(Almost) Three Decades of AES Cryptanalysis

Orr Dunkelman

Computer Science Dept., University of Haifa
Security in Telecommunications (SecT), TU Berlin

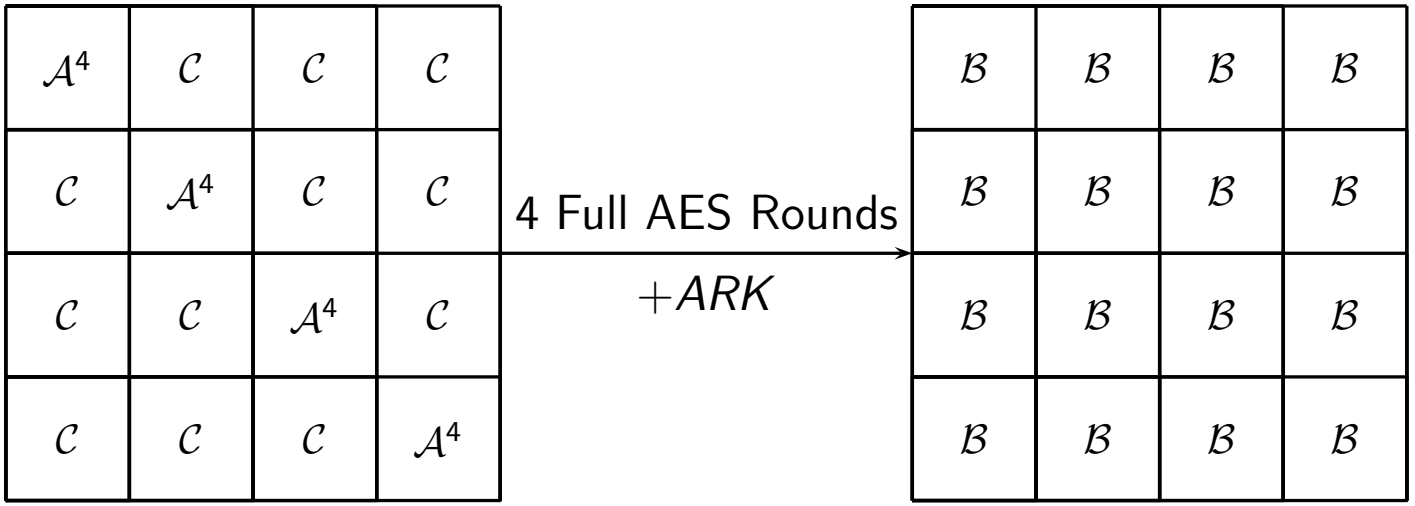
Outline

Post-Talk Comment

Due to time constraints, the slides may include typos and errors. If you find your paper should have been cited, and it wasn't, please contact me by email.
If there are many attacks on AES that are not covered at all due to lack of time, including important results (e.g., the related-key attacks on the full AES).

SQUARE property [DR97]

Consider a set of 2^{32} plaintexts:



- $\mathcal{C}$ — constant (same value for all values in the set)
- $\mathcal{A}^4$ — takes all possible values in these 32 bits
- $\mathcal{B}$ — XOR all the values in the set, and obtain a zero

Attacks based on the SQUARE property [DR97]

- By swapping the order of the *MC* and the *ARK* operation, one can efficiently test for ballanceness.

Rounds	Data	Time	Memory
4	2^9	2^9	Small
5	2^{11}	2^{40}	Small
5	2^{32}	2^{40}	2^{32}
6	2^{32}	2^{72}	2^{32}

Obviously, this can extend for 7-round AES at increased time complexity cost of 2^{200} (without key scheduling considerations).

Partial Sums [F+00]

- Let $k_0, k_1, \ldots, k_4$ by the 5 key **bytes** in the evaluation of the SQUARE property in byte y_i, and let the c^j be the ciphertext bytes going into the calculation.
- The sum computed is:

$$\begin{aligned} y_i \;=\; S^{-1}\left(k_4 \oplus \right. & \; S_0\left(c_i^0 \oplus k_0\right) \oplus S_1\left(c_i^1 \oplus k_1\right) \\ & \left. \oplus S_2(c_i^2 \oplus k_2) \oplus S_3(c_i^3 \oplus k_3)\right). \end{aligned}$$

Partial Sums (cont.)

- The main idea behind partial sums is the "reduction" of the candidates that need to be considered in each step.
- Namely, let $A[\cdot]$ be a bitmap array of 2^{32} values, counting the parity of how many times c^0, c^1, c^2, and c^3 appear.
- Start by guessing two subkey bytes k_0 and k_1, which allows computing

$$t_1 = S_0(c_i^0 \oplus k_0) \oplus S_1(c_i^1 \oplus k_1)$$

- This means that we can now test for the given key guess and the given 2^{32} values how many times (t_1, c_2, c_3) happens.

Partial Sums (cont.)

- We now guess k_2 for each of the 2^{24} options and compute

$$t_2 = t_1 \oplus S_1(c_i^2 \oplus k_2)$$

- And we do again for all (t_2, c_3) values with k_3, to obtain t_3 values, for which we guess k_4.

Partial Sums (cont.)

- We now guess k_2 for each of the 2^{24} options and compute

$$t_2 = t_1 \oplus S_1(c_i^2 \oplus k_2)$$

- And we do again for all (t_2, c_3) values with k_3, to obtain t_3 values, for which we guess k_4.
- The complexity is thus $2^{32} \cdot 2^{16} = 2^{48}$ operations for 6-round attack.

Attacking 7-Round AES-128 [F+00]

- To attack 7-round AES-128 one cannot add a round for the key recovery.
- The solution is to extend the SQUARE property from 4-round to 5-round.
- This is "essentially" done by taking the entire codebook.

Attacking 7-Round AES-128 [F+00]

- To attack 7-round AES-128 one cannot add a round for the key recovery.
- The solution is to extend the SQUARE property from 4-round to 5-round.
- This is "essentially" done by taking the entire codebook.
- OK, not the entire codebook.
- This does not work, the entire code book always sums to zero.
- Solution: Divide the entire code book to sets defined by a diagonal and another byte (i.e., 2^{40} sets of 2^{88} plaintexts each).
- Each such set also needs to be "balanced".

Attacking 7-Round AES-128 (cont.)

- The trick is one can evaluate these sets (called herds) very efficiently, by concentrating on 32-bit of the plaintext and 32-bit of the ciphertext.
- The full attack takes the entire codebook (with a "few" missing values), and counts these 64-bit appearances.
- Then, the attack is very straightforward.
- Time complexity: 2^{128} counter increments (which are estimated as 2^{120} encryptions).
- Data complexity: $2^{128} - 2^{119}$ CPs.

Summary of the [F+00] Results

Rounds	Key Size	Data	Time	Memory
6	All	$6 \cdot 2^{32}$	2^{44}	2^{32}
7	192	$19 \cdot 2^{32}$	2^{155}	2^{32}
7	256	$21 \cdot 2^{32}$	2^{172}	2^{32}
7	All	$2^{128} - 2^{119}$	2^{120}	2^{64}
8	192	$2^{128} - 2^{119}$	2^{188}	2^{64}
8	256	$2^{128} - 2^{119}$	2^{204}	2^{64}

The paper also had a related-key differential attack on 9-round AES-256 (using 256 keys, 2^{85} CPs, and 2^{224} time).

Some More Tricks

- One can reduce the data complexity a bit (the 6,19, and 21) by a more careful analysis.
- The calculation of the SQUARE "check" can be done using the FFT trick [TA14].
- Recall

$$\begin{aligned} y_i \;=\; S^{-1}\left(k_4 \oplus \right. & \quad S_0\left(c_i^0 \oplus k_0\right) \oplus S_1\left(c_i^1 \oplus k_1\right) \\ & \oplus S_2(c_i^2 \oplus k_2) \oplus \left. S_3(c_i^3 \oplus k_3)\right). \end{aligned}$$

- This allows performing 2^8 the FFT quick evaluation of all keys k_0, k_1, k_2, k_3 (as we need to know k_4).
- The resulting attack uses $6 \cdot 2^{50}$ additions for the FFT.

Some More Tricks

- One can reduce the data complexity a bit (the 6,19, and 21) by a more careful analysis.
- The calculation of the SQUARE "check" can be done using the FFT trick [TA14].
- Recall

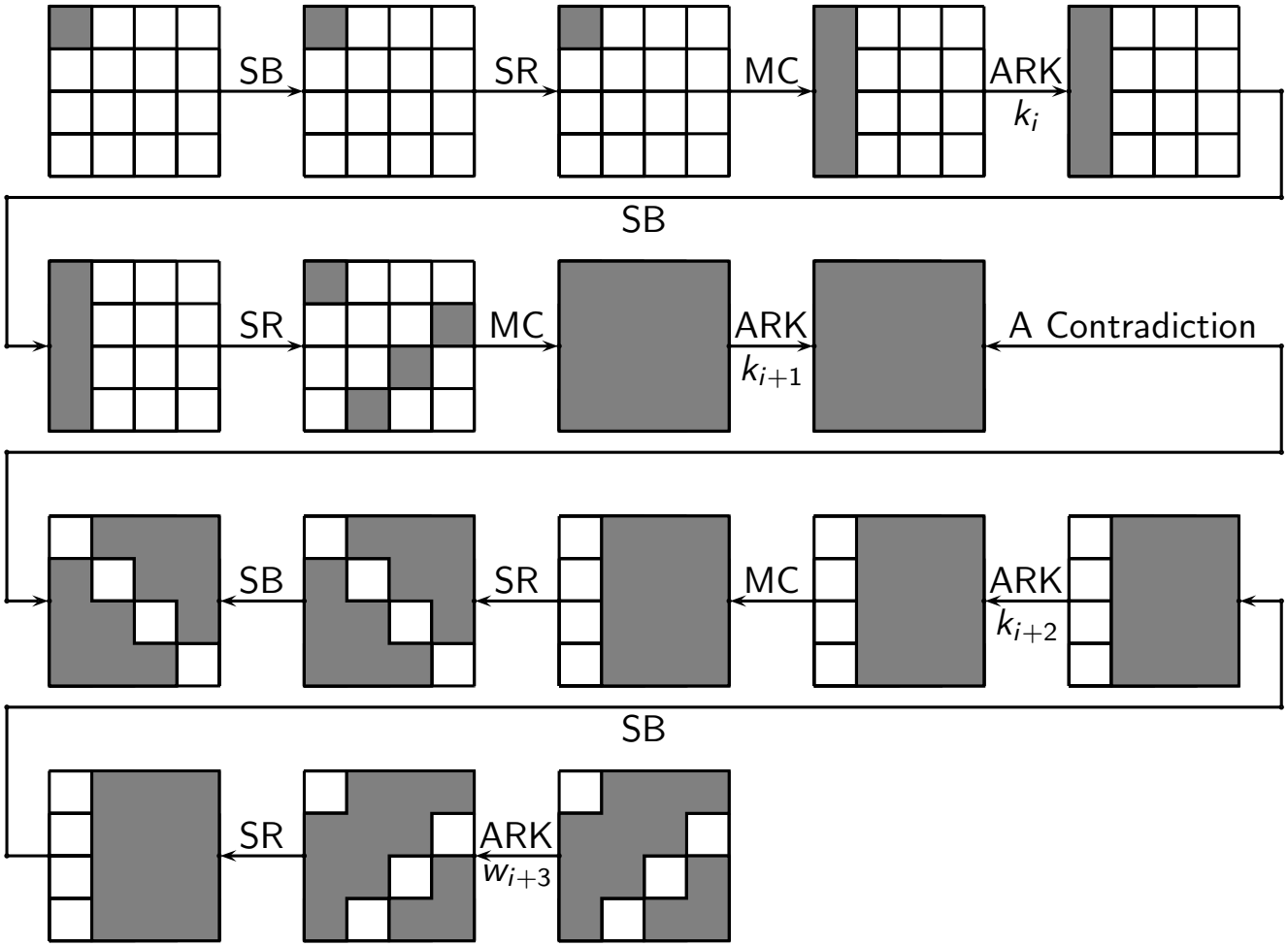
$$y_i = S^{-1}\left(k_4 \oplus \begin{array}{c} S_0\left(c_i^0 \oplus k_0\right) \oplus S_1\left(c_i^1 \oplus k_1\right) \\ \oplus S_2(c_i^2 \oplus k_2) \oplus S_3(c_i^3 \oplus k_3)) \end{array}\right).$$

- This allows performing 2^8 the FFT quick evaluation of all keys k_0, k_1, k_2, k_3 (as we need to know k_4).
- The resulting attack uses $6 \cdot 2^{50}$ additions for the FFT.
- As we shown in [D+24], one can combine FFT with Partial sums to obtain the best 6-round attack on AES.

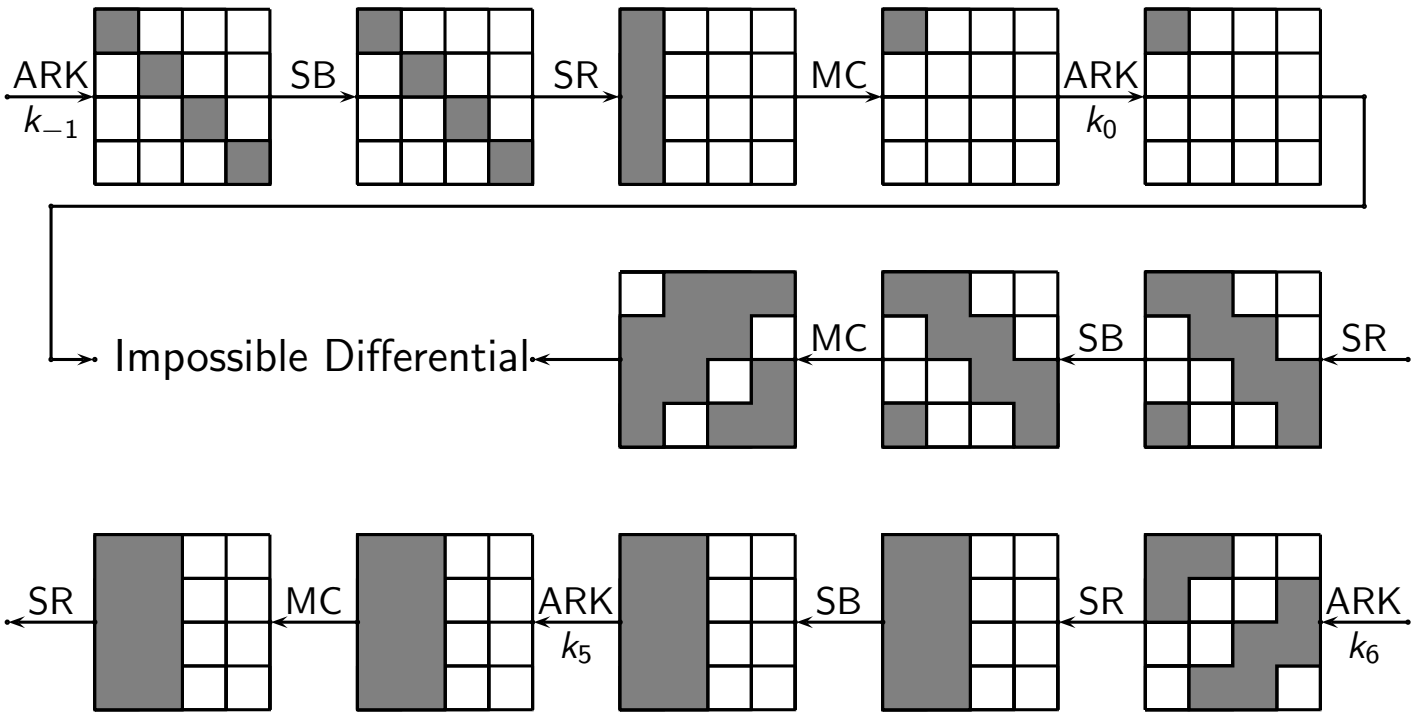
Summary SQUARE

Rounds	Key Size	Data	Time	Memory
5	All	2^8	2^{40}	2^8
6	All	2^{32}	2^{40}	2^{32}
7	192	2^{32}	2^{155}	2^{32}
7	256	2^{32}	2^{172}	2^{32}
7	All	$2^{128} - 2^{119}$	2^{120}	2^{64}
8	192	$2^{128} - 2^{119}$	2^{188}	2^{64}
8	256	$2^{128} - 2^{119}$	2^{204}	2^{64}

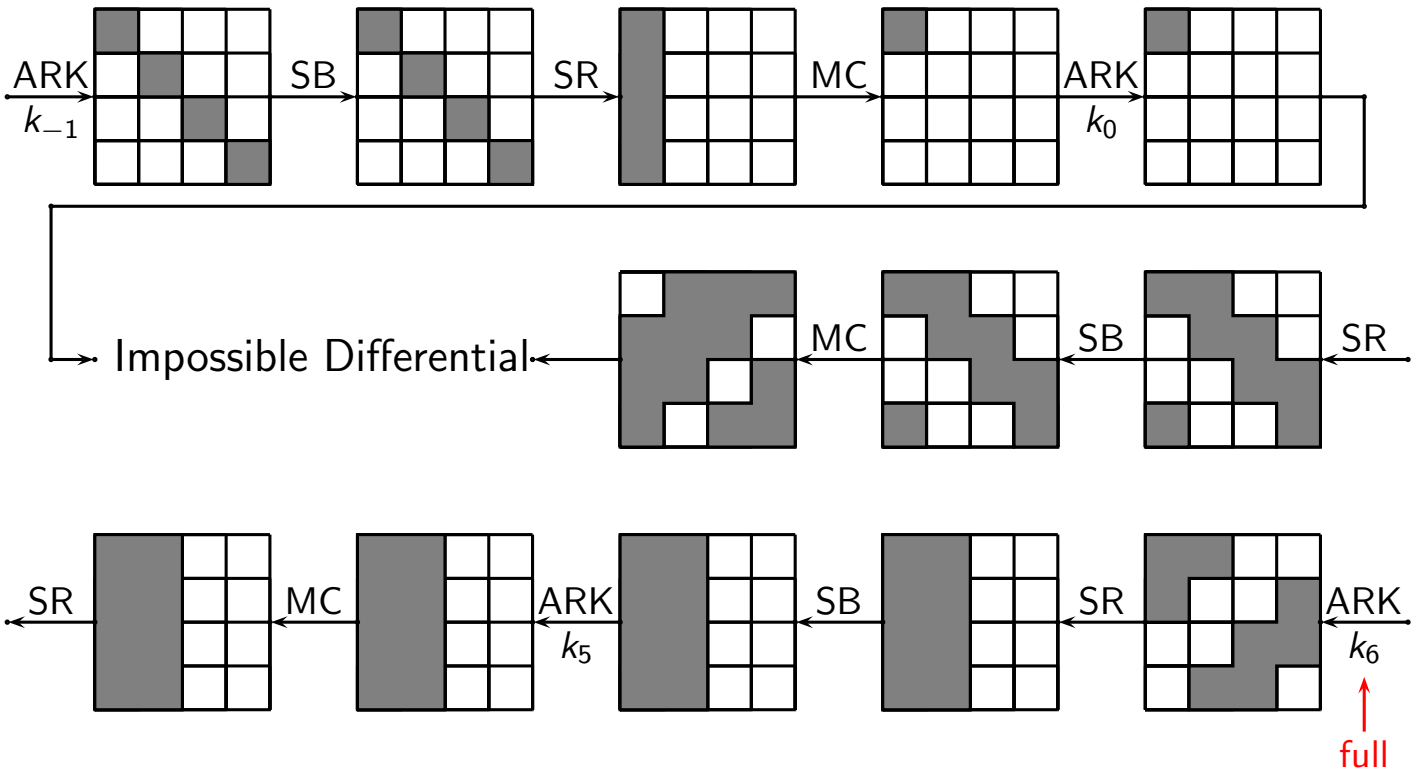
The Basic 4-Round Impossible Differential [BK00]



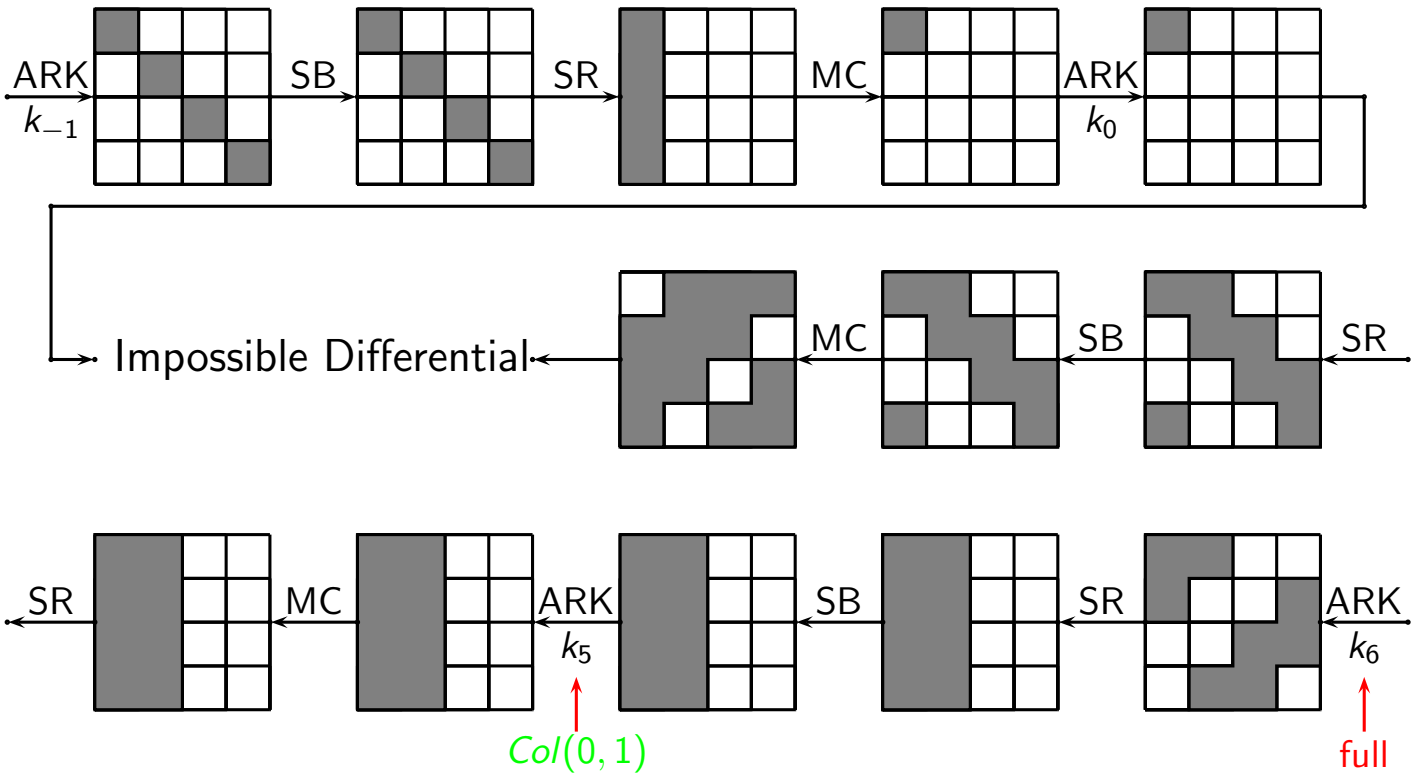
The "Phan" Impossible Differential Attack [P04]



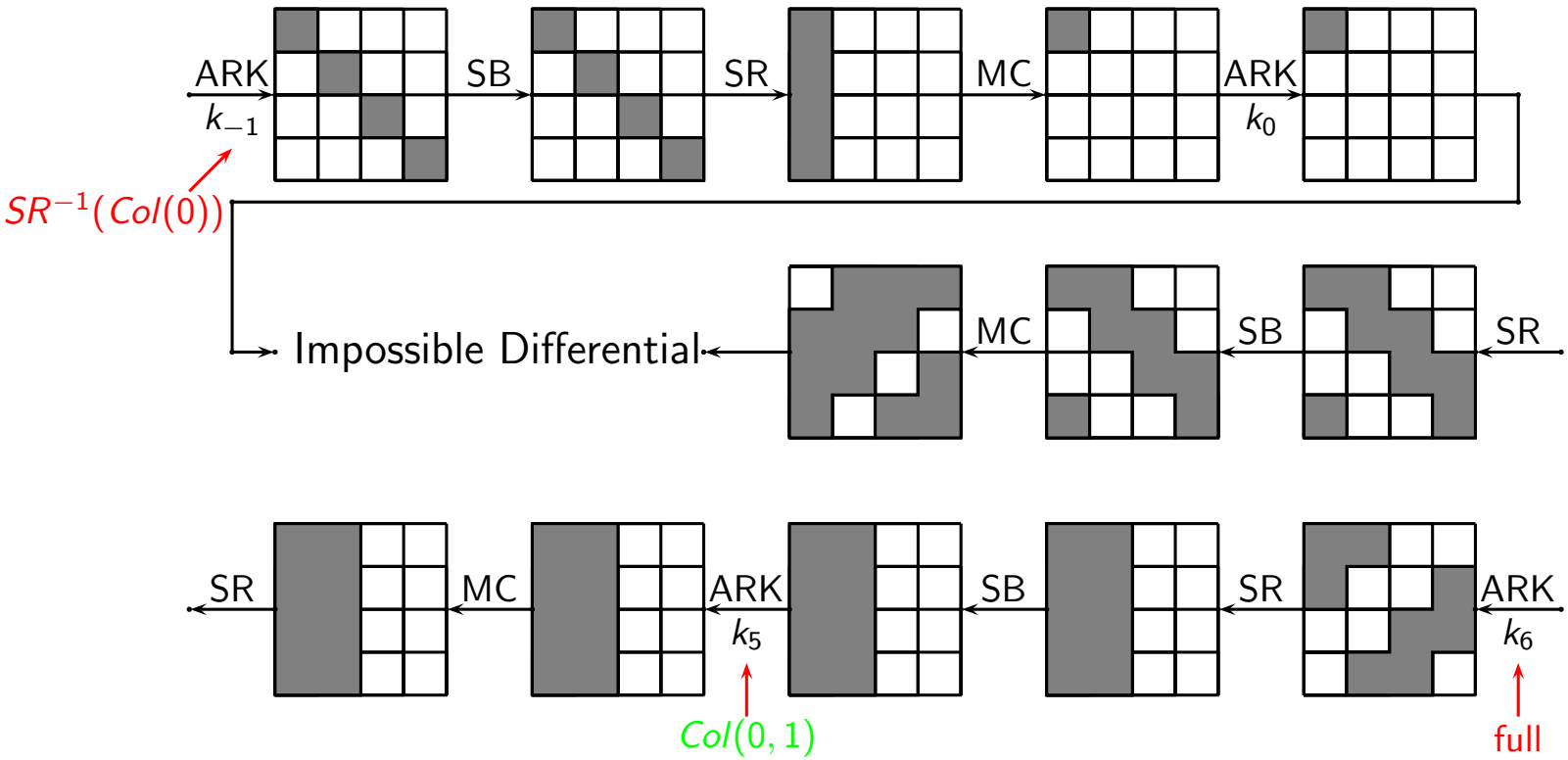
The "Phan" Impossible Differential Attack [P04]



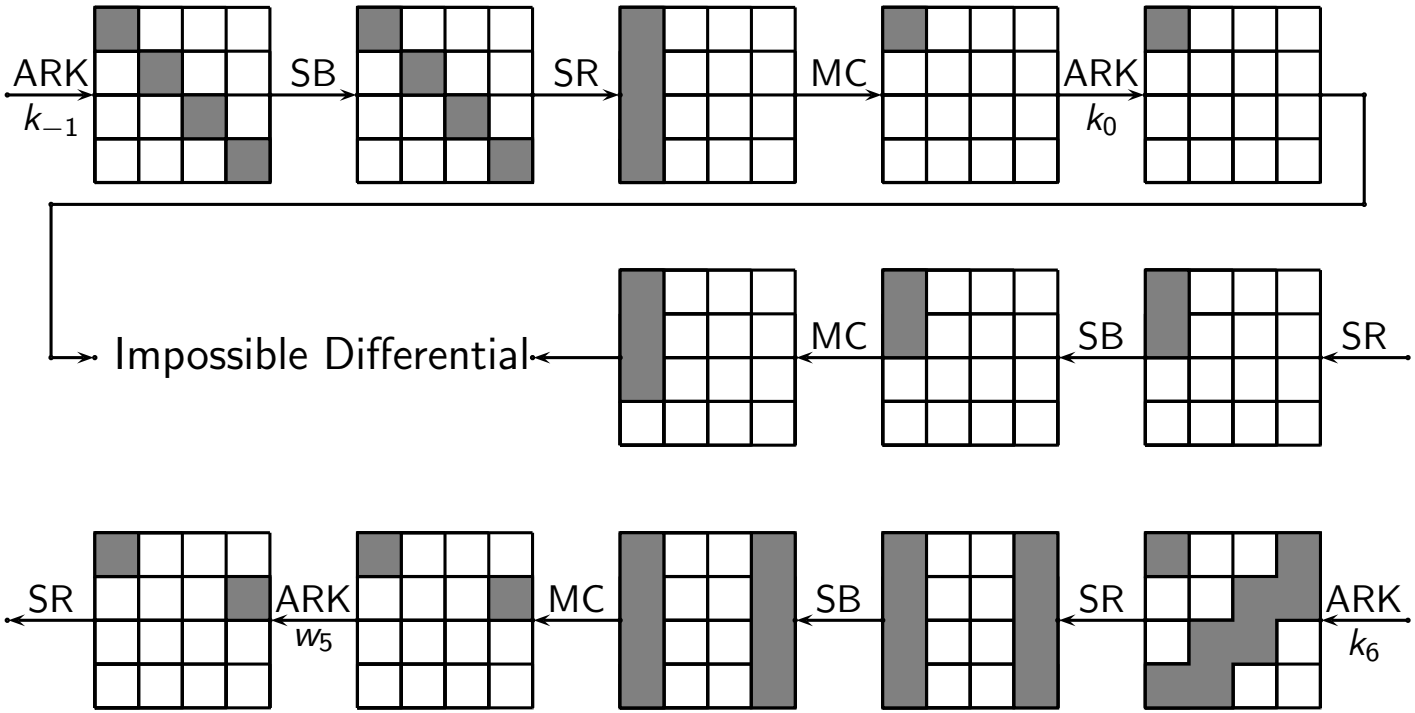
The "Phan" Impossible Differential Attack [P04]



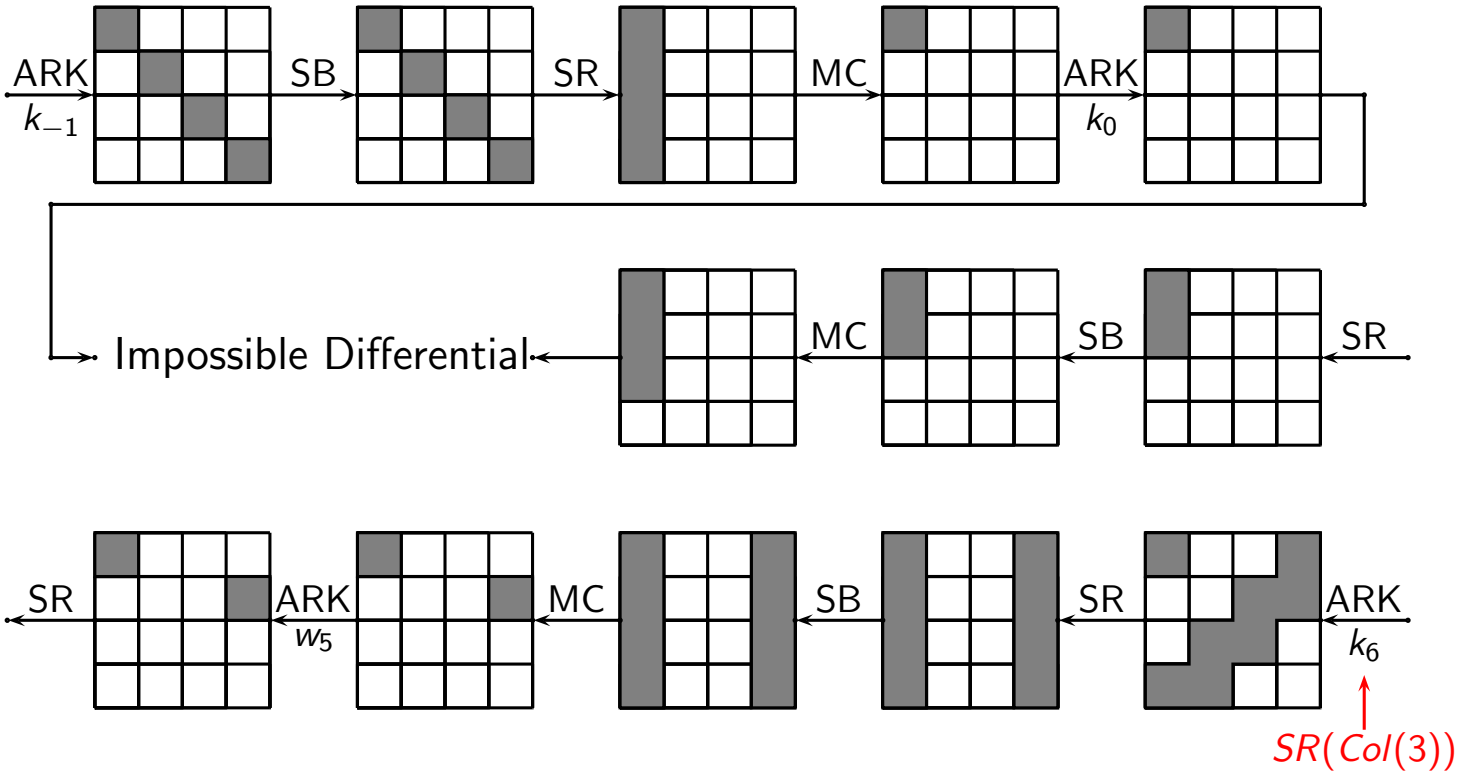
The "Phan" Impossible Differential Attack [P04]



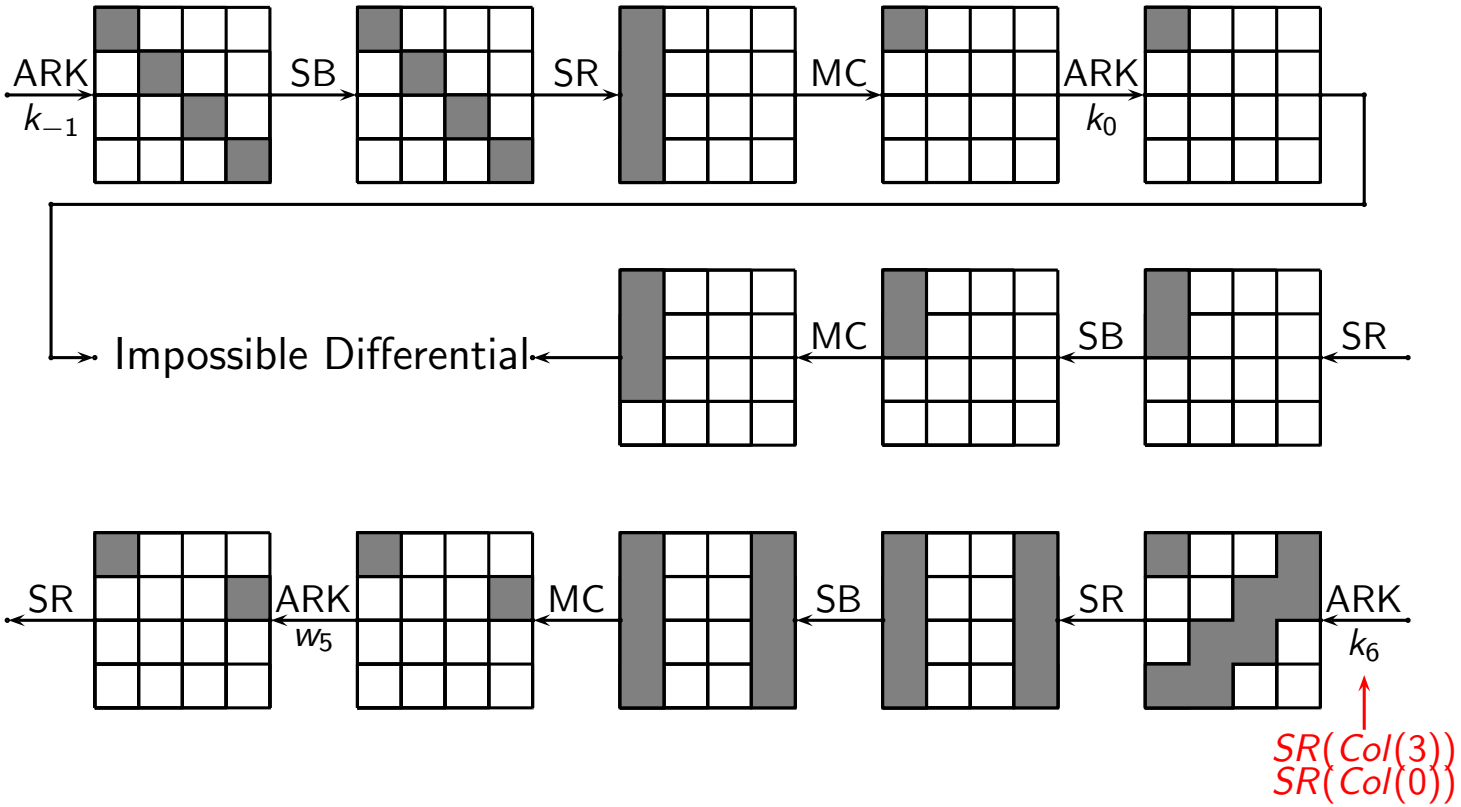
The "Bahrak-Aref" Impossible Differential Attack [BA07]



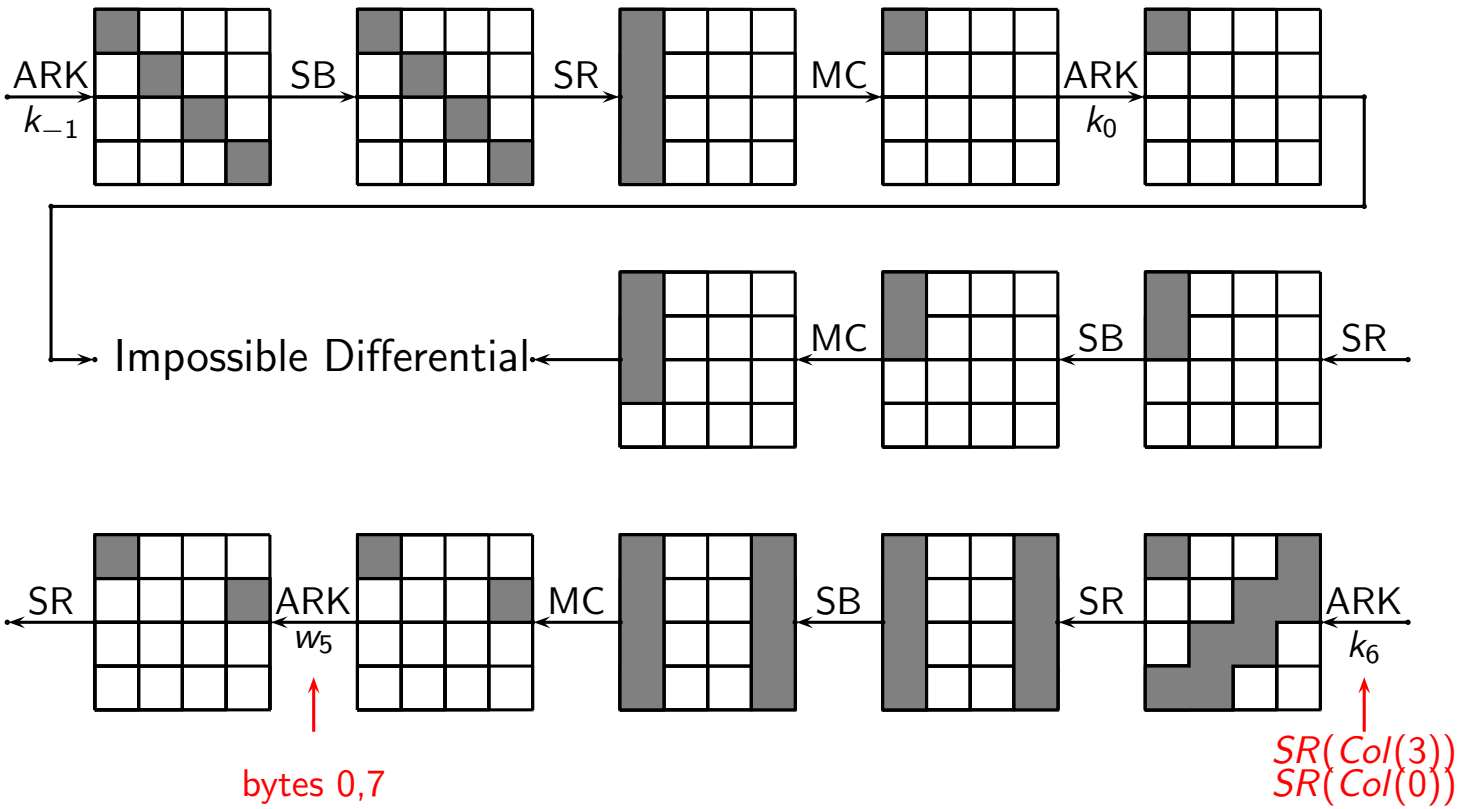
The "Bahrak-Aref" Impossible Differential Attack [BA07]



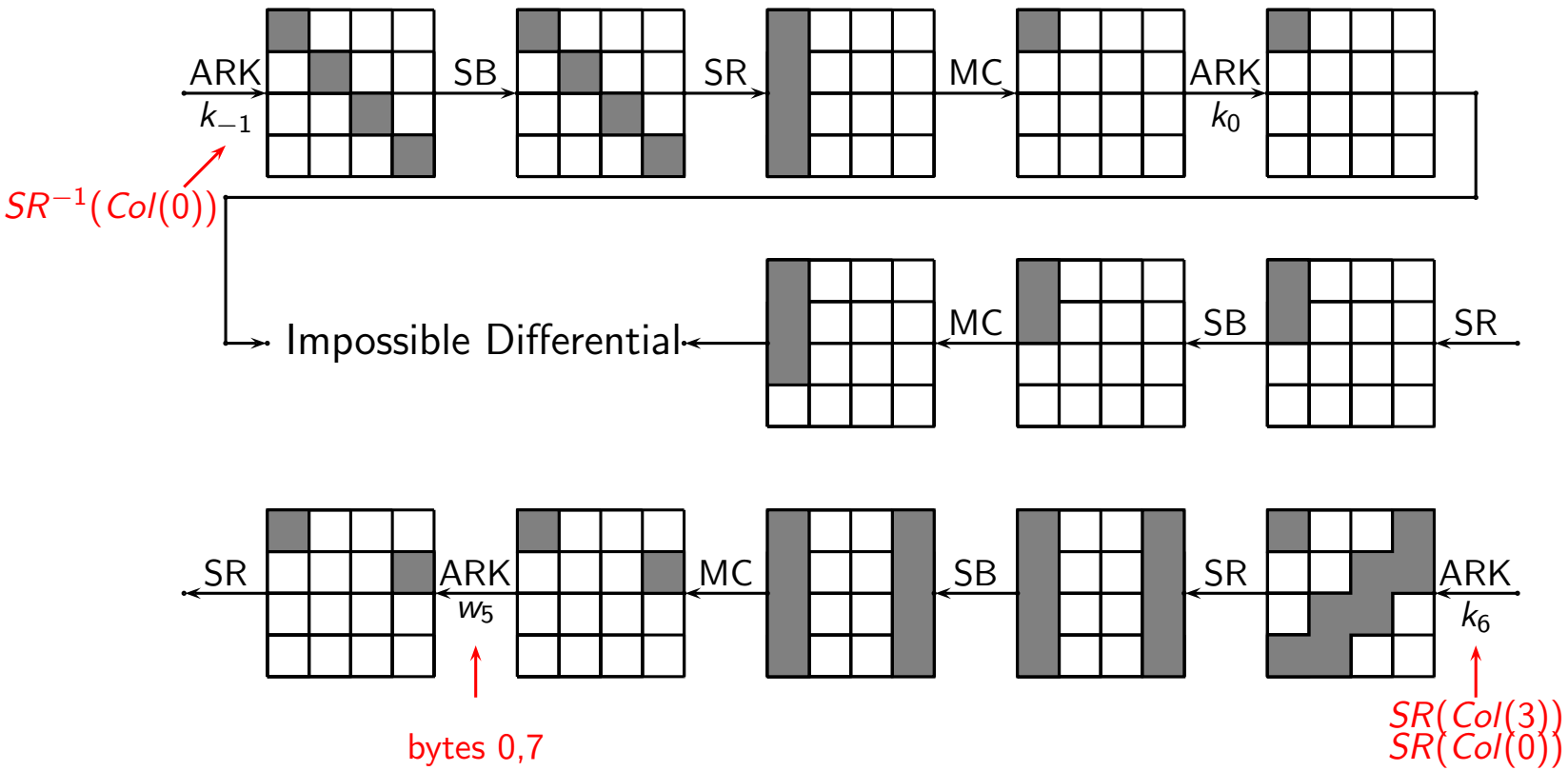
The "Bahrak-Aref" Impossible Differential Attack [BA07]



The "Bahrak-Aref" Impossible Differential Attack [BA07]



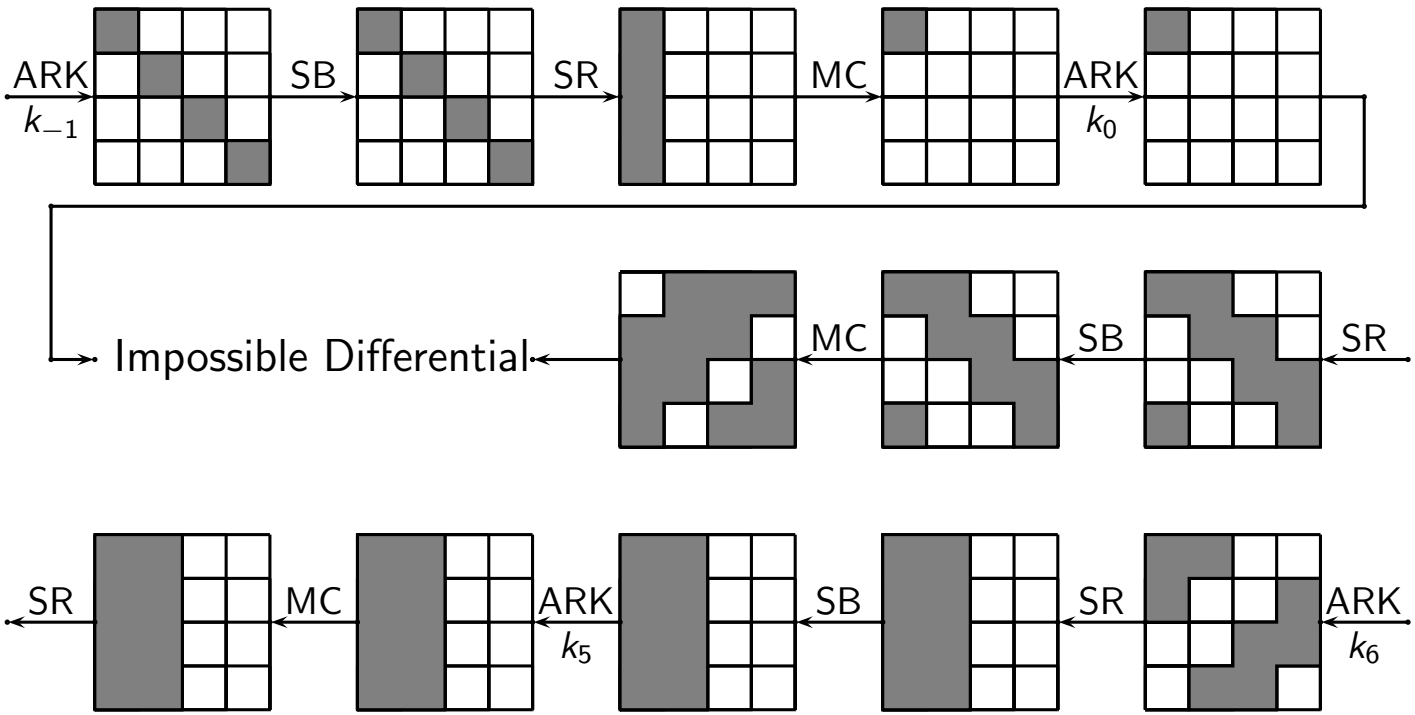
The "Bahrak-Aref" Impossible Differential Attack [BA07]



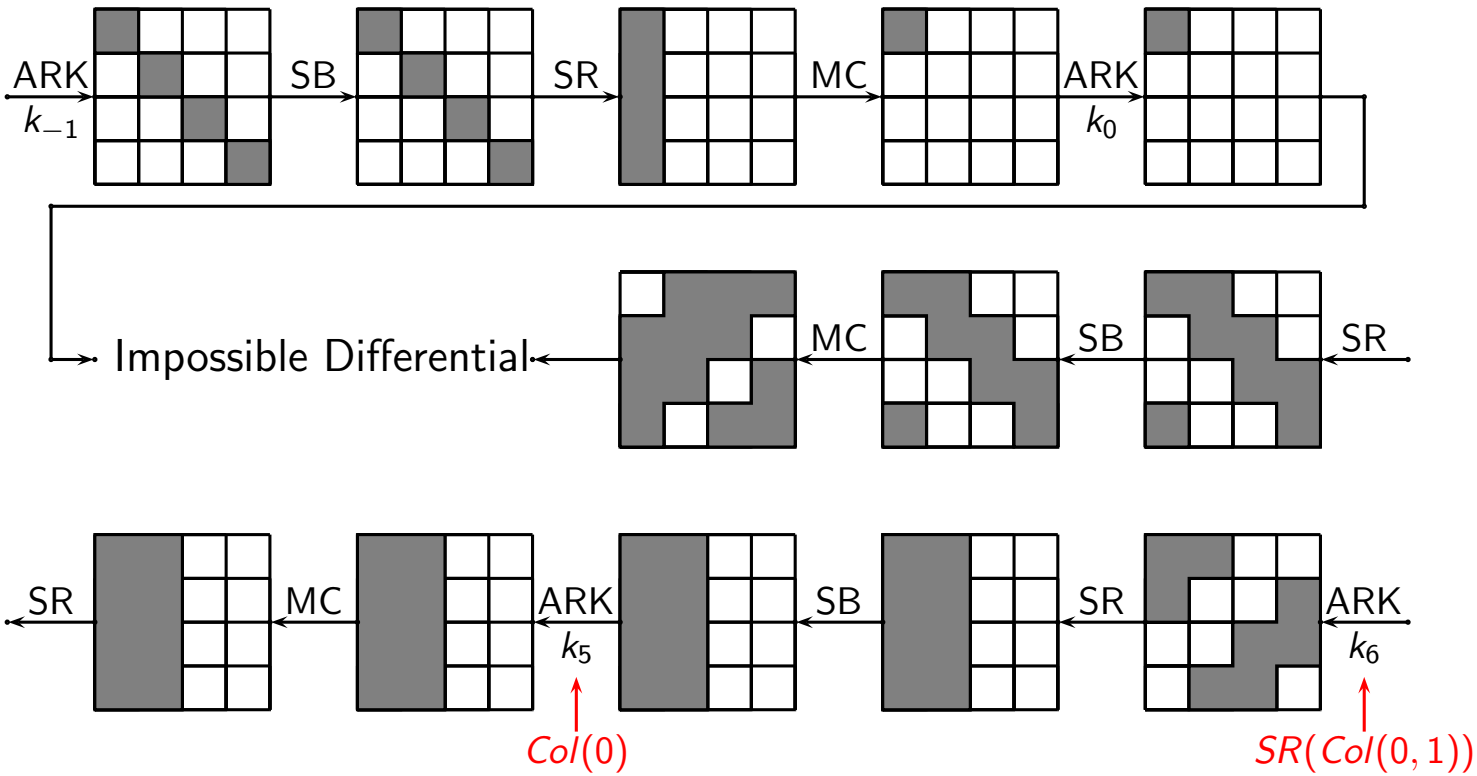
More Tricks

- Using the key schedule to reduce the number of guesses,
- Using different (shifted) columns or activated bytes to generate many impossible differentials from the same data.
- When analyzing a pair, it is sometimes worthwhile to guess a difference then key (and use MC properties, input/output difference relations, etc.)

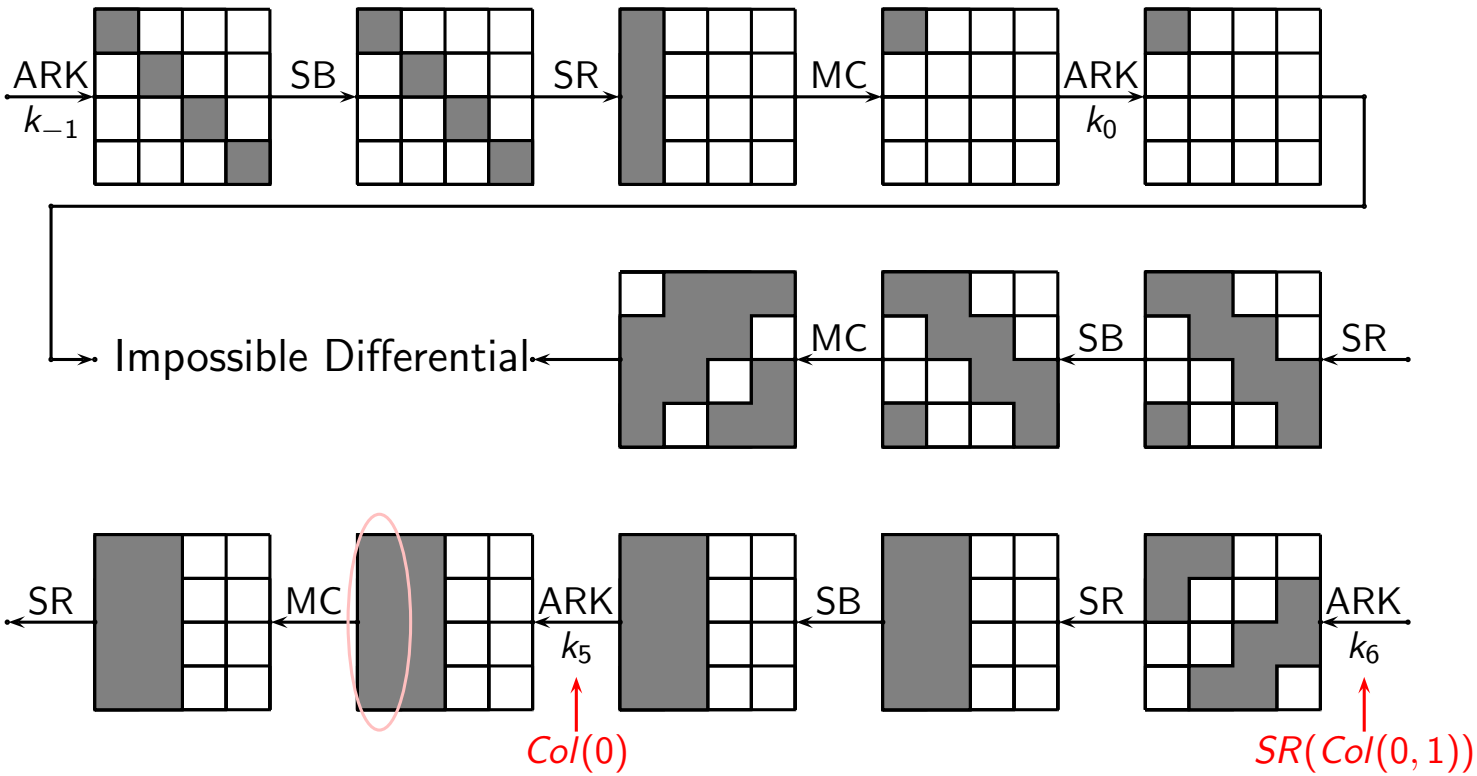
A Better Variant of Phan's Attack [L+]



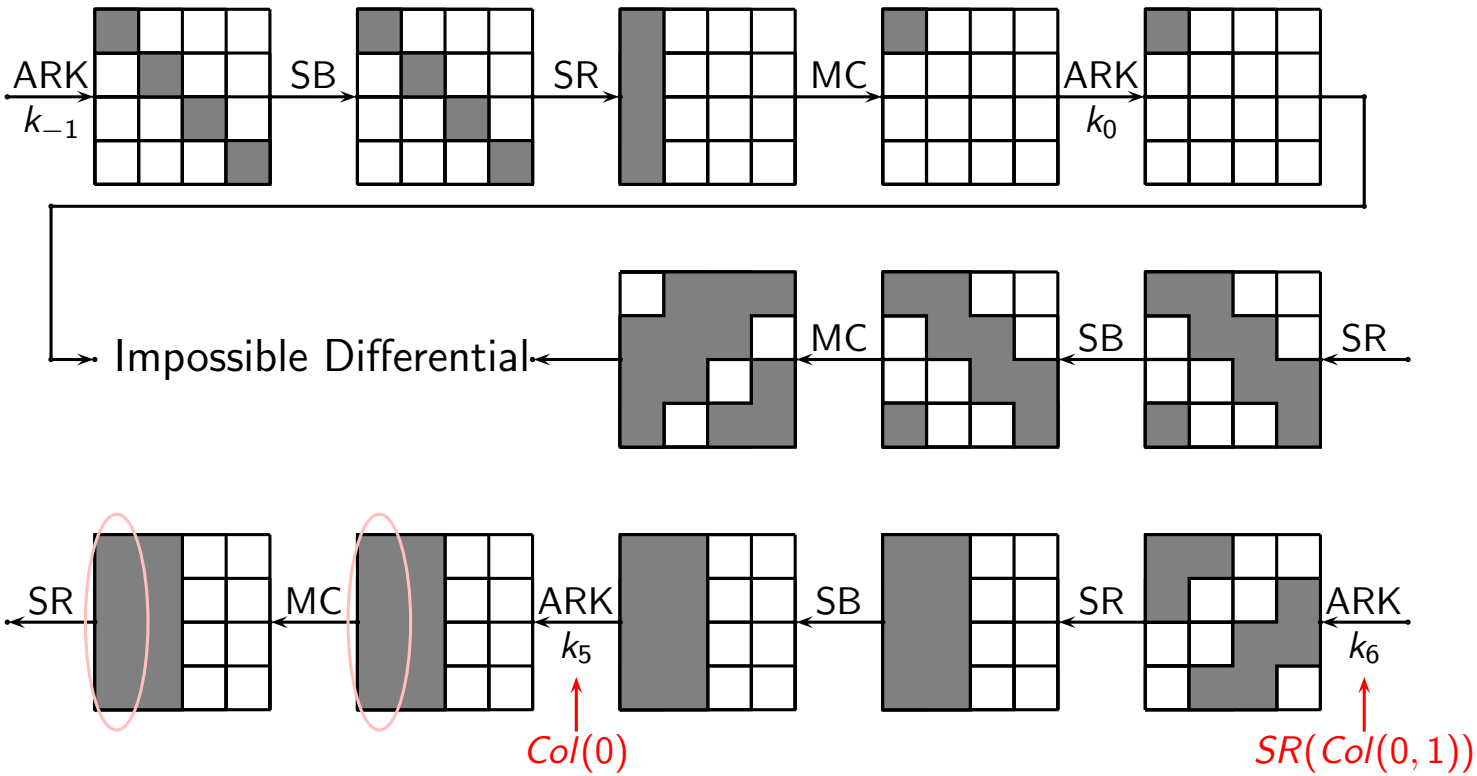
A Better Variant of Phan's Attack [L+]



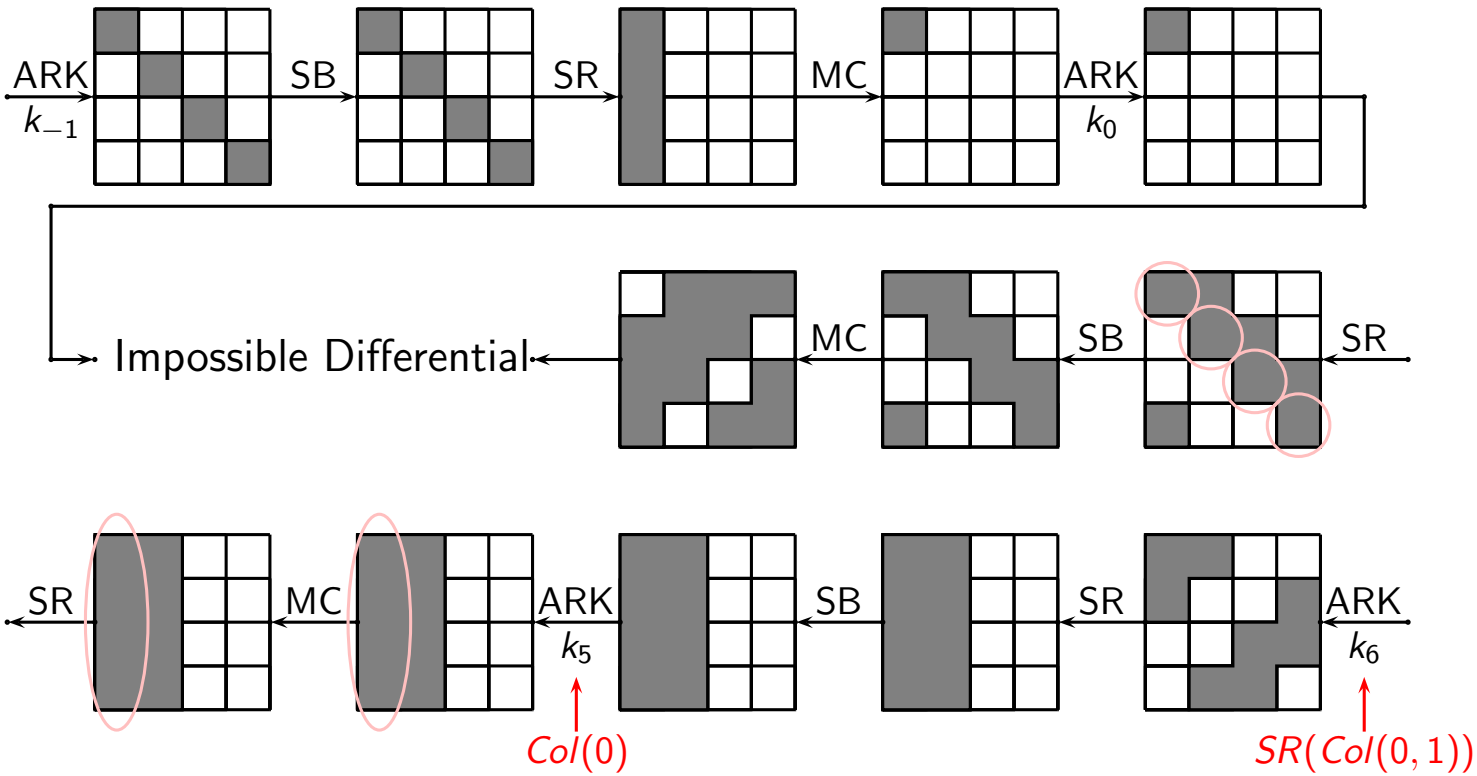
A Better Variant of Phan's Attack [L+]



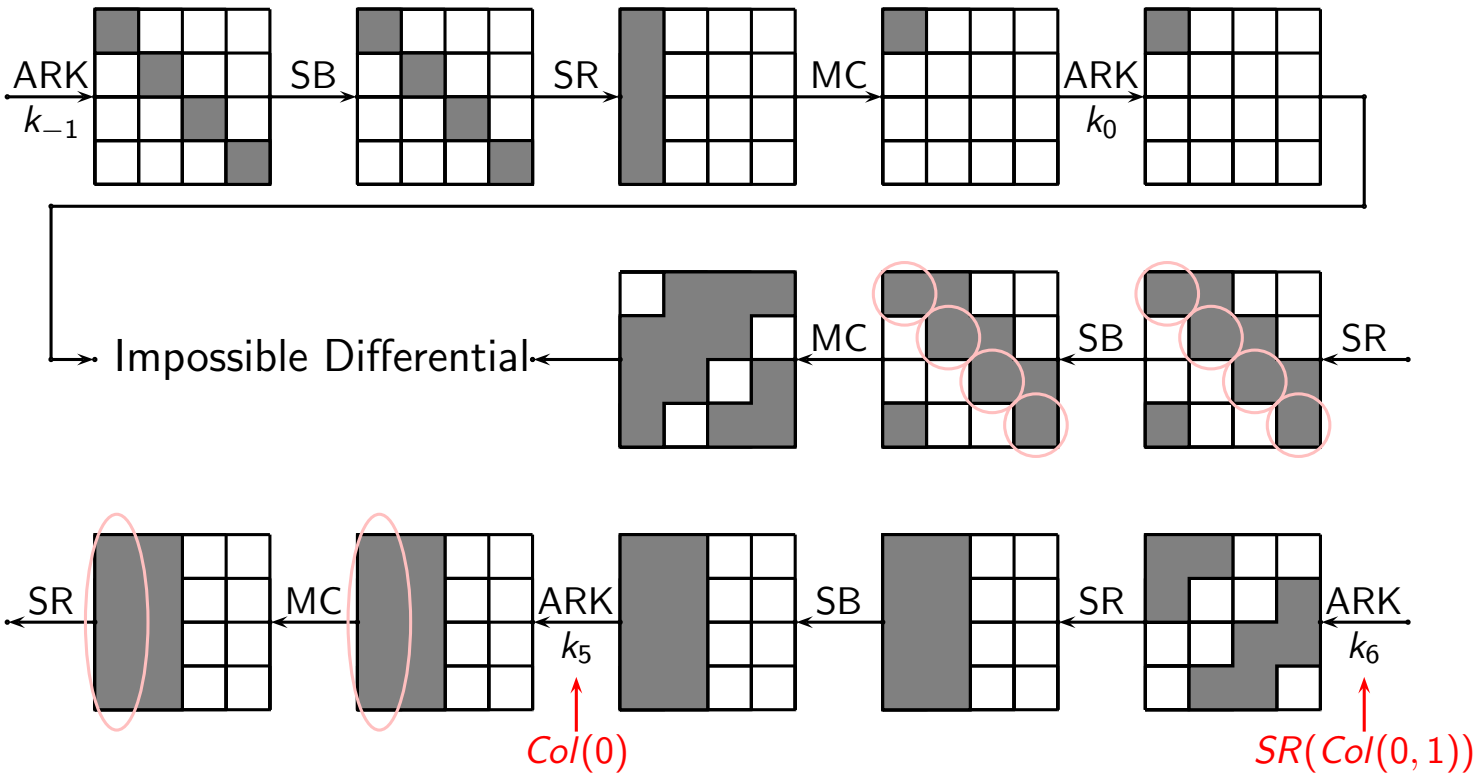
A Better Variant of Phan's Attack [L+]



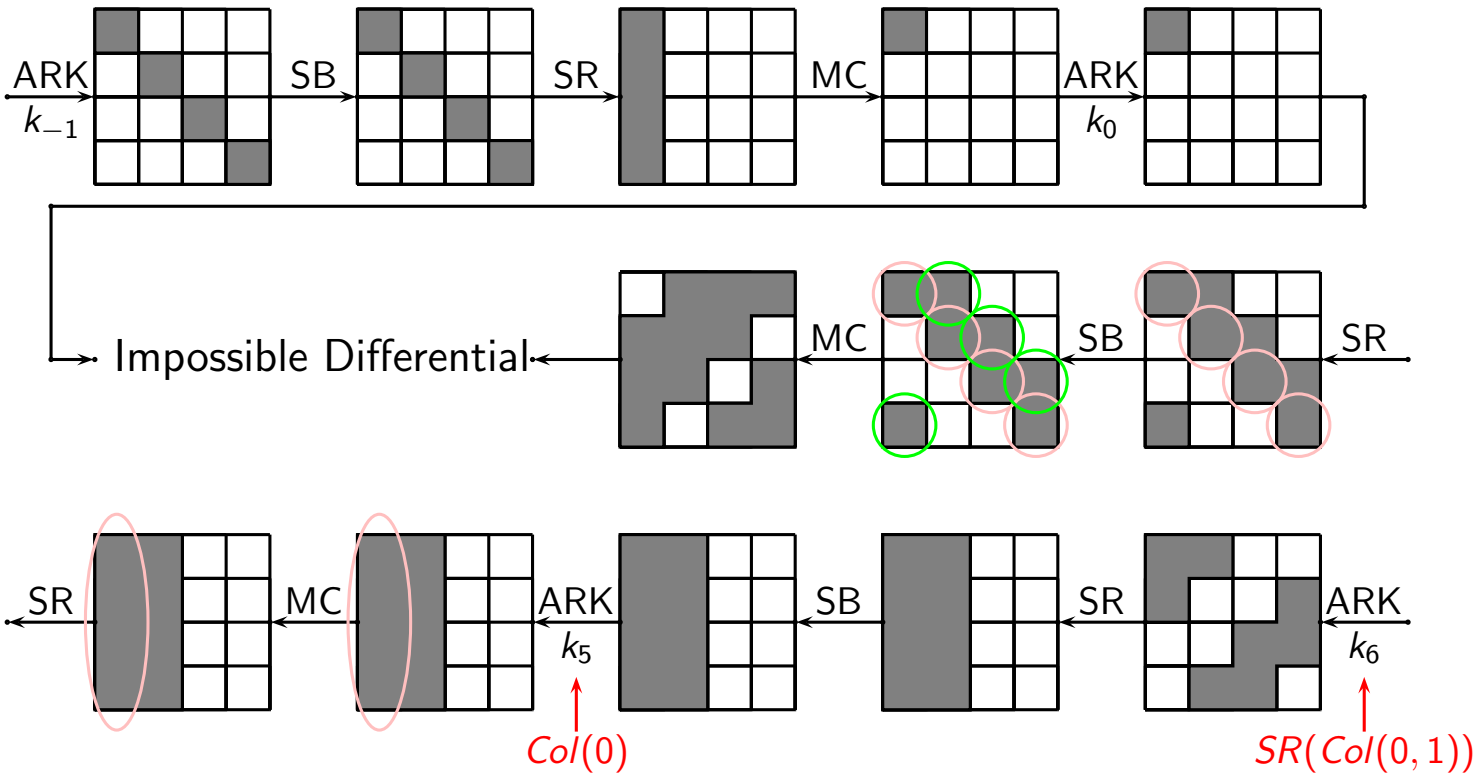
A Better Variant of Phan's Attack [L+]



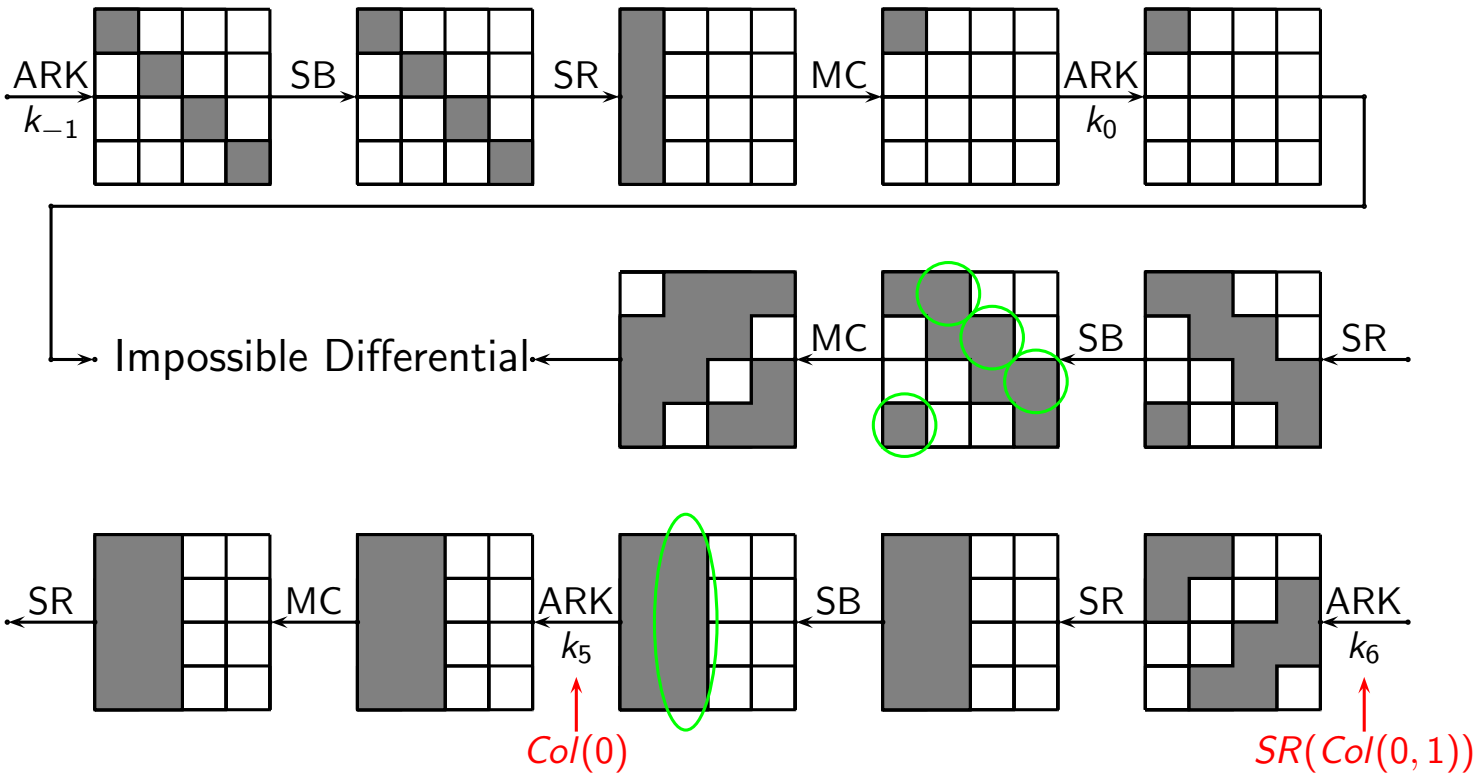
A Better Variant of Phan's Attack [L+]



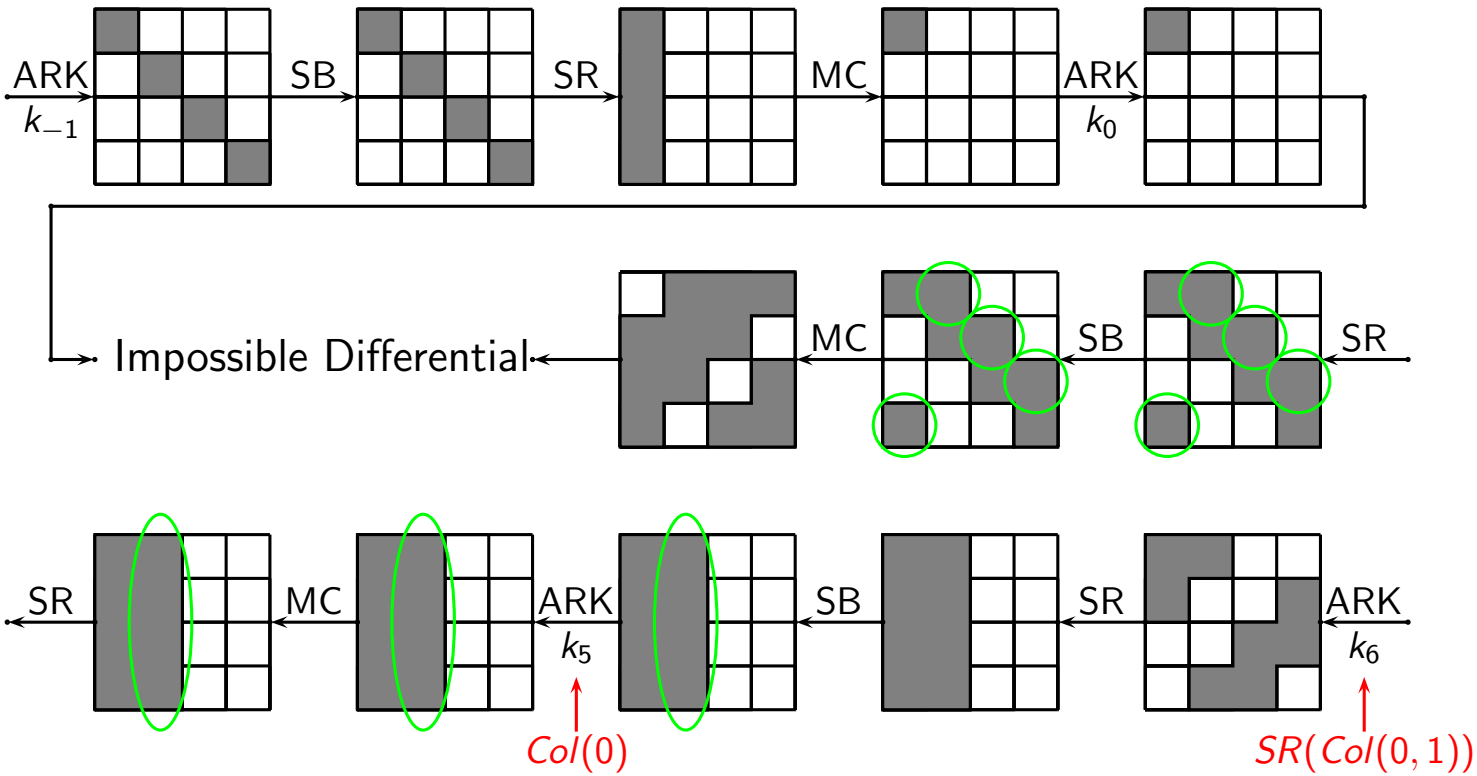
A Better Variant of Phan's Attack [L+]



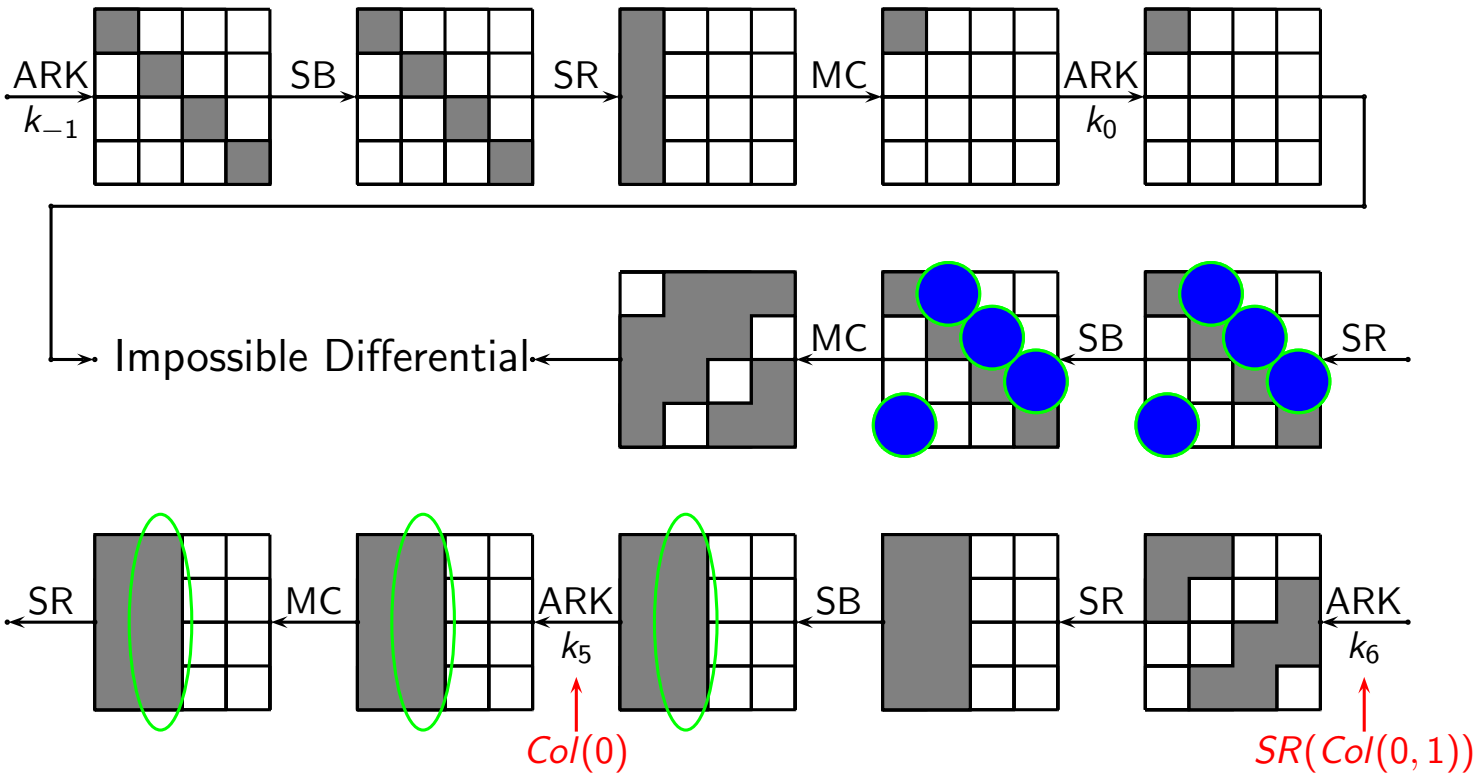
A Better Variant of Phan's Attack [L+]



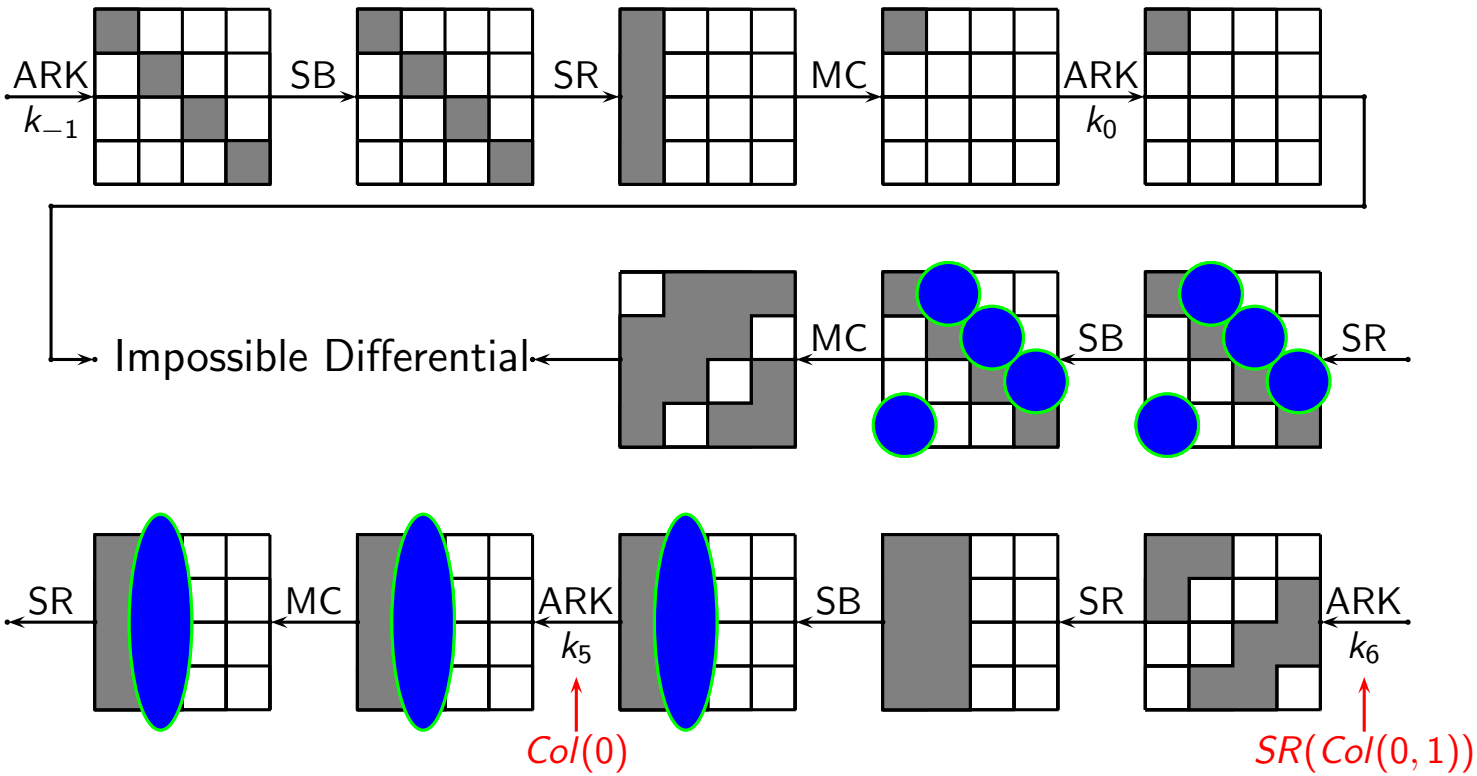
A Better Variant of Phan's Attack [L+]



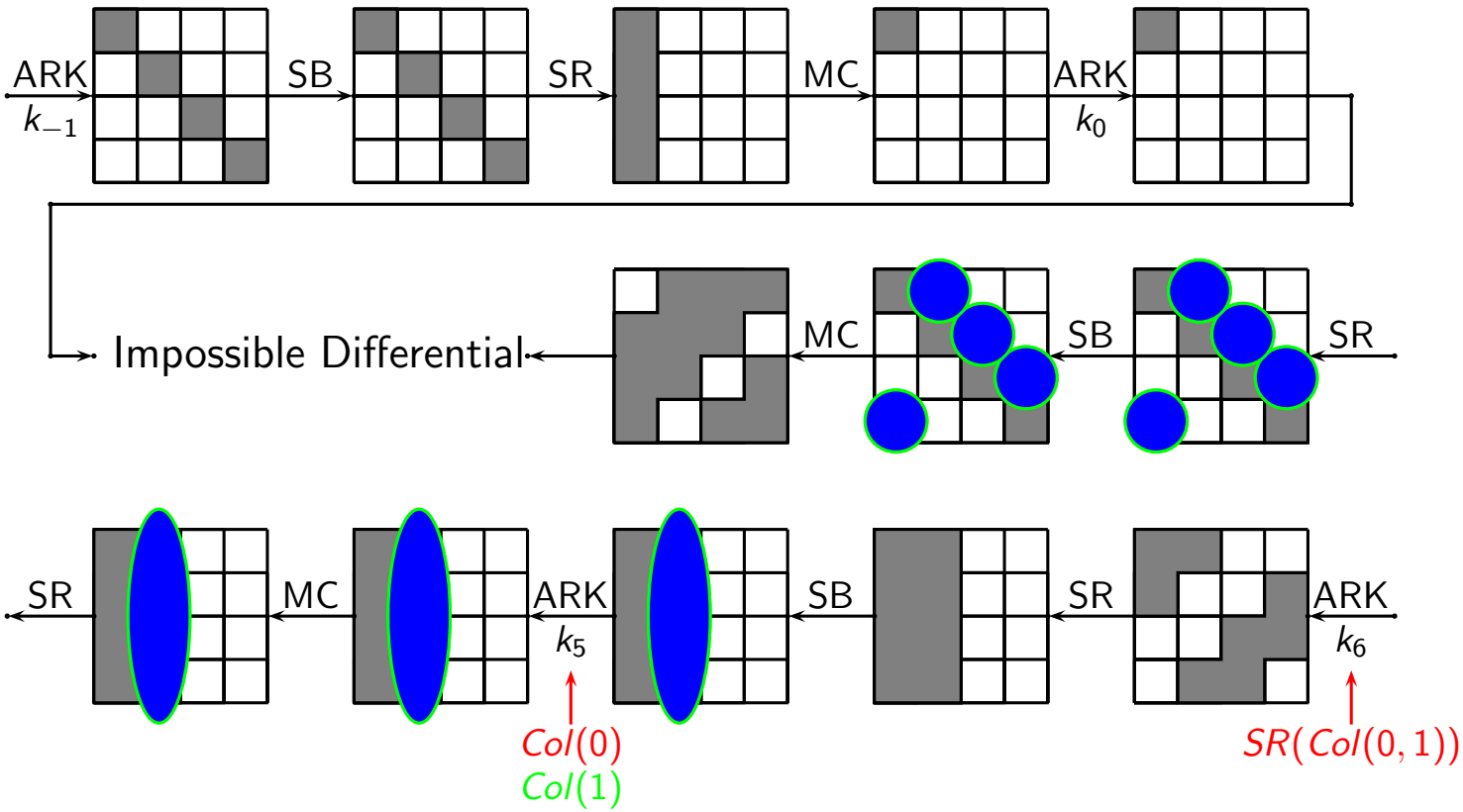
A Better Variant of Phan's Attack [L+]



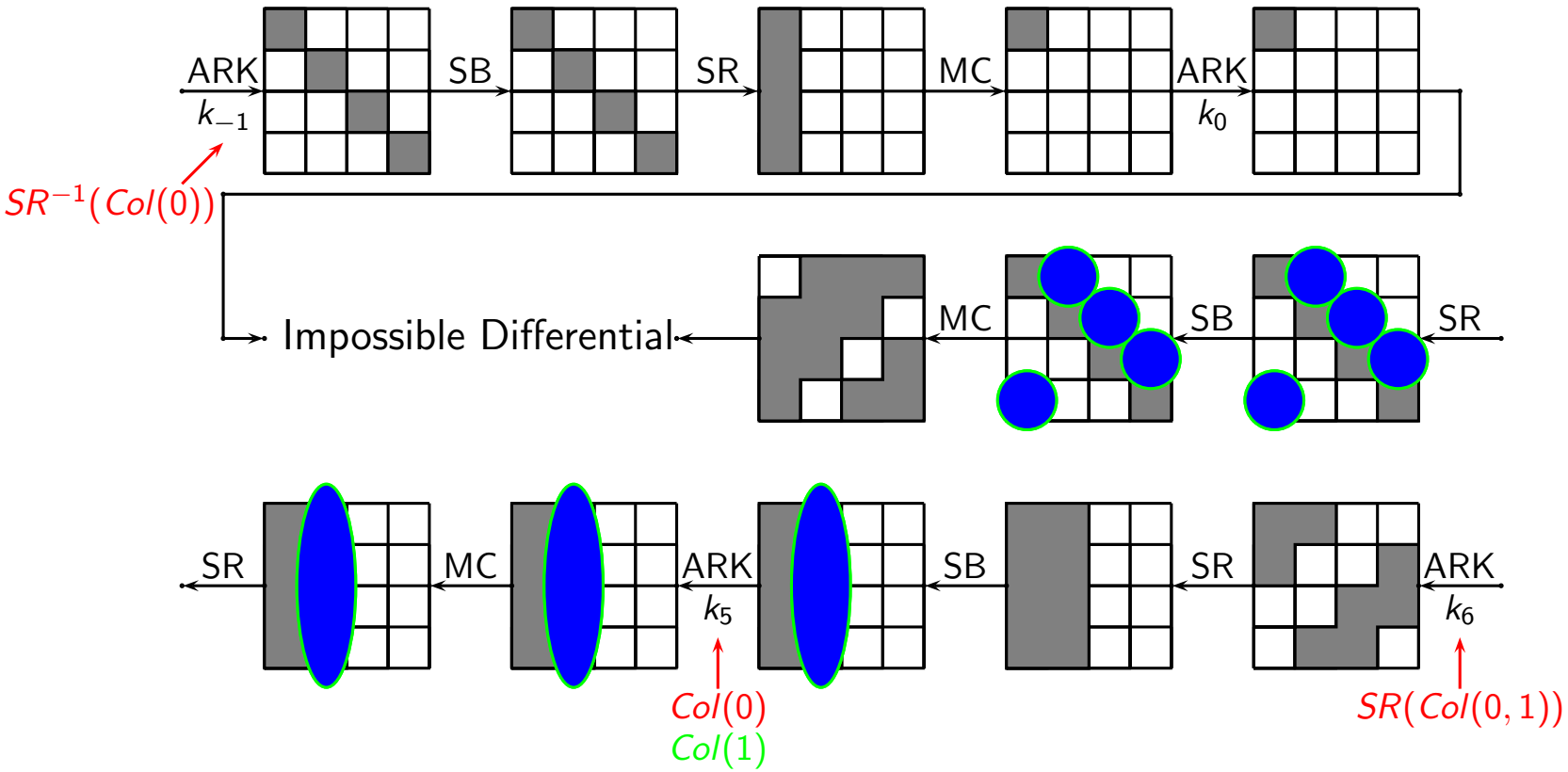
A Better Variant of Phan's Attack [L+]



A Better Variant of Phan's Attack [L+]



A Better Variant of Phan's Attack [L+]

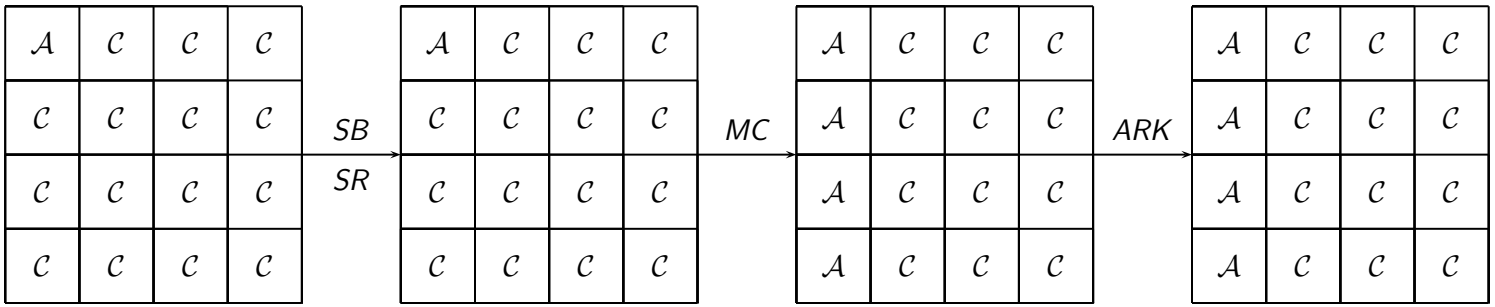


Summary of Impossible Differential Results

Rounds	Key Size	Data	Time	Memory	Source
5	All	$2^{29.5}$	2^{31}	2^{42}	[BK00]
6	All	$2^{91.5}$	2^{122}	2^{93}	[C+01]
7	128	$2^{113.1}$	$2^{113.1}$	$2^{74.1}$	[B+17]
7	128	$2^{112.2}$	$2^{117.2}$ MA	$2^{93.2}$	[L+08]
7	128	2^{105}	$2^{106.9}$	2^{74}	[B+17]
7	192	$2^{113.8}$	$2^{118.8}$ MA		[L+08]
7	256	$2^{113.8}$	$2^{118.8}$ MA		[L+08]
7	192	$2^{91.2}$	$2^{139.2}$		
7	256	2^{92}	2^{163} MA		
8	256	$2^{111.1}$	$2^{227.8}$ MA		
8	256	$2^{89.1}$	$2^{229.7}$ MA		

The Gilbert-Minier Attack

- Consider a set of 2^8 plaintexts of the form:

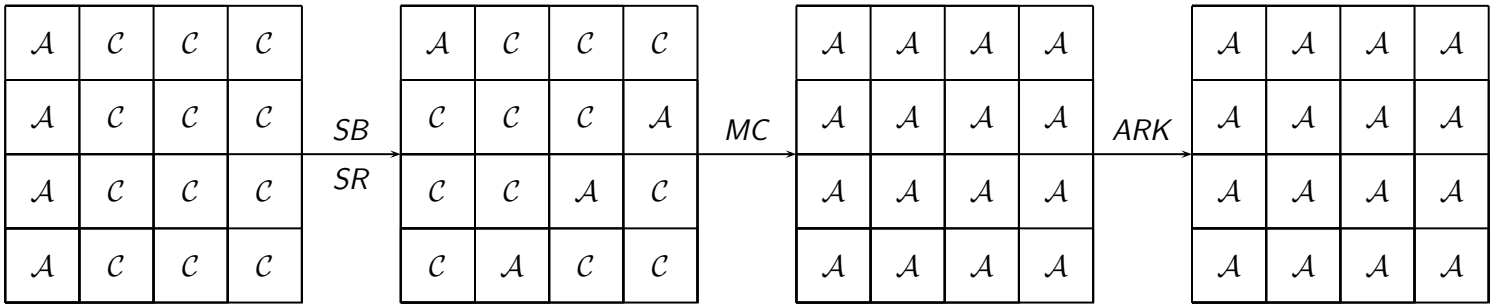


- Consider any non-fixed byte, its value depends on the value of the permuted byte XORed with one constant byte (which depends on the actual plaintext bytes and the key bytes). For example byte 0:

$$\begin{aligned} X_0^{1,i} &= K_0^1 \oplus MC(SB(i), c_0, c_1, c_2) \\ &= K_0^1 \oplus (2 \cdot SB(i) + 3 \cdot c_0 + c_1 + c_2) \\ &= K_0^1 \oplus C \oplus 2 \cdot SB(i) \\ &= C \oplus 2 \cdot SB(i) \end{aligned}$$

The Gilbert-Minier Attack (cont.)

- After another round:

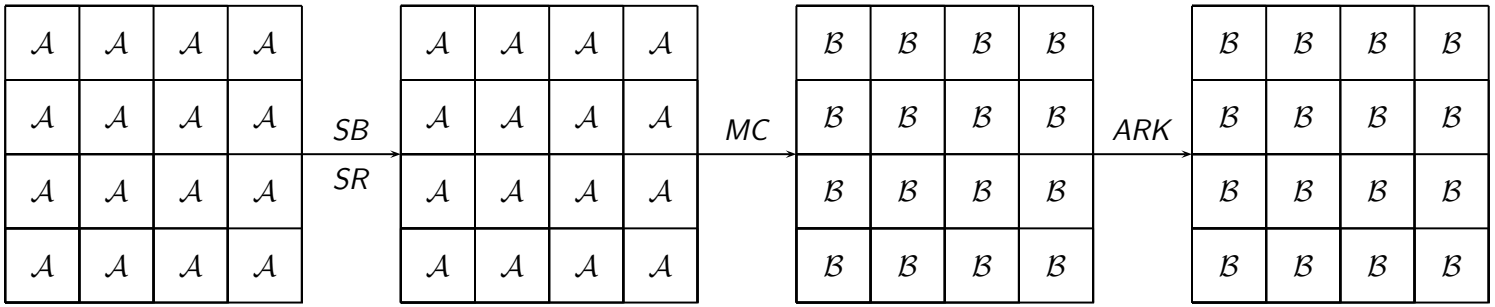


- As before, each byte depends on two constant bytes and the permuted byte:

$$\begin{aligned} X_0^{2,i} &= K_0^2 \oplus MC(SB(2 \cdot SB(i) \oplus C)), c_0, c_1, c_2) \\ &= K_0^2 \oplus (2 \cdot SB(2 \cdot SB(i) \oplus C) + 3 \cdot c_0 + c_1 + c_2) \\ &= K_0^2 \oplus C' \oplus 2 \cdot SB(2 \cdot SB(i) \oplus C) = C'' \oplus 2 \cdot SB(2 \cdot SB(i \end{aligned}$$

The Gilbert-Minier Attack (cont.)

- After the third round:



- After this round, each byte depends on the order of the permuted byte and 9 constants (which depend on the key and the constant plaintexts).

The Gilbert-Minier Attack (cont.)

- In [GM00] the fact that each byte can be written as a function of 9 constants was used to offer a collision attack (where sets of plaintexts for which the constants collide are searched for).
- Using 2^{32} chosen plaintexts it is possible to attack 7-round AES with time complexity of $2^{128}/2^{140}/2^{140}$.
- The attack is based on partial encryption of the set of plaintexts (one round), and partial decryption of two rounds, to identify collisions in the sets.

The Demirci-Selçuk Attack

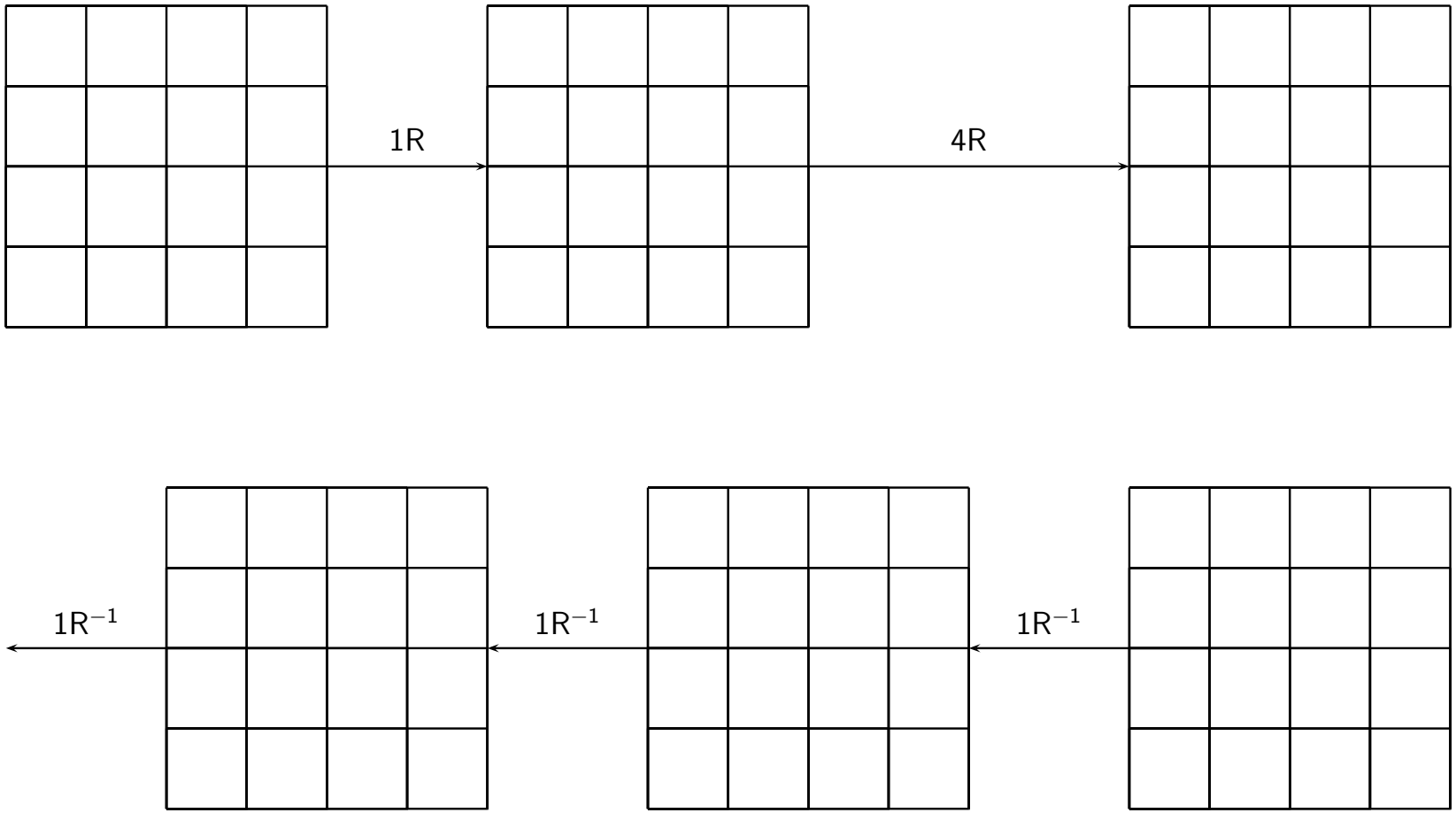
- In [DS08], Demirci and Selçuk tried to extend the basic Gilbert-Minier property by one extra round.
- A simple extension yields that every byte depends on 33 constants, but luckily, some of them are redundant, and the total number of constants is 25.
- In other words, every byte after four rounds can be written as

$$f_{c_1,c_2,\ldots,c_{25}}(i).$$

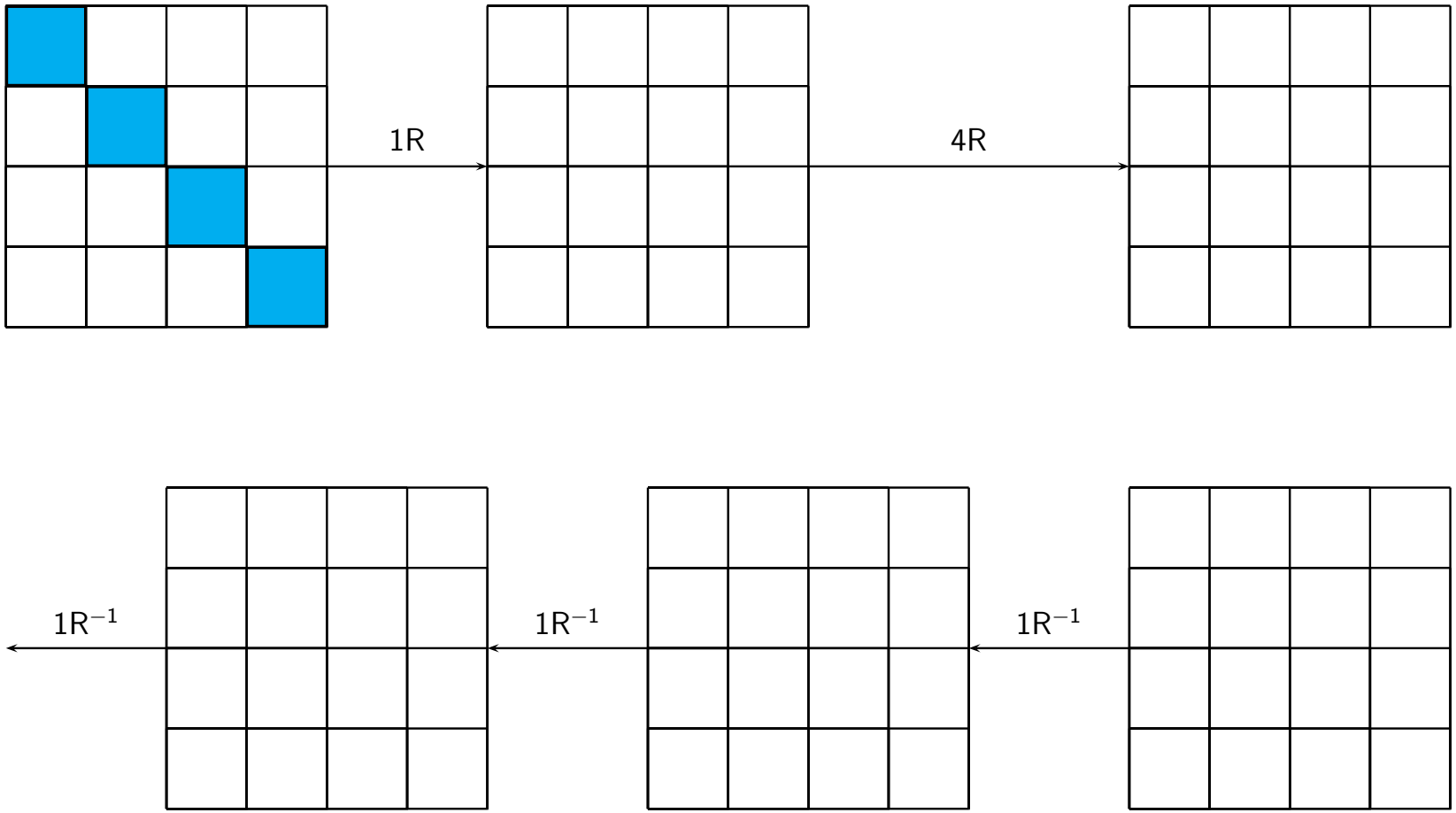
The Demirci-Selçuk Attack (cont.)

- The attack is based on a precomputation that computes $f_{c_1,\ldots,c_{25}}(i)$ for all 2^{200} constants and all i's, thus generating for each set of constants the sequence of admissible values.
- In the online phase, a partial decryption is done, to check whether the set of ciphertexts indeed generates the admissible sequence. If not, then the key guess is wrong.

The Demirci-Selçuk Attack (cont.)

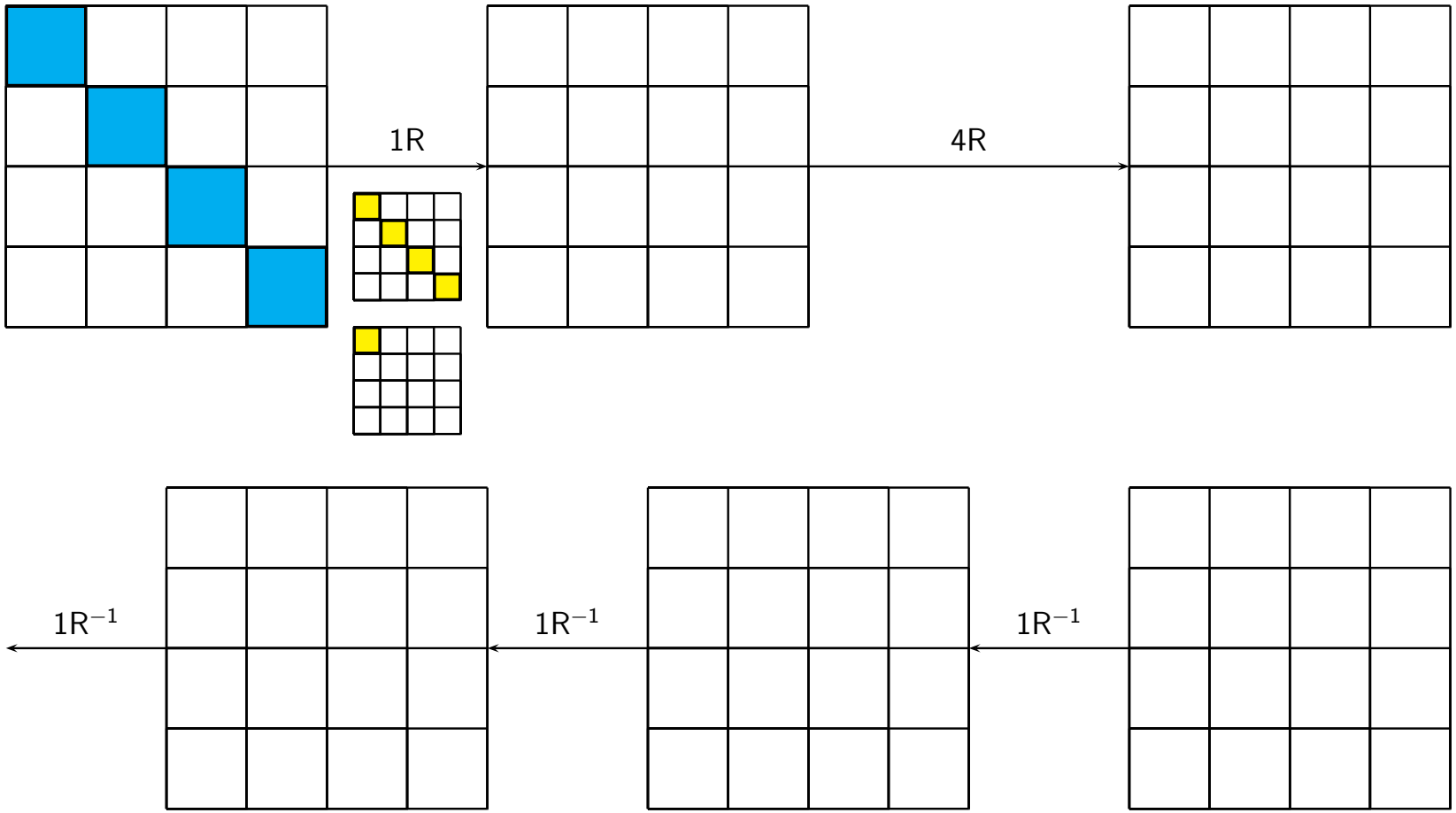


The Demirci-Selçuk Attack (cont.)



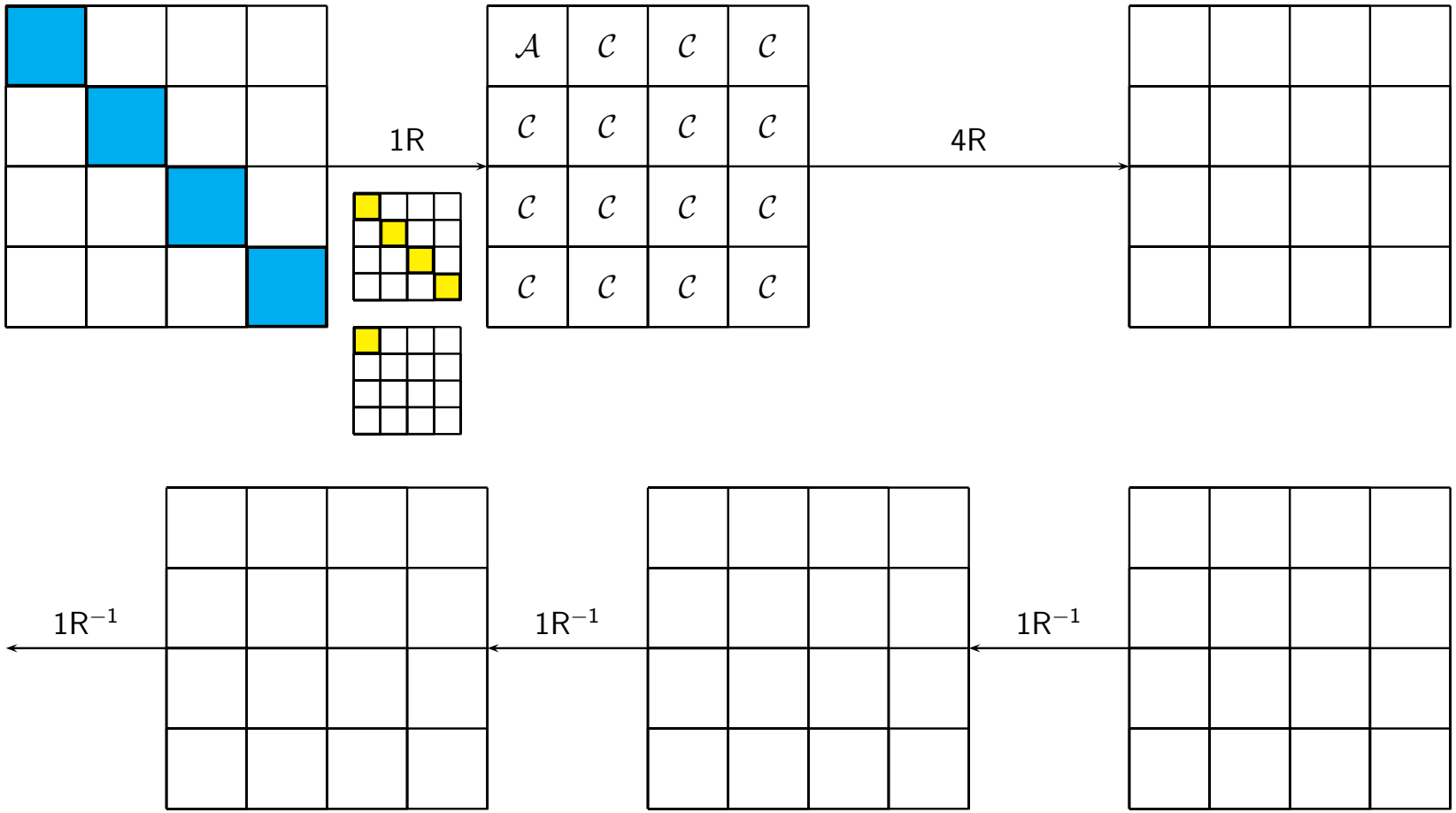
Take a structure of 2^{32} plaintexts.

The Demirci-Selçuk Attack (cont.)



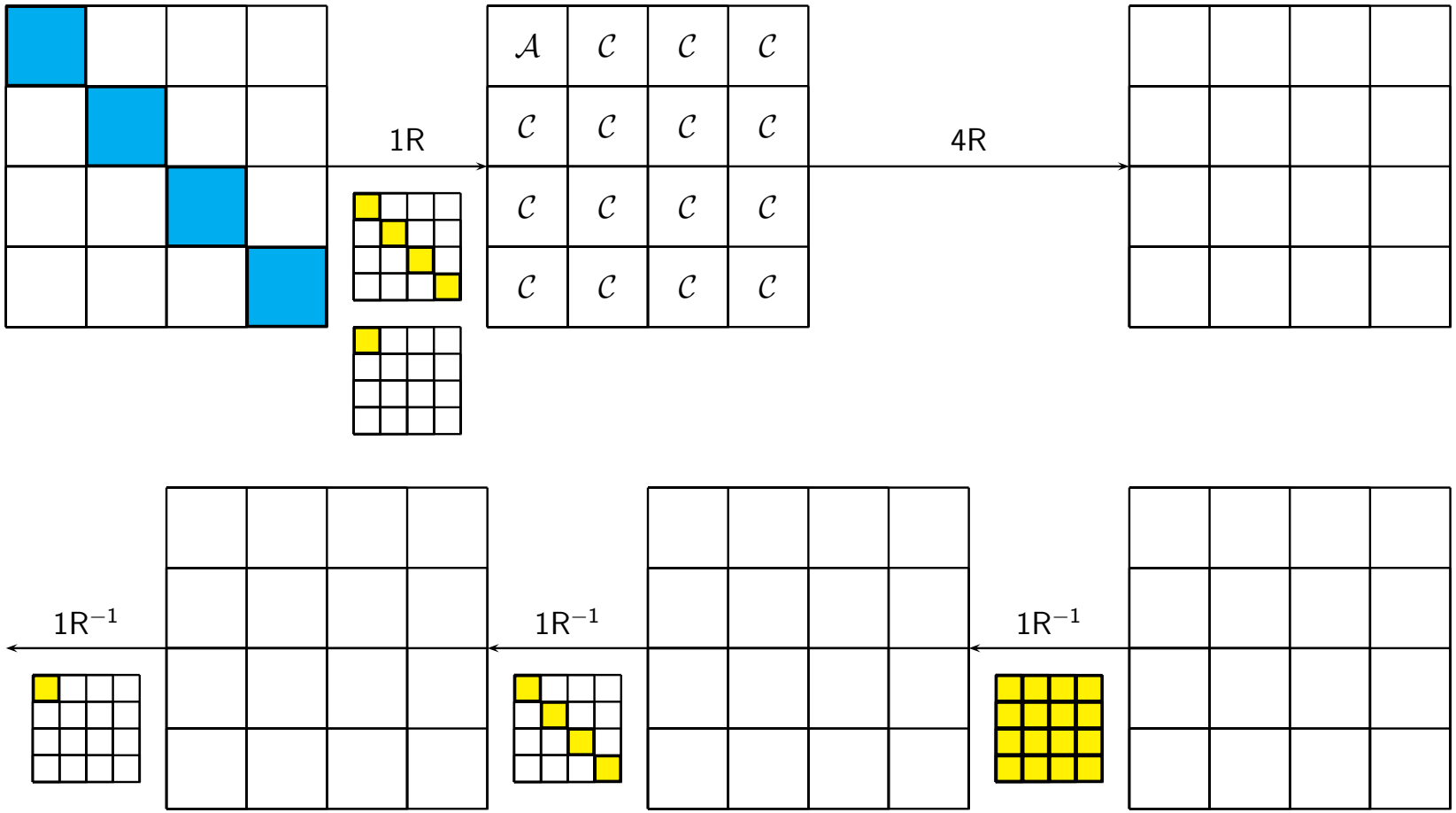
Guess 4 bytes of K^0 and one byte of K^1.

The Demirci-Selçuk Attack (cont.)



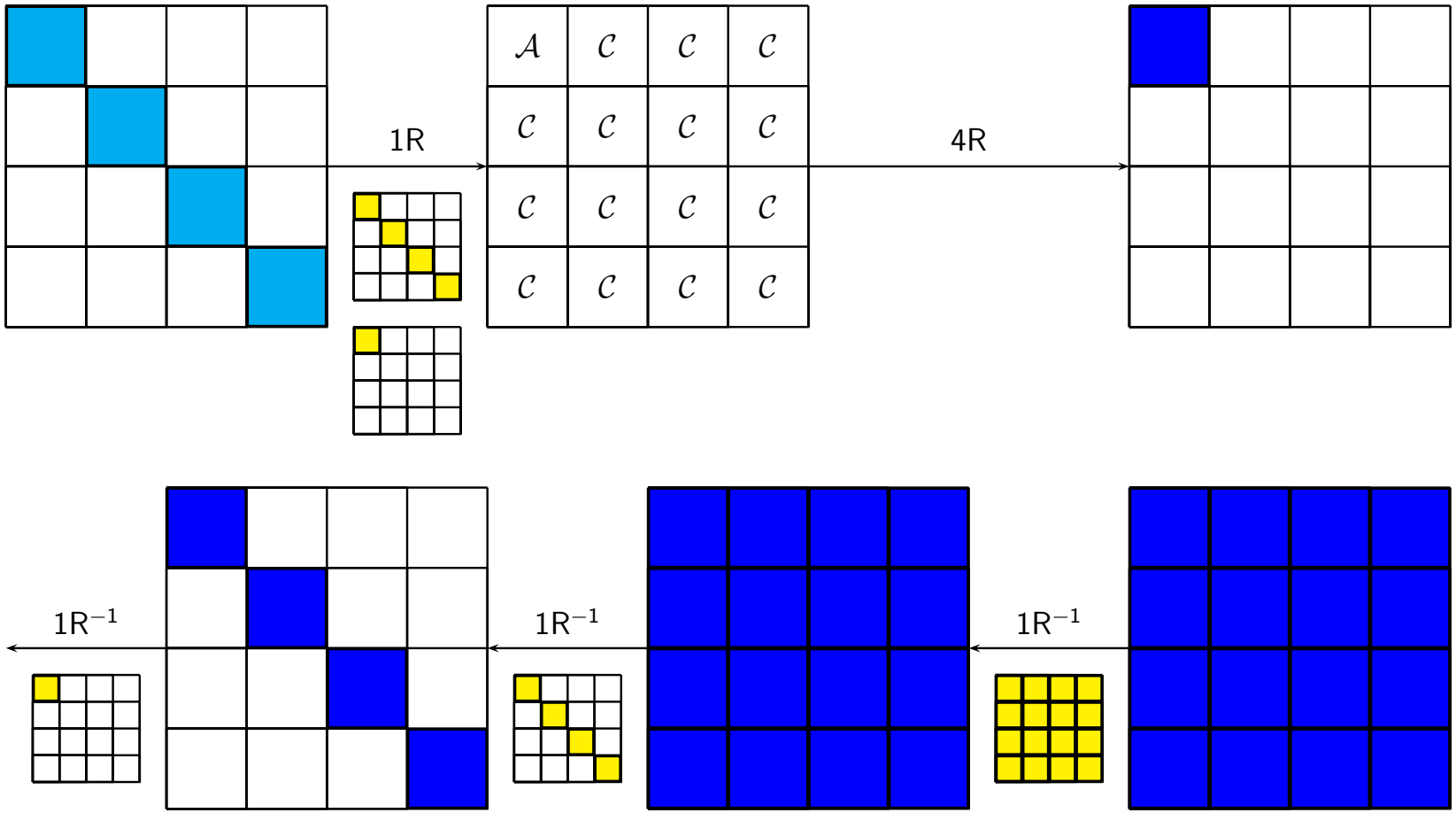
Identify a structure of 256 plaintexts.

The Demirci-Selçuk Attack (cont.)



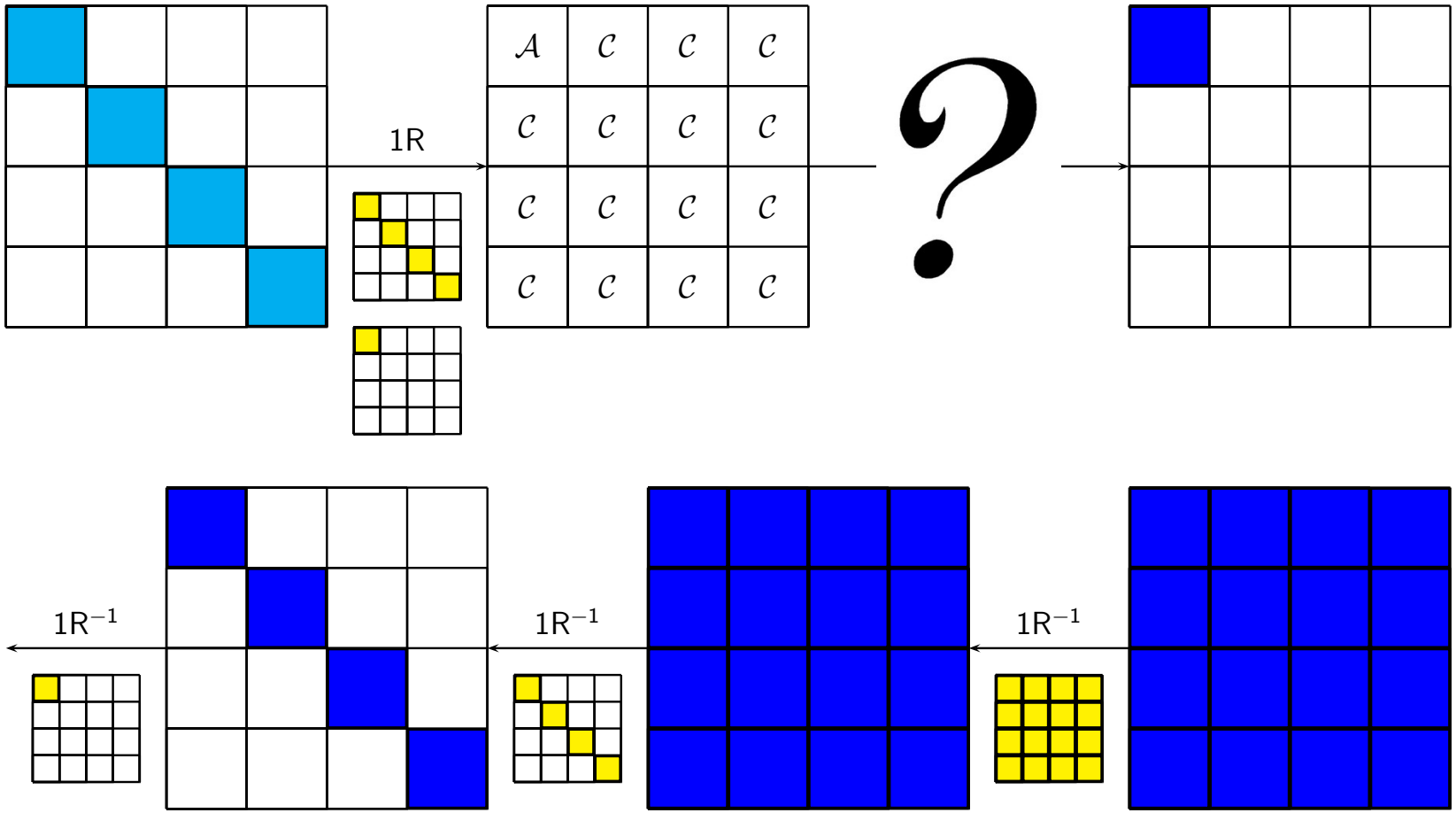
Guess 16 bytes of K^8, 4 of K^7, and one of K^6.

The Demirci-Selçuk Attack (cont.)



Partially decrypt the ciphertext of the structure.

The Demirci-Selçuk Attack (cont.)



Check if the sequence is valid.

The Demirci-Selçuk Attack (cont.)

- There is a precomputation phase of $2^{200} \cdot 2^8$ operations, and memory consumption for storing 2^{200} sequences.
- One can reduce the precomputation and memory requirements, by covering a fraction ϵ of the possible sequences.
- In exchange, the data complexity is increased by a factor $1/\epsilon$.
- The time complexity increases by the same amount of work, which results in a tradeoff:

Version	Rounds	Data	Memory (Pre)	Time	MinMax
AES-192	7	$\mathbf{2}^{46+n}$	$\mathbf{2}^{192-n}$	$\mathbf{2}^{94+n}$	2^{143}
AES-256	7	$\mathbf{2}^{34+n}$	$\mathbf{2}^{204-n}$	$\mathbf{2}^{82+n}$	$\mathbf{2}^{143}$
AES-256	8	$\mathbf{2}^{34+n}$	$\mathbf{2}^{206-n}$	$\mathbf{2}^{205.6+n}$	$\mathbf{2}^{205.8}$

Final Comments on Demirci-Selçuk's Attack

- Actually, starting with a structure of 2^{32} plaintexts offers 2^{24} structures of 256 plaintexts of the required form. Hence, it is possible to save 2^{24} in the data complexity in exchange for memory (the time still costs).
- There is also a way to reduce the size of the table by a factor of 256 entries as

$$f_{c_1,c_2,\ldots,c_{25}}(i) = g_{c_1,c_2,\ldots,c_{24}}(i) \oplus c_{25}$$

so by looking at $f'(i) = f(i) \oplus f(0)$ reduces the number of constants.

Some Tricks [DKS10]

- Multiset tabulation — One can save a constant, by recalling that one of the plaintexts will have $f(0) = 0$, and using it.
- Differential enumeration — use differentials that when they are satisfied reveal a lot of information about the internal state — less constants! (but more data — we need a right pair in the data).
- For AES-192 — one can use key relations over 8 rounds!

Summary of Collision-Based Attacks

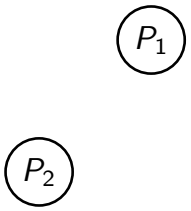
Rounds	Key Size	Data	Time	Memory	Source
7	All	2^{116}	2^{116}	2^{116}	[DKS10]
7	128	2^{105}	2^{99}	2^{90}	[DFJ13]
7	128	2^{97}	2^{99}	2^{98}	[DFJ13]
7	192 & 256	2^{99}	2^{99}	2^{96}	[DFJ13]
8	192	2^{113}	2^{172}	2^{129}	[DKS10]
8	192	2^{113}	2^{172}	2^{82}	[DFJ13]
8	192	2^{107}	2^{172}	2^{96}	[DFJ13]
8	256	2^{113}	2^{196}	2^{129}	[DKS10]
8	256	2^{113}	2^{196}	2^{82}	[DFJ13]
8	256	2^{107}	2^{196}	2^{96}	[DFJ13]
9	256	2^{120}	2^{203}	2^{203}	[DFJ13]

The Boomerang Attack

- Introduced by [W99].
- Targets ciphers with good short differentials, but bad long ones.
- The core idea: Treat the cipher as a cascade of two sub-ciphers. Where in the first sub-cipher a differential $\alpha \xrightarrow{E_0} \beta$ exists, and a differential $\gamma \xrightarrow{E_1} \delta$ exists for the second.

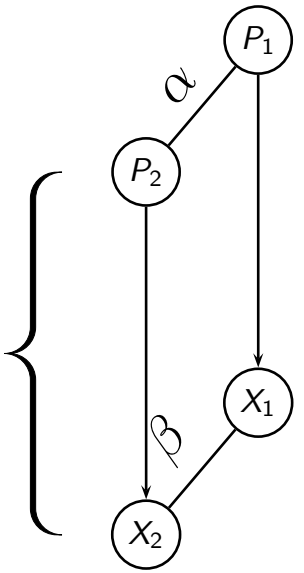
The Boomerang Attack

- Introduced by [W99].
- Targets ciphers with good short differentials, but bad long ones.
- The core idea: Treat the cipher as a cascade of two sub-ciphers. Where in the first sub-cipher a differential $\alpha \xrightarrow{E_0} \beta$ exists, and a differential $\gamma \xrightarrow{E_1} \delta$ exists for the second.
- The process starts with a pair of plaintexts: $P_1, P_2 = P_1 \oplus \alpha$.



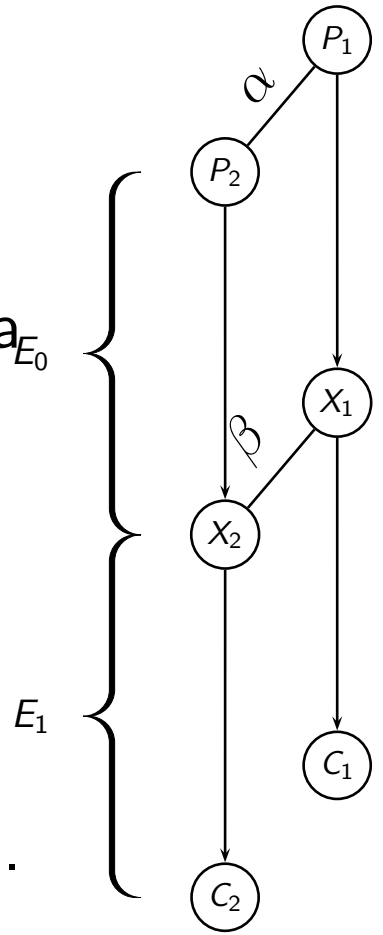
The Boomerang Attack

- Introduced by [W99].
- Targets ciphers with good short differentials, but bad long ones.
- The core idea: Treat the cipher as a cascade of two sub-ciphers. Where in the first sub-cipher a differential $\alpha \xrightarrow{E_0} \beta$ exists, and a differential $\gamma \xrightarrow{E_1} \delta$ exists for the second.
- After the first sub-cipher, $X_1 \oplus X_2 = \beta$.



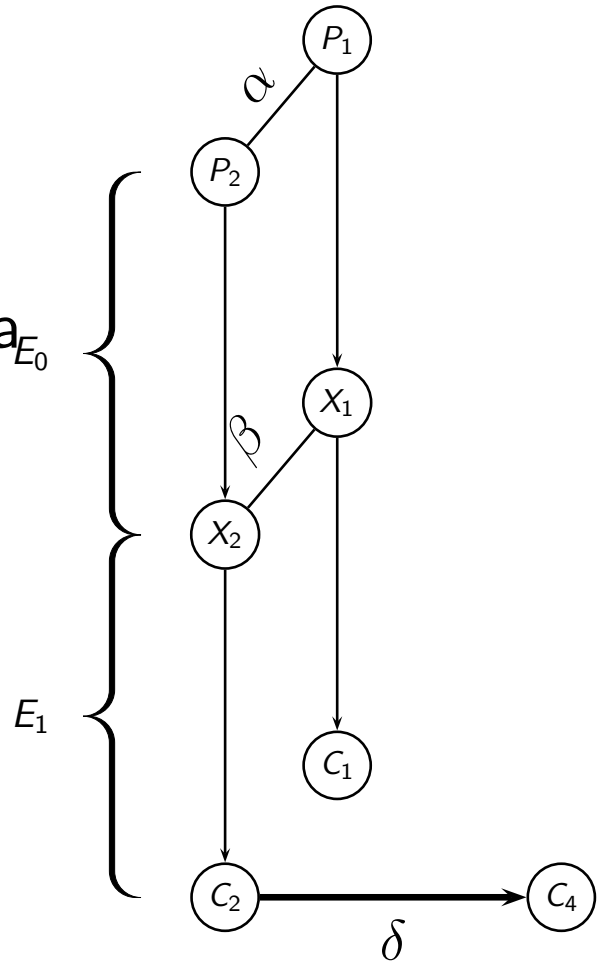
The Boomerang Attack

- Introduced by [W99].
- Targets ciphers with good short differentials, but bad long ones.
- The core idea: Treat the cipher as a cascade of two sub-ciphers. Where in the first sub-cipher a differential $\alpha \xrightarrow{E_0} \beta$ exists, and a differential $\gamma \xrightarrow{E_1} \delta$ exists for the second.
- But the encryption process continues and we obtain C_1 and C_2.



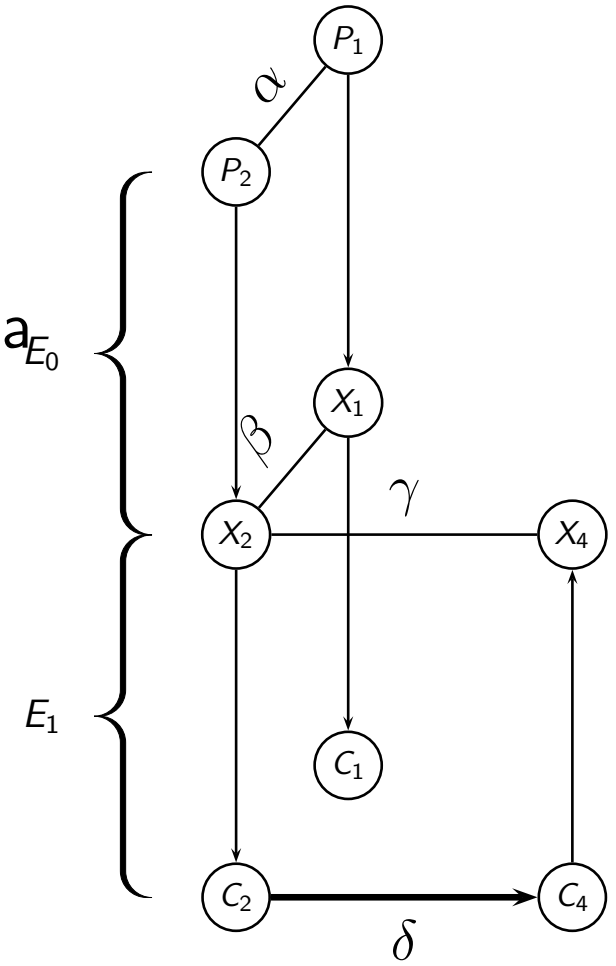
The Boomerang Attack

- Introduced by [W99].
- Targets ciphers with good short differentials, but bad long ones.
- The core idea: Treat the cipher as a cascade of two sub-ciphers. Where in the first sub-cipher a differential $\alpha \xrightarrow{E_0} \beta$ exists, and a differential $\gamma \xrightarrow{E_1} \delta$ exists for the second.
- From C_2 we compute $C_4 = C_2 \oplus \delta$.



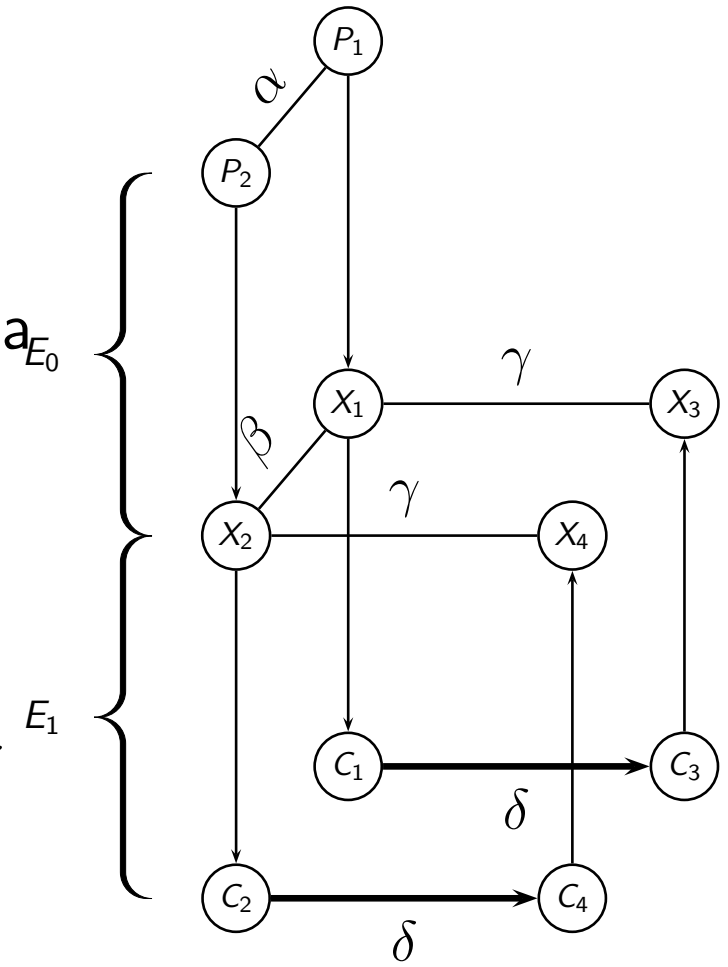
The Boomerang Attack

- Introduced by [W99].
- Targets ciphers with good short differentials, but bad long ones.
- The core idea: Treat the cipher as a cascade of two sub-ciphers. Where in the first sub-cipher a differential $\alpha \xrightarrow{E_0} \beta$ exists, and a differential $\gamma \xrightarrow{E_1} \delta$ exists for the second.
- Which is partially decrypted to X_4, for which $X_4 \oplus X_2 = \gamma$.



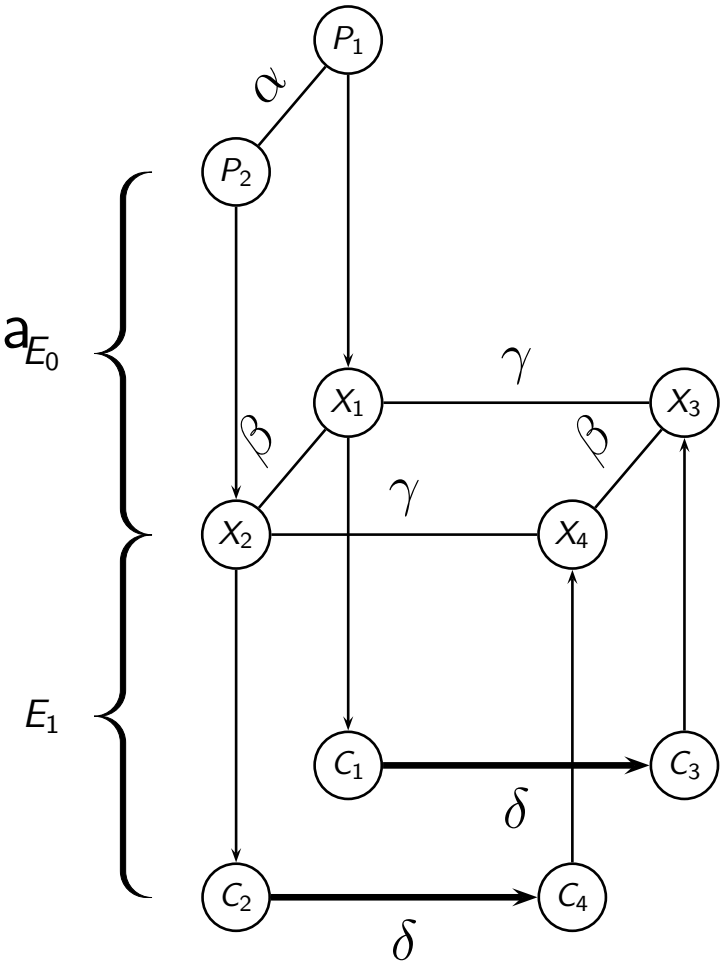
The Boomerang Attack

- Introduced by [W99].
- Targets ciphers with good short differentials, but bad long ones.
- The core idea: Treat the cipher as a cascade of two sub-ciphers. Where in the first sub-cipher a differential $\alpha \xrightarrow{E_0} \beta$ exists, and a differential $\gamma \xrightarrow{E_1} \delta$ exists for the second.
- Similarly, we compute $C_3 = C_1 \oplus \delta$ and obtain $X_3 \oplus X_1 = \gamma$.



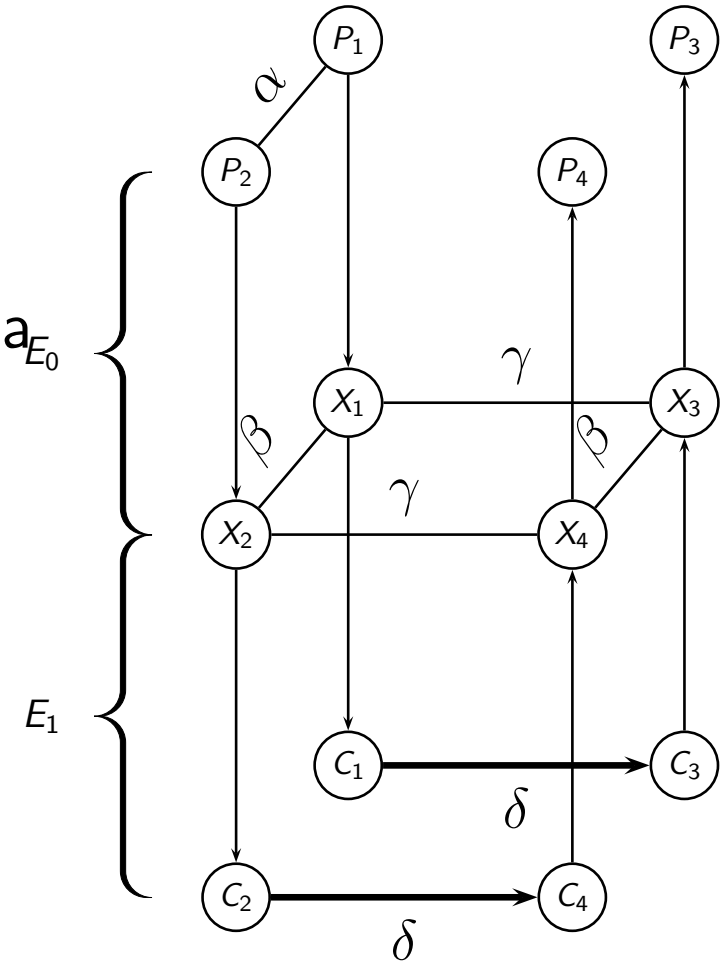
The Boomerang Attack

- Introduced by [W99].
- Targets ciphers with good short differentials, but bad long ones.
- The core idea: Treat the cipher as a cascade of two sub-ciphers. Where in the first sub-cipher a differential $\alpha \xrightarrow{E_0} \beta$ exists, and a differential $\gamma \xrightarrow{E_1} \delta$ exists for the second.
- Hence, we obtained $X_3 \oplus X_4 = \beta$.



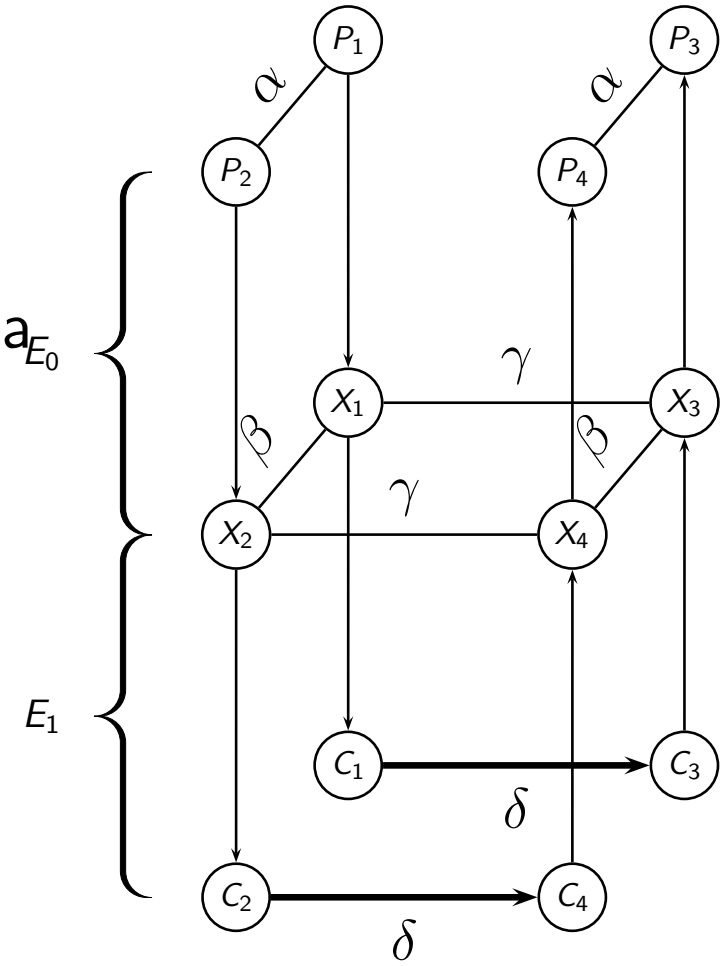
The Boomerang Attack

- Introduced by [W99].
- Targets ciphers with good short differentials, but bad long ones.
- The core idea: Treat the cipher as a cascade of two sub-ciphers. Where in the first sub-cipher a differential $\alpha \xrightarrow{E_0} \beta$ exists, and a differential $\gamma \xrightarrow{E_1} \delta$ exists for the second.
- The decryption process continues, and we obtain P_3 and P_4.

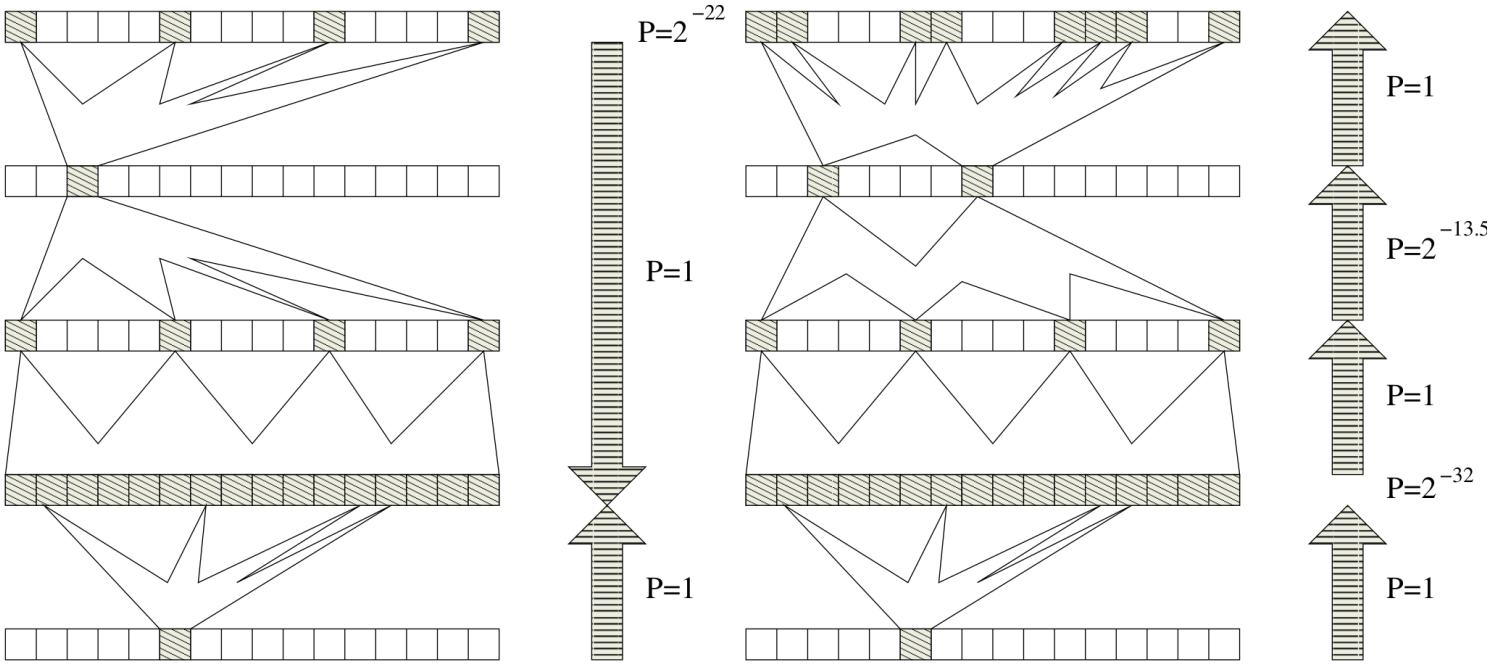


The Boomerang Attack

- Introduced by [W99].
- Targets ciphers with good short differentials, but bad long ones.
- The core idea: Treat the cipher as a cascade of two sub-ciphers. Where in the first sub-cipher a differential $\alpha \xrightarrow{E_0} \beta$ exists, and a differential $\gamma \xrightarrow{E_1} \delta$ exists for the second.
- Which by the first differential satisfy $P_3 \oplus P_4 = \alpha$.



First Boomerang Attack [B04]



First Boomerang Attack (cont.)

- Add one round at the top.
- Data generation is the "standard" 2^{32} set of plaintexts.
- After applying the boomerang process, there are still 8 bytes with zero difference in the plaintext.
- Almost immediate detection of the right boomerang.
- For 6-round attack — guess 4 bytes of the key at the *end* and ask for their decryption.

Rounds	Key Size	Data (ACPC)	Time	Memory
5	All	2^{39}	2^{39}	2^{33}
6	All	2^{71}	2^{71}	2^{33}

Mixture Differentials [G18] (but related to [G+17])

- Consider two states at round i — x_i and y_i.
- Assume that $x_{i,Col(1,2,3)} = y_{i,Col(1,2,3)}$.
- Let z_i and w_i be a *mixture* of them, i.e.,

$$z_i = \begin{cases} x_i & \text{if } i \notin \mathcal{I} \\ y_i & \text{if } i \in \mathcal{I} \end{cases} \qquad w_i = \begin{cases} y_i & \text{if } i \notin \mathcal{I} \\ x_i & \text{if } i \in \mathcal{I} \end{cases}$$

- Then after two rounds $x_{i+2} \oplus y_{i+2} \oplus z_{i+2} \oplus w_{i+2} = 0$.
- But this works also backwards!

Yoyo Attack on 4-Round AES [R+17]

- Take two plaintexts P_0 and P_1 that have a zero difference in some Super S-box.
- After two rounds, you have zero difference in this Super S-box.
- But encryption continues for two more rounds.
- Look at the two ciphertexts C_0 and C_1.
- Swap outputs of Super S-boxes, to obtain C_0' and C_1'.

Yoyo Attack on 4-Round AES [R+17]

- Take two plaintexts P_0 and P_1 that have a zero difference in some Super S-box.
- After two rounds, you have zero difference in this Super S-box.
- But encryption continues for two more rounds.
- Look at the two ciphertexts C_0 and C_1.
- Swap outputs of Super S-boxes, to obtain C_0' and C_1'.
- Start decryption.
- "Before" two rounds of decryption the intermediate difference is *the same* as in the forward direction.
- This means that the original Super S-box has zero output difference!

Yoyo Attack on 4-Round AES [R+17]

- Take two plaintexts P_0 and P_1 that have a zero difference in some Super S-box.
- After two rounds, you have zero difference in this Super S-box.
- But encryption continues for two more rounds.
- Look at the two ciphertexts C_0 and C_1.
- Swap outputs of Super S-boxes, to obtain C_0' and C_1'.
- Start decryption.
- "Before" two rounds of decryption the intermediate difference is *the same* as in the forward direction.
- This means that the original Super S-box has zero output difference!
- Which means the input difference between P_0' and P_1' is 0 in the original Super S-box!

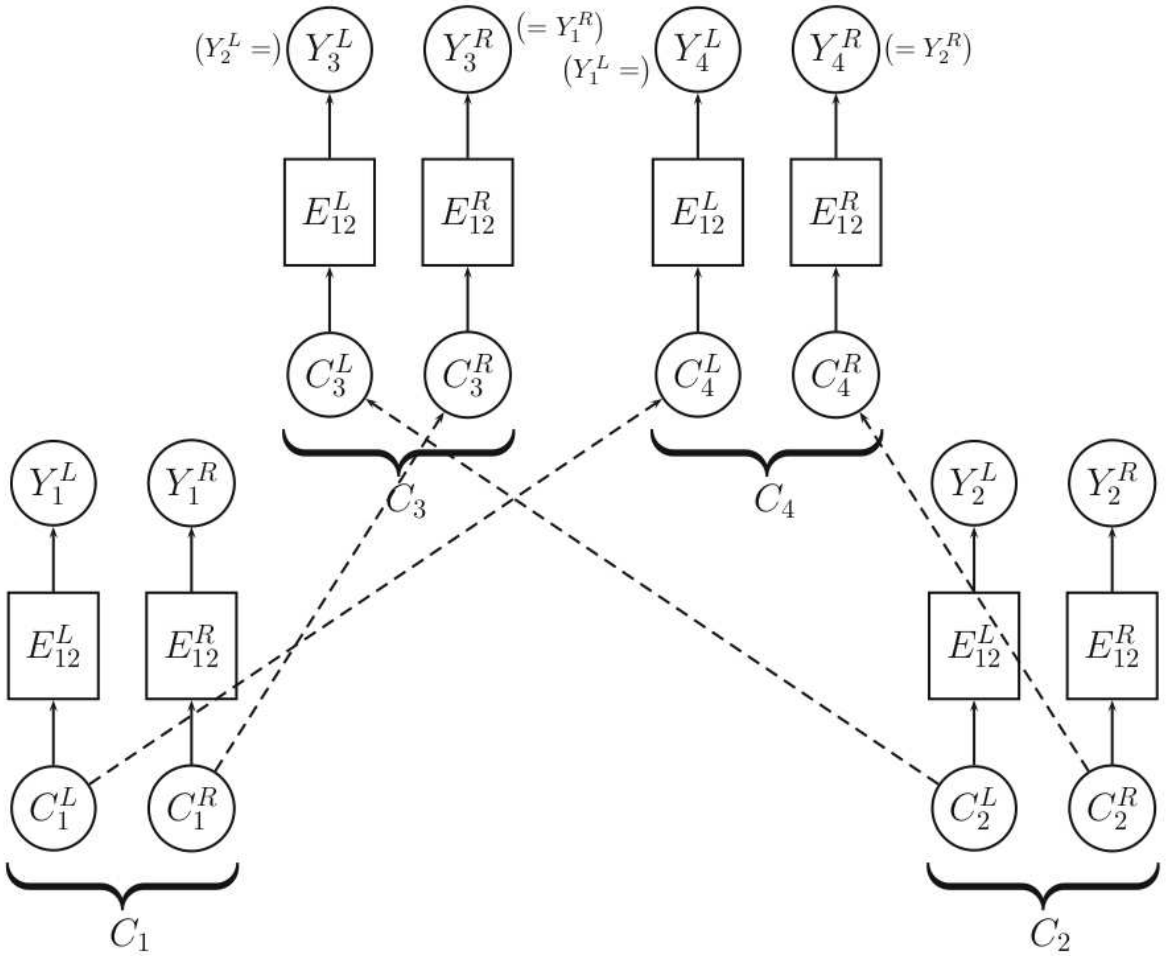
Complexity of Attacks of "This Type"

Rounds	Key Size	Data	Time	Memory	Source
5	All	$2^{11.3}$ ACPC	2^{31}	small	[R+17]
5	All	2^{32} CP	2^{32}	2^{32}	[G18]
5	All	$2^{22.25}$ CP	$2^{22.5}$	2^{20}	[B+19]
7	192	2^{30} CP	2^{153}	2^{32}	[B+19]
7	192	2^{32} CP	2^{145}	2^{40}	[B+19]
7	256	2^{30} CP	$2^{161.6}$	2^{48}	[B+19]

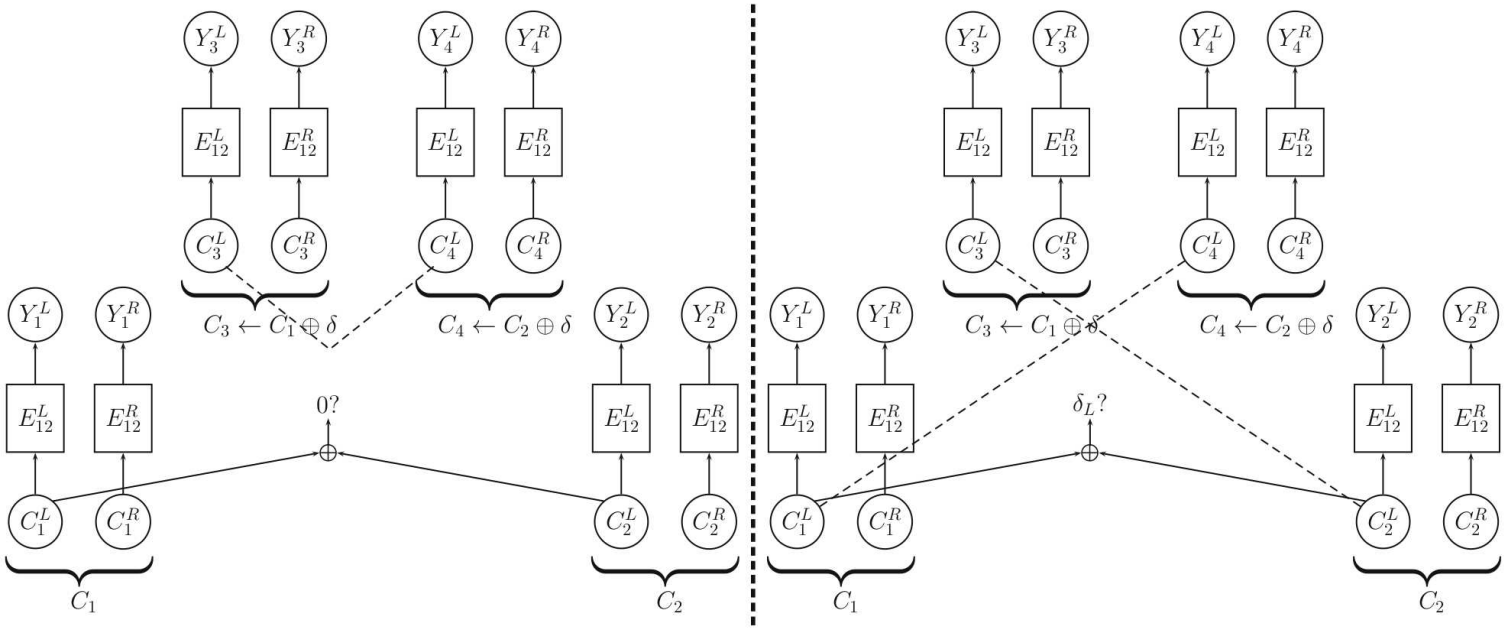
The Retracing Boomerang [D+20]

- As one can "feel", there is a relation between the boomerang process and the Yoyo one.
- One can use this ability to make the second differential part to behave "nicely".
- We introduced two types of retracing boomerangs which use this idea:
  - Mixing boomerangs — using the mixture trick to get higher probability of return.
  - Shifting boomerangs — hoping for some good event in the original ciphertext that allows for many "mixtures".

The Retracing Boomerang (cont.)



The Retracing Boomerang (cont.)



Results of the Retracing Boomerang Attack [B+20]

Rounds	Key Size	Data	Time	Memory
5	All	2^9 ACPC	2^{23}	2^9
5	All	2^{15} ACPC	$2^{16.5}$	2^9

Some 6-Round Results

Technique	Key Size	Data	Time	Memory	Source
Mixture Diff.	All	2^{31} CP	2^{73}	2^{31}	[B+18]
Mixture Diff.	All	2^{44} CP	2^{63}	2^{44}	[Y+24]
Boomeyong	All	2^{80} ACPC	2^{78}	2^{28}	[R+21]
Boomerang	All	2^{71} ACPC	2^{71}	2^{33}	[B04]
Retracing Boom.	All	2^{55} ACPC	2^{80}	2^{21}	[B+20]
Truncated Boom.	All	2^{59} ACPC	2^{61}	2^{59}	[BL23]
Retr. Trunc. Boom.	All	2^{51} ACPC	2^{68}	2^{32}	New!
Retr. Trunc. Boom.	All	2^{57} ACPC	2^{61}	2^{33}	New!

Related-Key Analysis of AES

- The AES key schedule, especially for AES-192 and AES-256 has “slow diffusion”.
- So one can introduce a key difference, and it would propagate very slowly.

Related-Key Analysis of AES

- The AES key schedule, especially for AES-192 and AES-256 has “slow diffusion”.
- So one can introduce a key difference, and it would propagate very slowly.
- Relatively slow.

Related-Key Analysis of AES

- The AES key schedule, especially for AES-192 and AES-256 has "slow diffusion".
- So one can introduce a key difference, and it would propagate very slowly.
- Relatively slow.
- Recall the [F+00] related-key attack.

Related-Key Boomerangs [H+05,BDK05]

- Well, all that we discussed about boomerangs?

Related-Key Boomerangs [H+05,BDK05]

- Well, all that we discussed about boomerangs?
- This works also when the differentials are related-key ones.

Related-Key Boomerangs [H+05,BDK05]

- Well, all that we discussed about boomerangs?
- This works also when the differentials are related-key ones.
- Independently, the idea was presented at FSE 2025 and EUROCRYPT 2025.

Related-Key Boomerangs [H+05,BDK05]

- Well, all that we discussed about boomerangs?
- This works also when the differentials are related-key ones.
- Independently, the idea was presented at FSE 2025 and EUROCRYPT 2025.

Rounds	Key Size	Data	Time	Memory	Source
8	192	$2^{86.5}$ RK-CP	$2^{86.5}$	–	[H+05]
9	192	$2^{86.5}$ RK-CP	2^{125}	–	[BDK05]
10	256	$2^{114.9}$ RK-CP	$2^{171.8}$	–	[BDK05]

The Full-Round Related-Key Attacks

Go and read Dmitry's and Ivica's papers from 2009, 2010, ...
They cover the first full-round related-key attacks on AES-192 and AES-256.

But Wait! There is More!

- Super S-box
- Biclique attacks
- Low-data complexity attacks
- New Observation about key schedule
- Algebraic attacks (or lack of)
- Known-key and chosen-key attacks
- Multiple-of-8 property
- Polytopic cryptanalysis
- Secret S-boxes
- Related-key impossible differential attacks
- Some more attacks [BR19, BR22]

Questions?

Thank you for your attention.