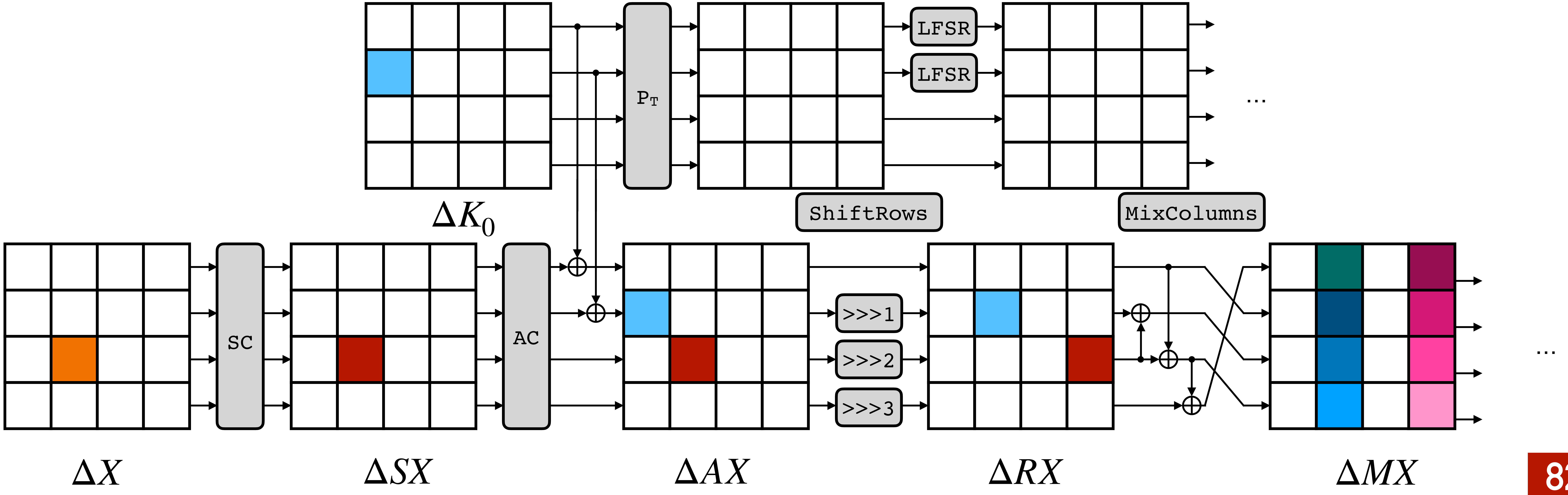
[Biryukov *et al.*, 10][Fouque *et al.*, 13]

Step 1: Abstract differential bytes $\delta B = B \oplus B'$ to booleans ΔB

For each differential byte δB: $\Delta B = 0$ if $\delta B = 0$; $\Delta B = 1$ otherwise

Minimize number of active S-Boxes (ΔSX = 1)

Step 2: Concretize booleans to differential bytes



Two-step Solving Process

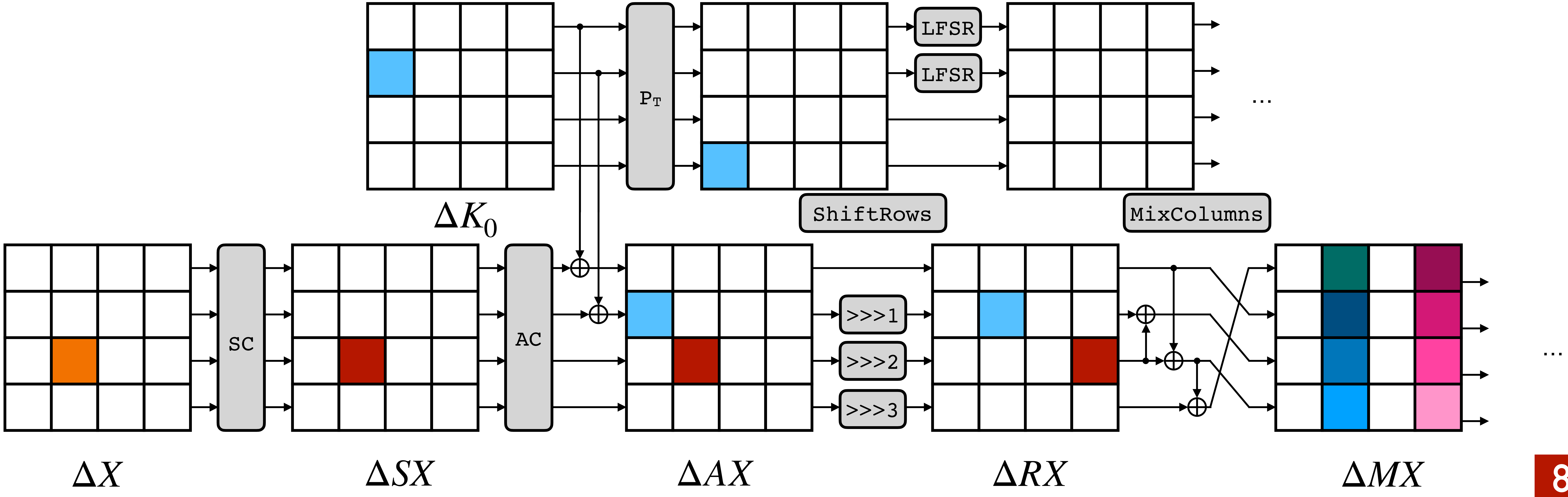
[Biryukov *et al.*, 10][Fouque *et al.*, 13]

Step 1: Abstract differential bytes $\delta B = B \oplus B'$ to booleans ΔB

For each differential byte δB: $\Delta B = 0$ if $\delta B = 0$; $\Delta B = 1$ otherwise

Minimize number of active S-Boxes (ΔSX = 1)

Step 2: Concretize booleans to differential bytes



Two-step Solving Process

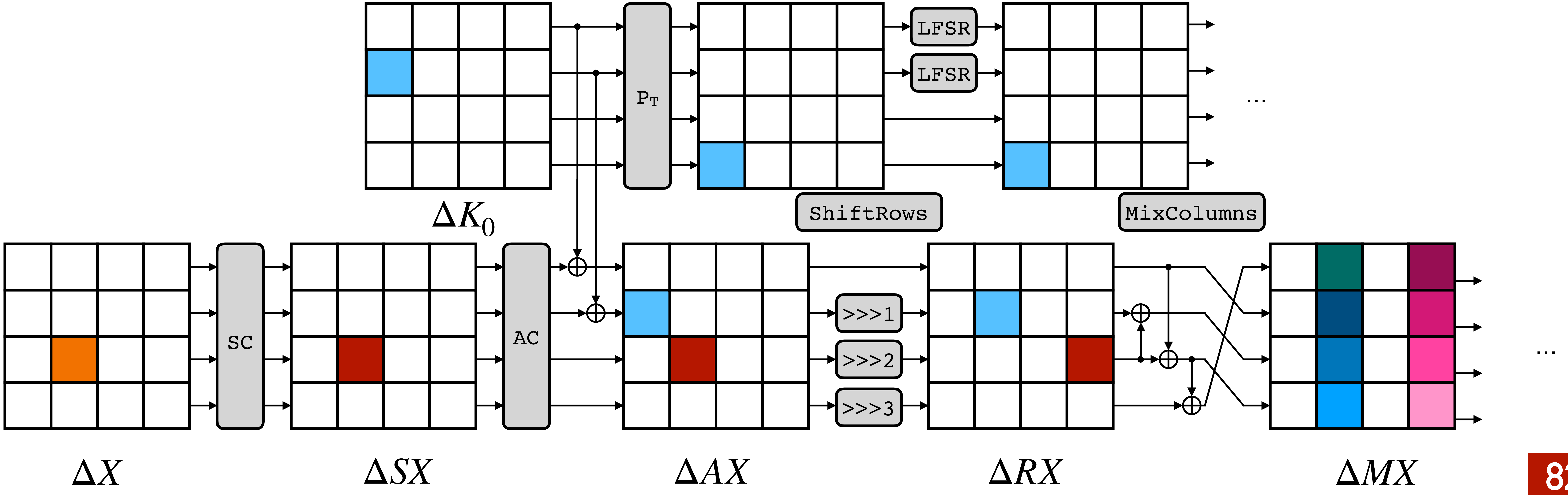
[Biryukov *et al.*, 10][Fouque *et al.*, 13]

Step 1: Abstract differential bytes $\delta B = B \oplus B'$ to booleans ΔB

For each differential byte δB: $\Delta B = 0$ if $\delta B = 0$; $\Delta B = 1$ otherwise

Minimize number of active S-Boxes (ΔSX = 1)

Step 2: Concretize booleans to differential bytes



Two-step Solving Process

[Biryukov *et al.*, 10][Fouque *et al.*, 13]

Step 1: Abstract differential bytes $\delta B = B \oplus B'$ to booleans ΔB

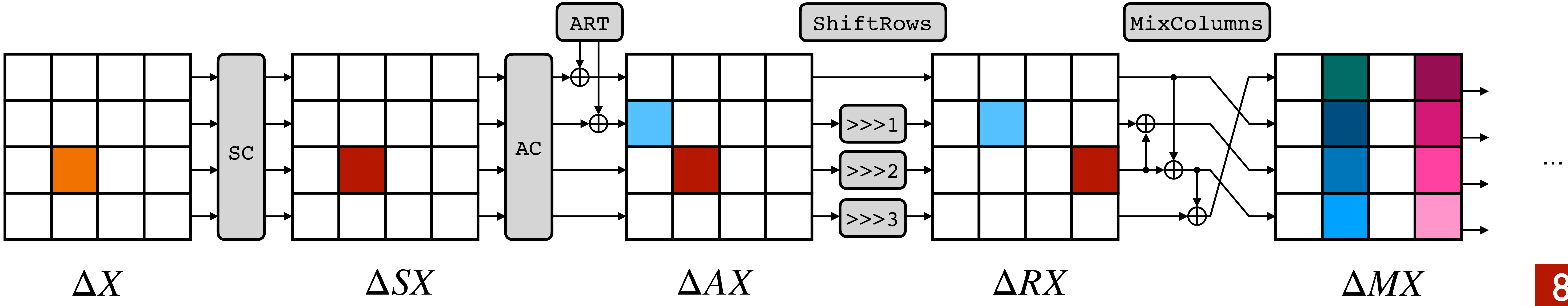
Step 2: Concretize booleans to differential bytes

If $\Delta B = 0$ then set δB to 0; otherwise search for $\delta B \in [1, 2^n]$

If not possible: byte-inconsistent solution

If possible: byte-consistent solution

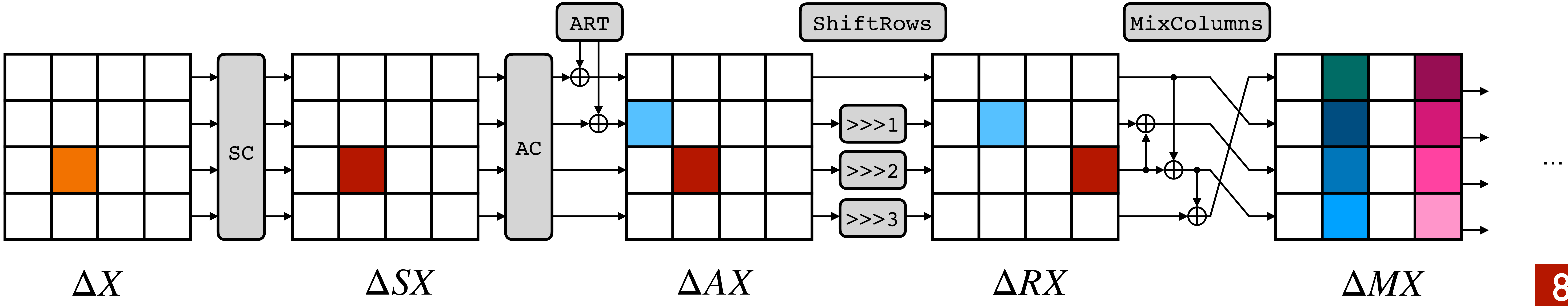
Maximize the probability Pr$[\delta SX_r \,|\, \delta X, \delta K_0]$



Tools Used

Step 1: Integer Linear Programming (ILP)
Constraint Programming (CP)
Satisfiability Modulo Theory (SMT/SAT)
Ad-hoc Method

Step 2: CP

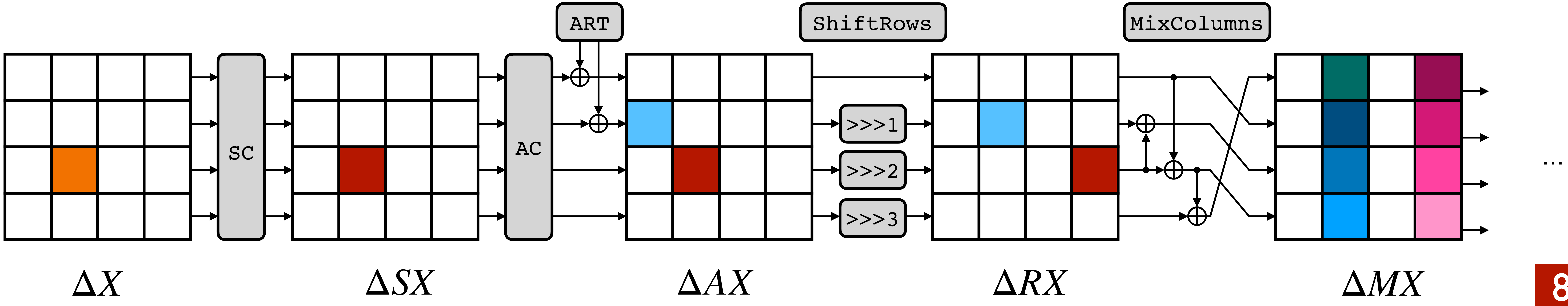


Tools Used

Step 1: Integer Linear Programming (ILP)
Constraint Programming (CP)
Satisfiability Modulo Theory (SMT/SAT)
Ad-hoc Method

Step 2: CP

Attack models**: SK**, **TK1**, **TK2** and **TK3** for both 64-bit and 128-bit versions



Main Results

- First Ad-hoc algorithm for **Step1** handling **all SKINNY models** including **TK3** Solutions output in reasonable time for **TK1** and **TK2**.

- First use of CP solver for **Step1**. Much faster previously used MILP approach.

- New results regarding the probability of the best differential trails for both the **TK1** and **TK2**

    - Best differential related-tweakey characteristics up to 14 rounds for **TK1** model and up to 12 rounds for the **TK2** model of SKINNY-128

    - No differential characteristic with probability higher than 2^{-128} for 15 rounds in the **TK1**

Main Results: Skinny-64

	Nb Rounds	Objstep1	Nb sol. Step 1	Step 2 time	Best Pr
SK	7	26	2	1s	2^{-52}
SK	8	36	17	1s	$< 2^{-64}$
TK1	10	23	1	1s	2^{-46}
TK1	11	32	2	1s	2^{-64}
TK2	13	25 → 27	10	1s	2^{-55}
TK2	14	31	1	1s	$< 2^{-64}$
TK3	15	24 → 26	46	2s	2^{-54}
TK3	16	27 → 31	87	4s	2^{-64}
TK3	17	31	2	1s	$< 2^{-64}$

Main Results: Skinny-128

	Nb Rounds	Objstep1	Nb sol. Step 1	Step 2 time	Best Pr
SK	9	41 → 43	52	16s	2^{-86}
SK	10	46 → 48	48	11s	2^{-96}
SK	11	51 → 52	15	4s	2^{-104}
SK	12	55 → 56	11	6s	2^{-112}
SK	13	58 → 61	18	2m27s	2^{-123}
SK	14	61 → 63	6	21s	$< 2^{-128}$
TK1	8	13 → 16	14	4s	2^{-33}
TK1	9	16 → 20	6	3s	2^{-41}
TK1	10	23 → 27	6	4s	2^{-55}
TK1	11	32 → 36	531	37s	2^{-74}
TK1	12	38 → 46	186 482	213m	2^{-93}
TK1	13	41 → 53	2 385 482	2 days	$2^{-106.2}$
TK1	14	45 → 59	11 518 612	20 days	2^{-120}
TK1	15	49 → 63	7 542 053	25 days	$< 2^{-128}$

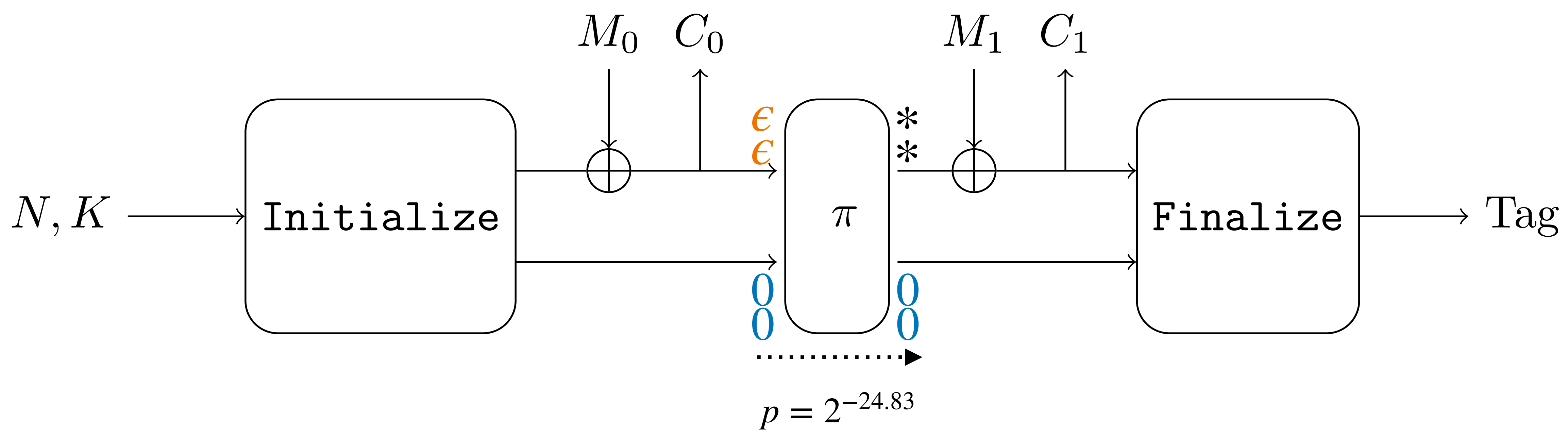
Main Results: Skinny-128

	Nb Rounds	Objstep1	Nb sol. Step 1	Step 2 time	Best Pr
TK2	9	$9 \rightarrow 10$	7	3s	2^{-20}
TK2	10	$12 \rightarrow 17$	132	11s	$2^{-34.4}$
TK2	11	$16 \rightarrow 25$	4 203	6m	$2^{-51.4}$
TK2	12	$21 \rightarrow 35$	1 922 762	512m	$2^{-70.4}$
TK2	13	$25 \rightarrow 44$	-	not solved	$> 2^{-89.7}$
TK2	14	$31 \rightarrow 54$	-	not solved	$> 2^{-108.4}$
TK2	15	$35 \rightarrow 56$	-	not solved	$> 2^{-113.2}$
TK2	16	$40 \rightarrow 63$	-	not solved	$> 2^{-127.6}$
TK2	17	$43 \rightarrow 63$	-	not solved	-
TK2	18	$47 \rightarrow 63$	62 681 709	not solved	-
TK2	19	$52 \rightarrow 63$	772 163	280m	$< 2^{-128}$

Forgery on Spook: Some More Details

Forgery

Attack Outline

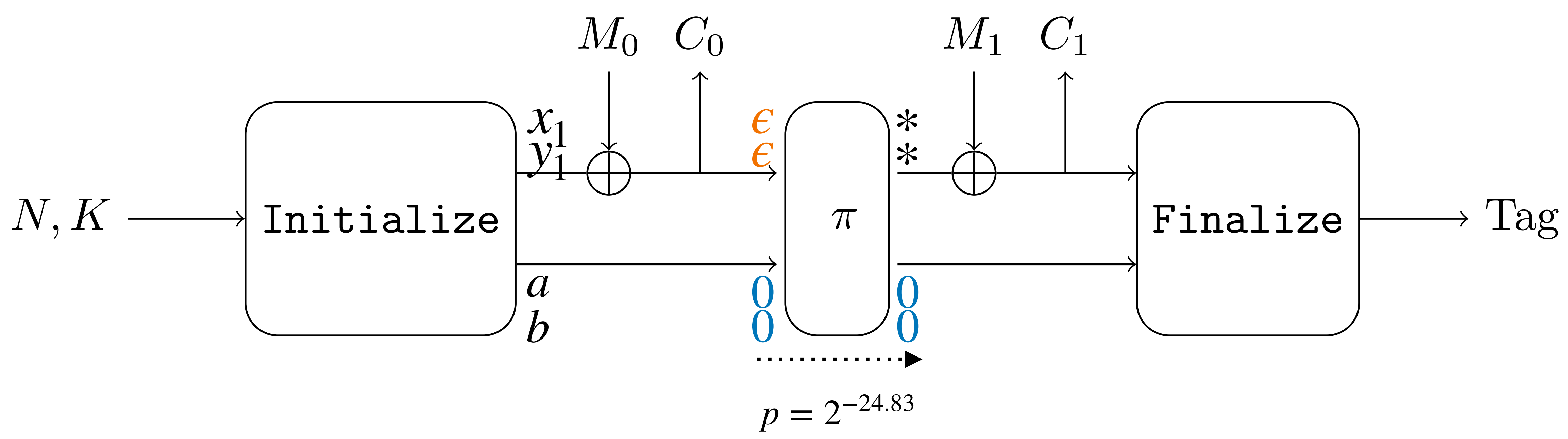


2 different plaintexts that yield the **same tag**

(M_0, M_1) and **(M'_0, M'_1)** that yield a **(0,0,0,0) difference after π**

Forgery

Attack Outline



1. **Query 1**: encrypt a two-block (4 bundles) message (0,0)(0,0) to recover the 2-bundle rate value after `Initialize` (x_1, y_1) **(C_0)**.

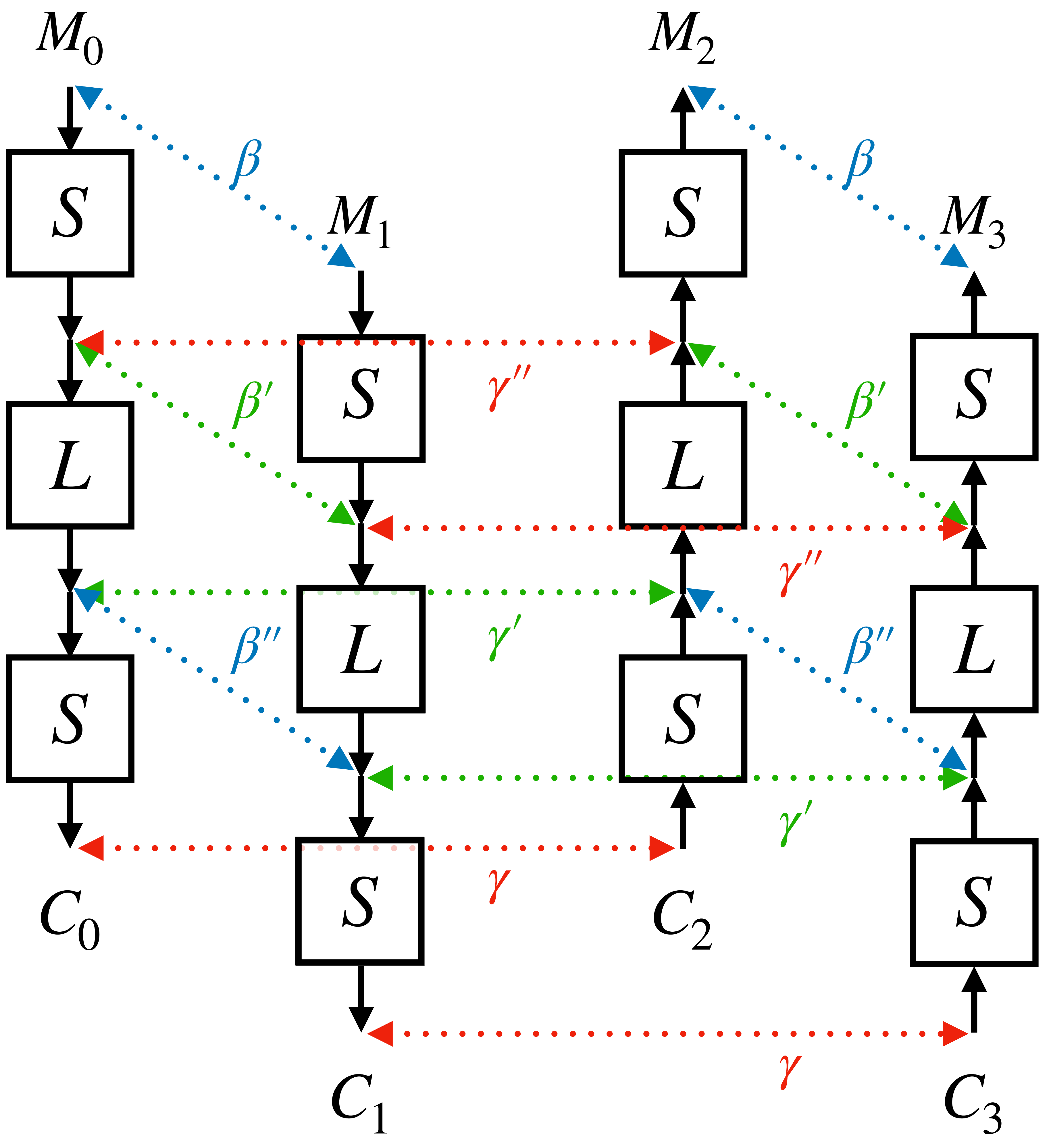
2. Generate two pairs of **rate bundles** $(x_1', y_1'), (x_1'', y_1'')$ that satisfy the truncated trail with probability p.

3. **Query 2 and 3**: get the difference after π.
   - Encrypt $(x_1 \oplus x_1', y_1 \oplus y_1'), (0,0)$ to obtain the **value of the rate after π on** (x_1', y_1', a, b), denoted by (c_2', c_3') **(C_1)**.
   - Encrypt $(x_1 \oplus x_1'', y_1 \oplus y_1''), (0,0)$ to obtain the **value of the rate after π on** (x_1'', y_1'', a, b), denoted by (c_2'', c_3'') **(C_1)**.

4. Cancel out the difference after π.
   - $(x_1 \oplus x_1', y_1 \oplus y_1'), (c_2', c_3')$ and $(x_1 \oplus x_1'', y_1 \oplus y_1''), (c_2'', c_3'')$ yield the same internal state before `Finalize` with probability $p \simeq 2^{-24.83}$.

FBCT: 2 Rounds and More

Two-round case



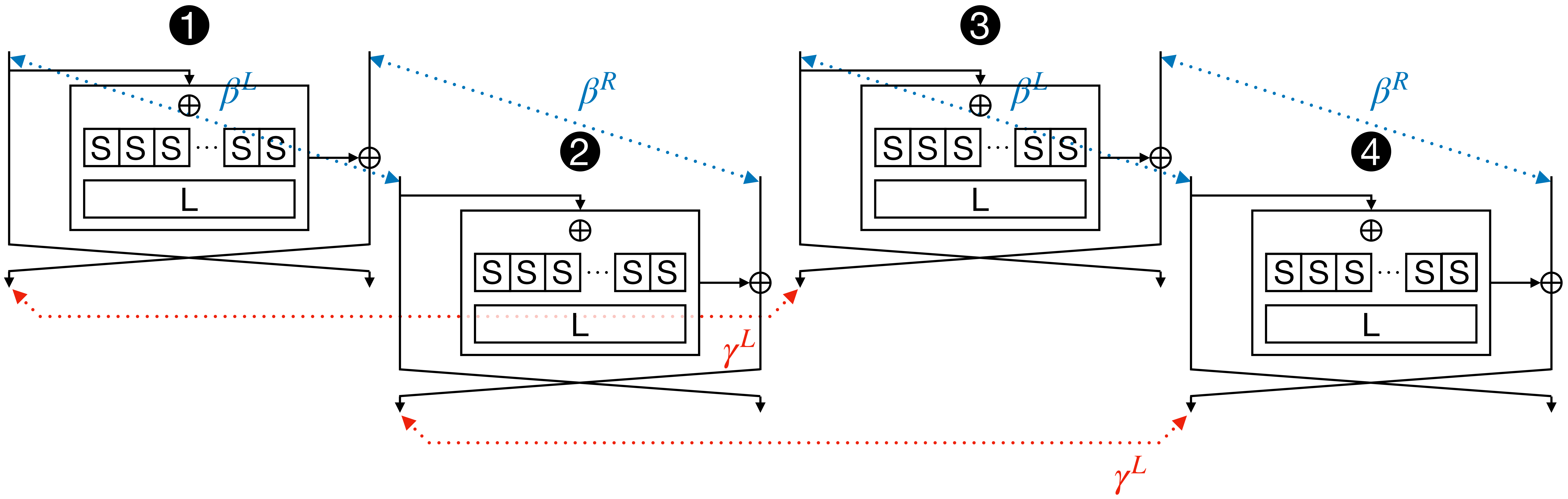
Boomerang Switch in Multiple Rounds
Wang & Peyrin, *ToSC 2019*

Boomerang Connectivity Table Revisited. Application to SKINNY and AES
Song, Qin & Hu, *ToSC 2019*

$$BDT(\beta, \beta', \gamma'') = \#\{x \mid S^{-1}(S(x) \oplus \gamma'') \oplus S^{-1}(S(x \oplus \beta) \oplus \gamma'') = \beta, \\ S(x) \oplus S(x \oplus \beta) = \beta'\}$$

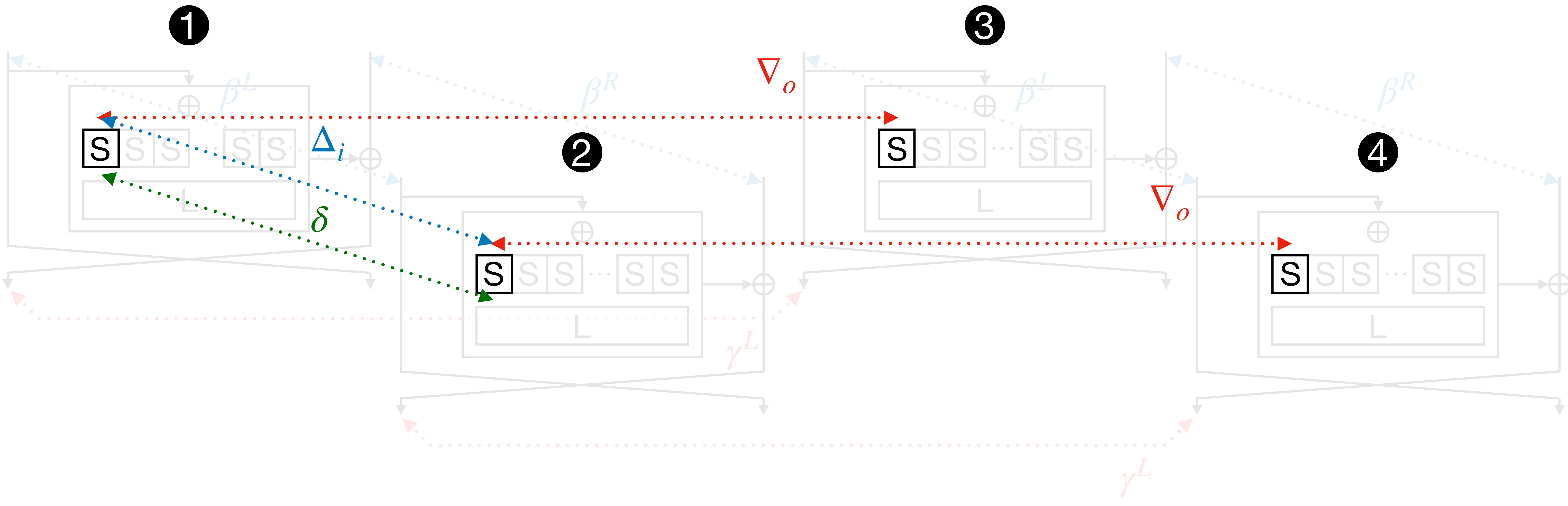
$$BDT'(\gamma, \gamma', \beta'') = \#\{x \mid S(S^{-1}(x) \oplus \beta'') \oplus S(S^{-1}(x \oplus \gamma) \oplus \beta'') = \gamma, \\ S^{-1}(x) \oplus S^{-1}(x \oplus \gamma) = \gamma'\}$$

Two-round case



$$FBDT(\Delta_i, \delta, \nabla_o) = \#\{x \in \mathbb{F}_2^n \,|\, S(x) \oplus S(x \oplus \Delta_i) \oplus S(x \oplus \nabla_o) \oplus S(x \oplus \Delta_i \oplus \nabla_o) = 0$$
$$\text{and } S(x) \oplus S(x \oplus \Delta_i) = \delta\}$$

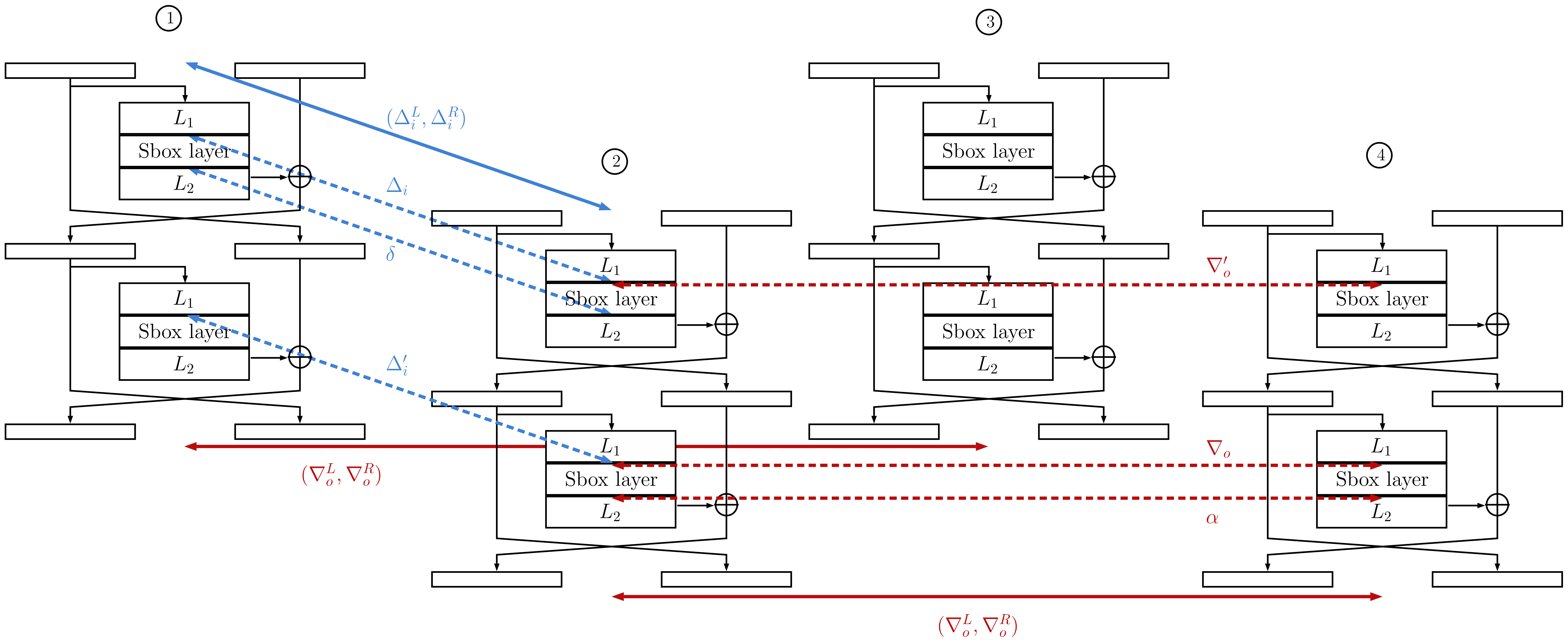
Two-round case



$$FBDT(\Delta_i, \delta, \nabla_o) = \#\{x \in \mathbb{F}_2^n \,|\, S(x) \oplus S(x \oplus \Delta_i) \oplus S(x \oplus \nabla_o) \oplus S(x \oplus \Delta_i \oplus \nabla_o) = 0$$

$$\text{and } S(x) \oplus S(x \oplus \Delta_i) = \delta\}$$

Two-round case

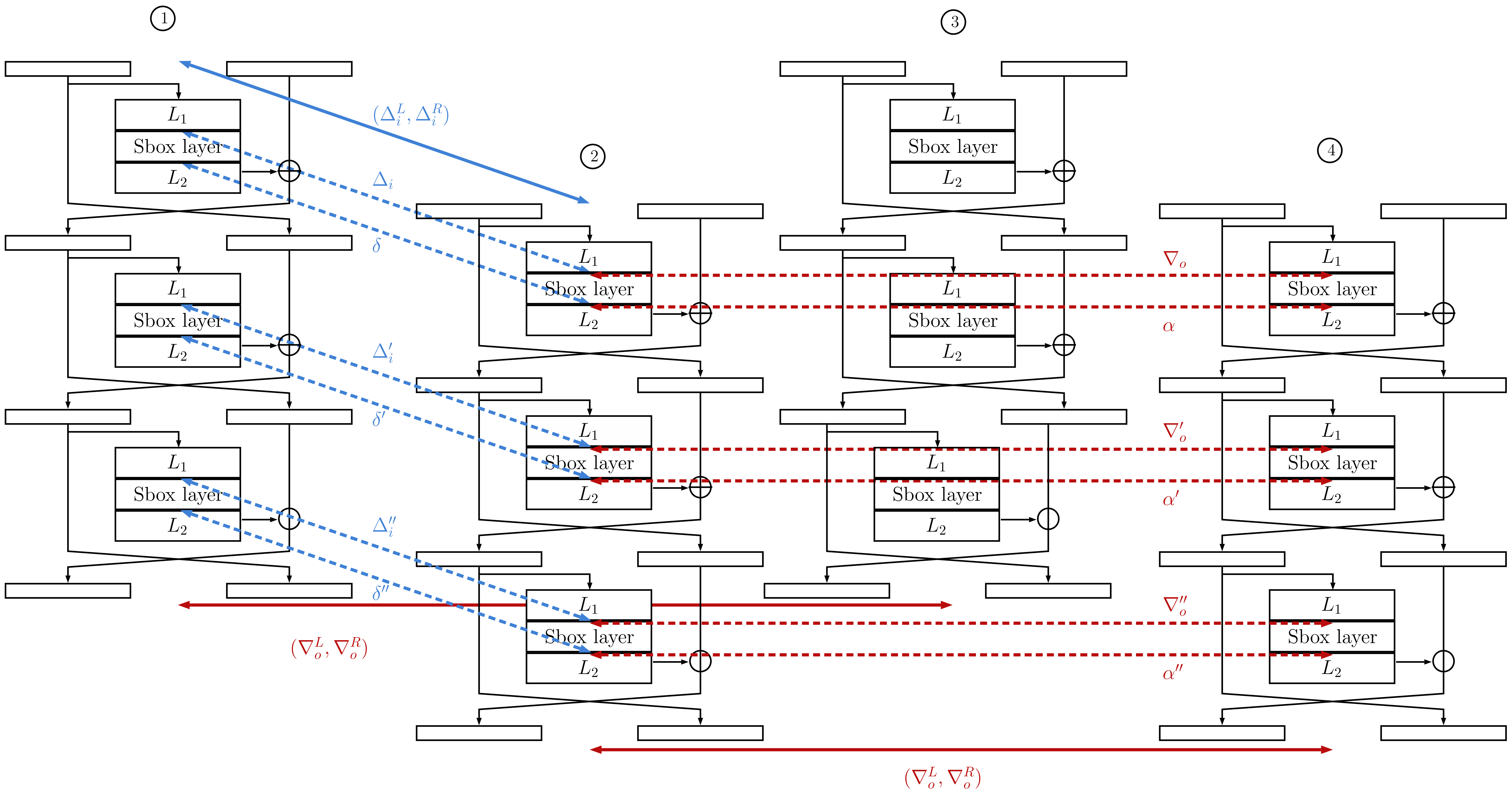


$$FBDT(\Delta_i, \delta, \nabla_o) = \#\{x \in \mathbb{F}_2^n \mid S(x) \oplus S(x \oplus \Delta_i) \oplus S(x \oplus \nabla_o) \oplus S(x \oplus \Delta_i \oplus \nabla_o) = 0$$

$$\text{and } S(x) \oplus S(x \oplus \Delta_i) = \delta\}$$

$$\boxed{2^{-2tn} \times \sum_{0 \le \delta, \alpha < 2^n} FBDT(\Delta_i, \delta, \nabla_o') \times FBDT(\nabla_o, \alpha, \Delta_i')}$$

Switches over 3 rounds and more…



FBET table:

$$\#\{x \in \mathbb{F}_2^n \,|\, S(x) \oplus S(x \oplus \Delta_i) \oplus S(x \oplus \nabla_o) \oplus S(x \oplus \Delta_i \oplus \nabla_o) = 0,$$
$$S(x) \oplus S(x \oplus \Delta_i) = \delta,$$
$$S(x \oplus \Delta_i) \oplus S(x \oplus \Delta_i \oplus \nabla_o) = \alpha\}$$