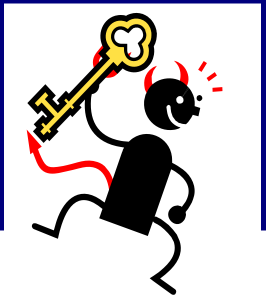
DFA Mechanism on the AES Key Schedule



Junko Takahashi, Toshinori Fukunaga and Kimihiro Yamakoshi

NTT Information Sharing Platform Laboratories, JAPAN

Outline

- Motivation
- Our results
- Analysis of DFA mechanism
- Our attack
- Conclusions

Motivation

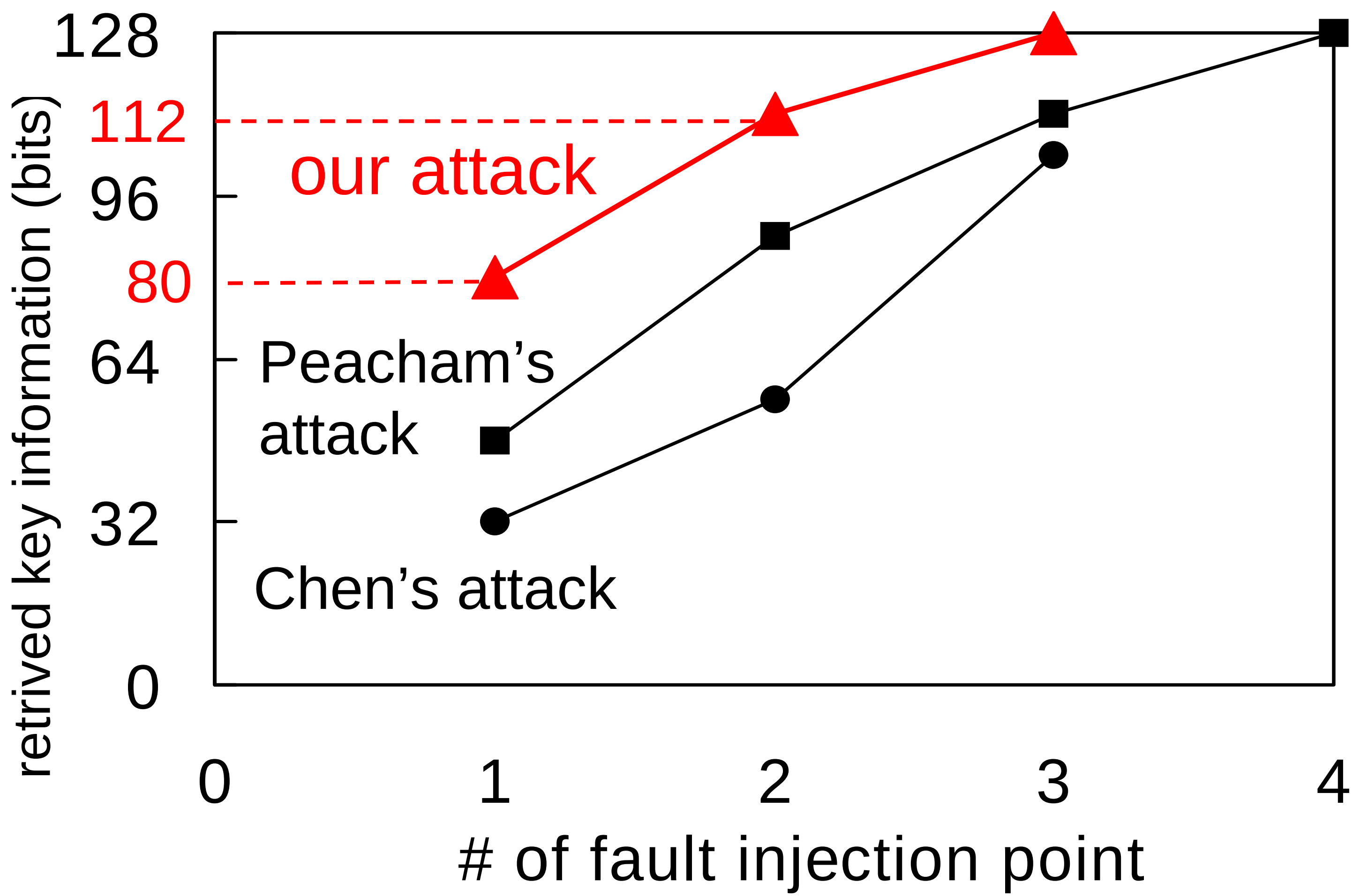
Previous studies have not addressed general attack approach for DFA against AES key schedule

- What is the general approach?
- Is there a more efficient attack than existing ones ?

Our results

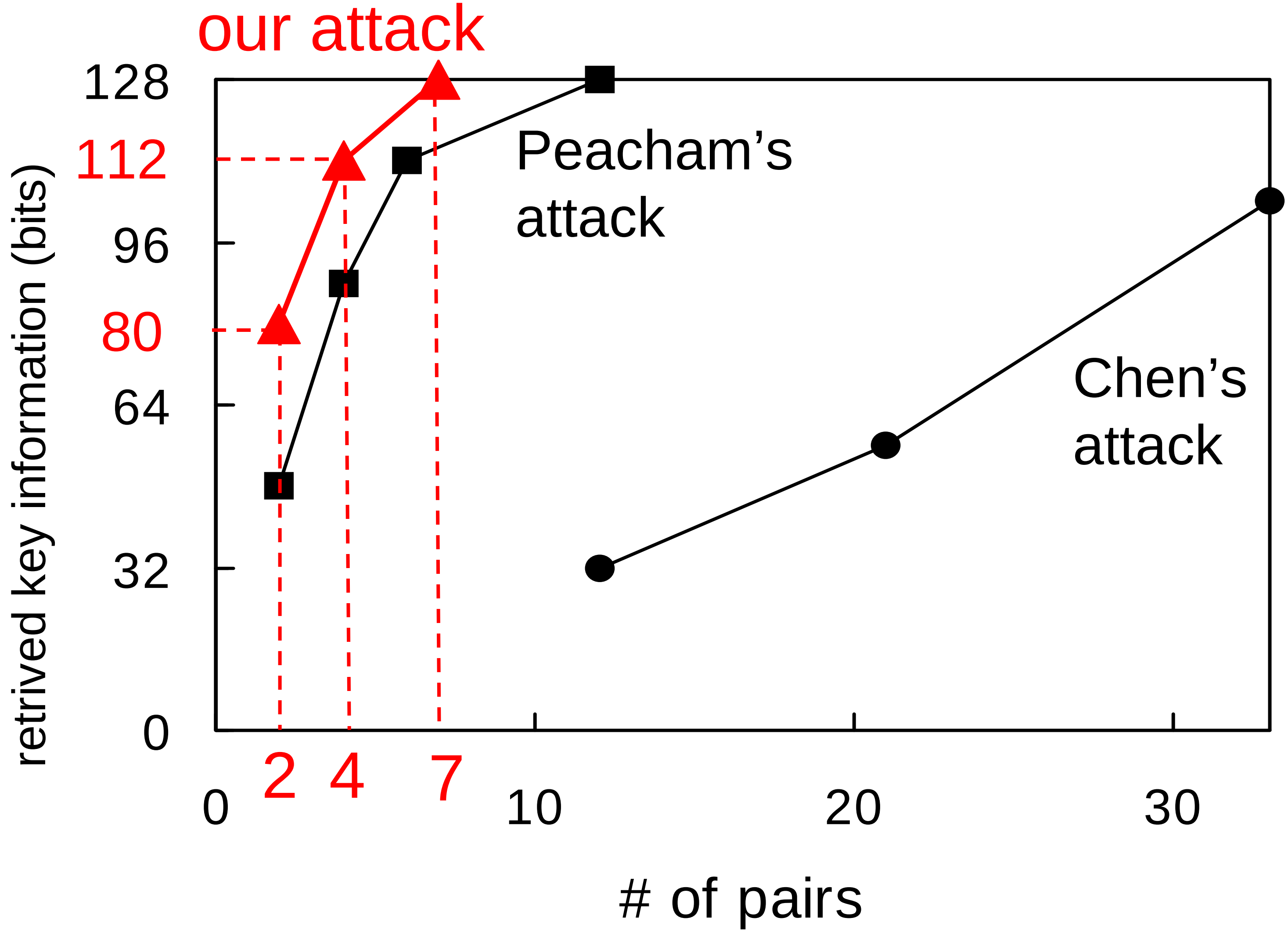
- Previous studies
  - No general expression of attack
  - Complicated simultaneous equations must be solved to obtain keys
- Our study
  - We found that DFA can be clearly represented, if seen from two sides
  - Only simple expressions and attack rules needed

Our results



128
112
96
80
64
32
0
retrived key information (bits)
our attack
Peacham's attack
Chen's attack
0
1
2
3
4
# of fault injection point

Our results

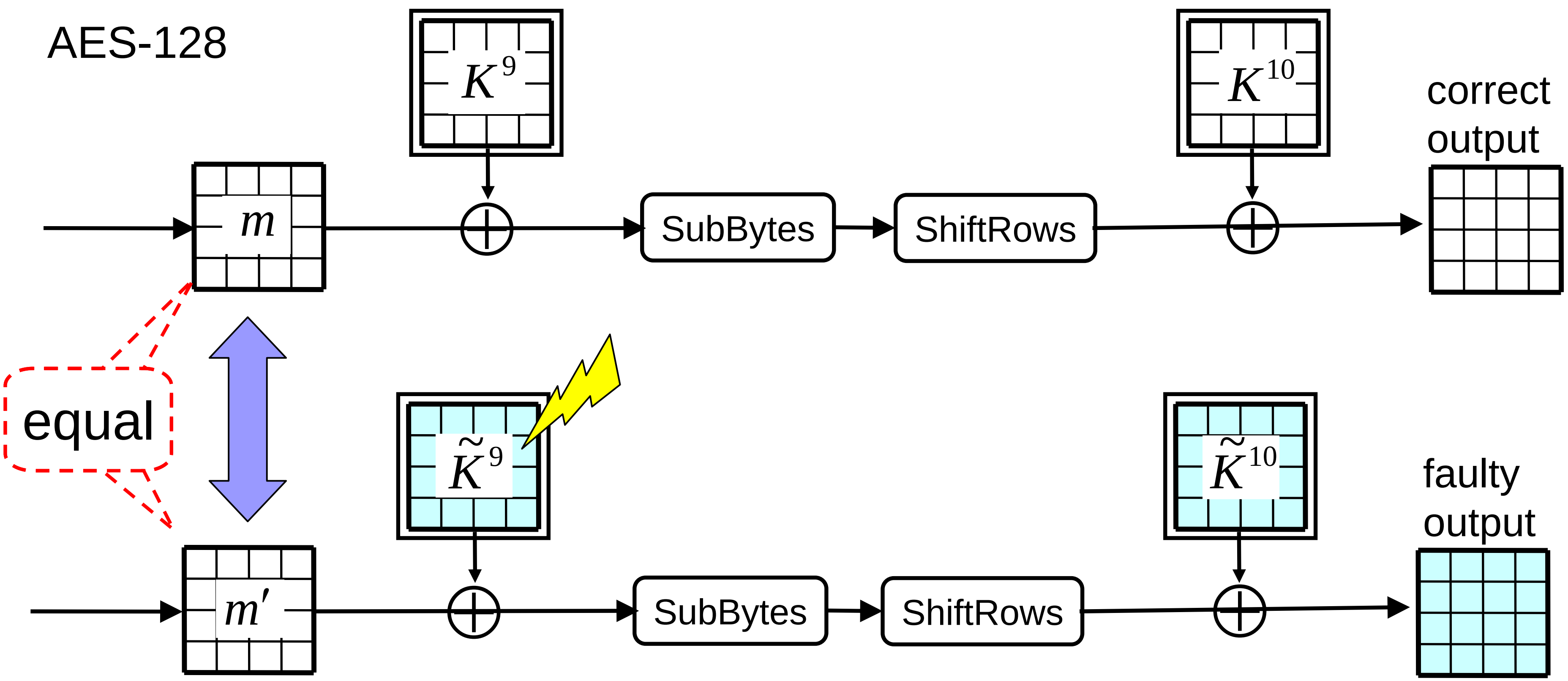


our attack
128
112
96
80
64
32
0
retrived key information (bits)
Peacham's attack
Chen's attack
0
2
4
7
10
20
30
# of pairs

- Motivation
- Our results
- Analysis of DFA mechanism
- Our attack
- Conclusions

DFA against AES key schedule

- States calculated by correct and faulty outputs must be equal, $m = m'$
- Solve simultaneous equations to obtain keys



Attack assumptions

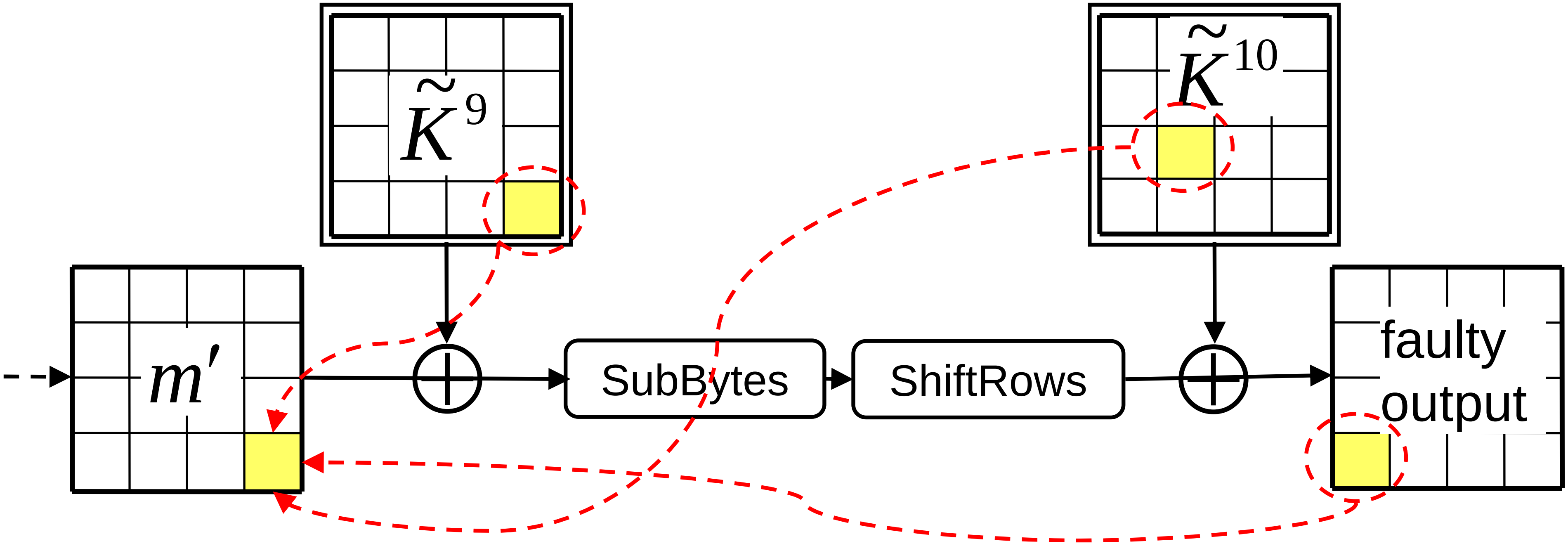
- Attacker can corrupt any byte(s) of the round key, but he can not choose the corrupted value of the byte(s) as he likes.

- Faults are not injected into byte(s) of the same row of the 9th round.

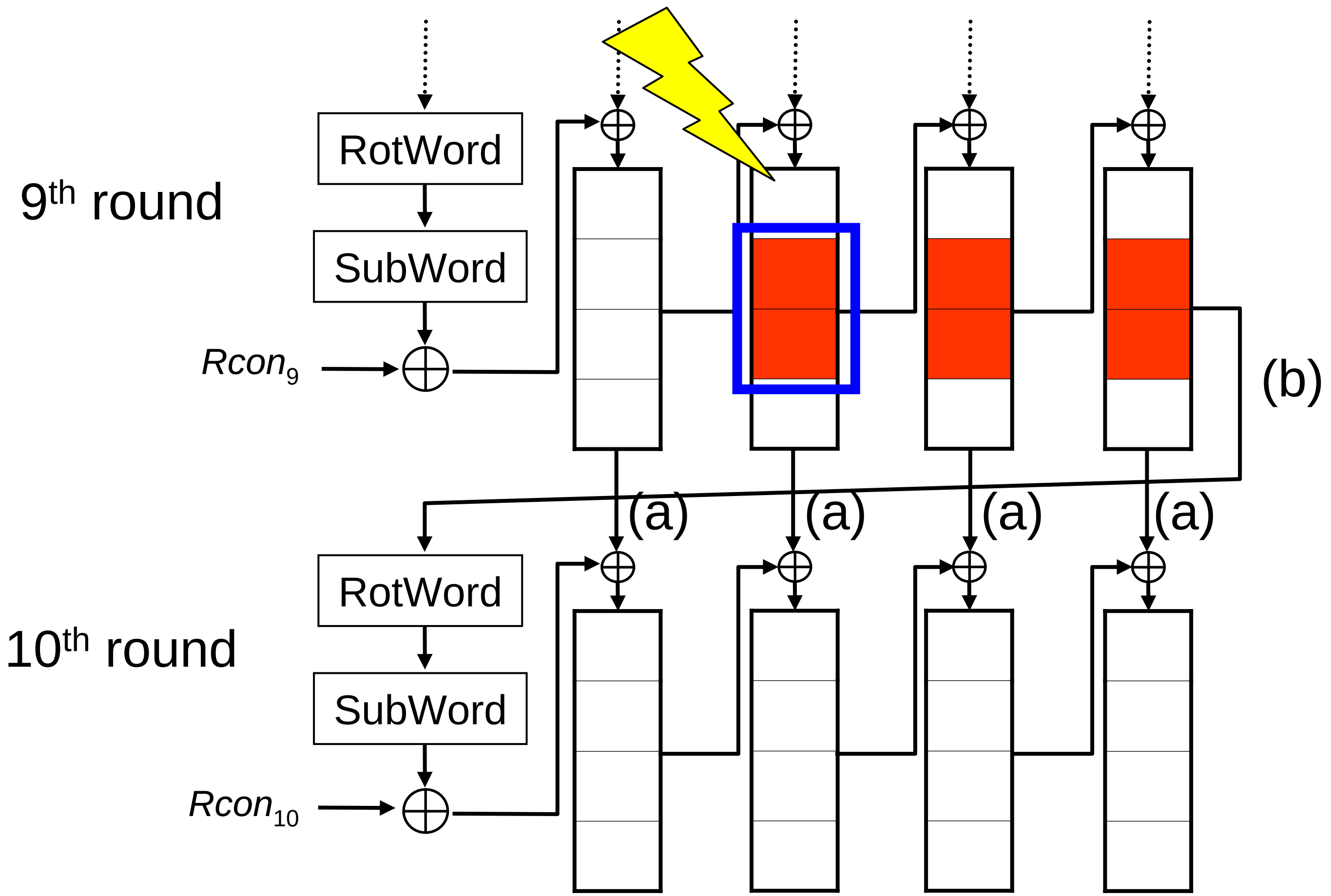
- $\varepsilon_{i,j} = K_{i,j} \oplus \tilde{K}_{i,j}$: error values (difference between correct and faulty keys)

Relation between m' and output

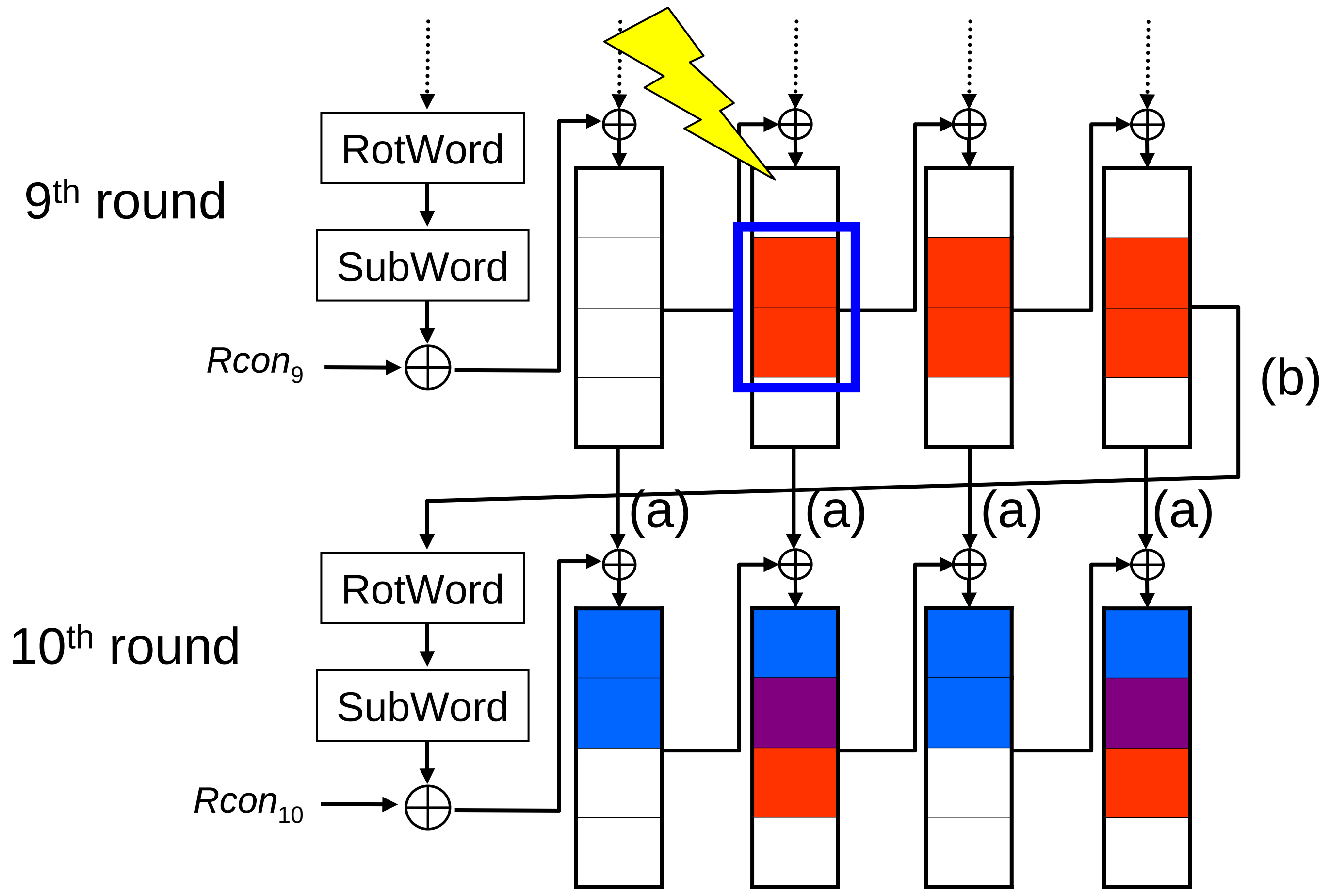
- Each byte of $\left\{ \begin{array}{c} m' \\ \text{equation } m = m' \end{array} \right\}$ represents a one-to-one correspondence with keys and outputs



Fault propagation in AES-128

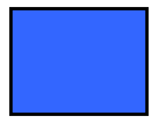
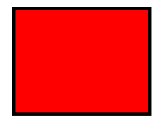
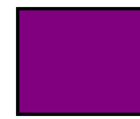


Fault propagation in AES-128

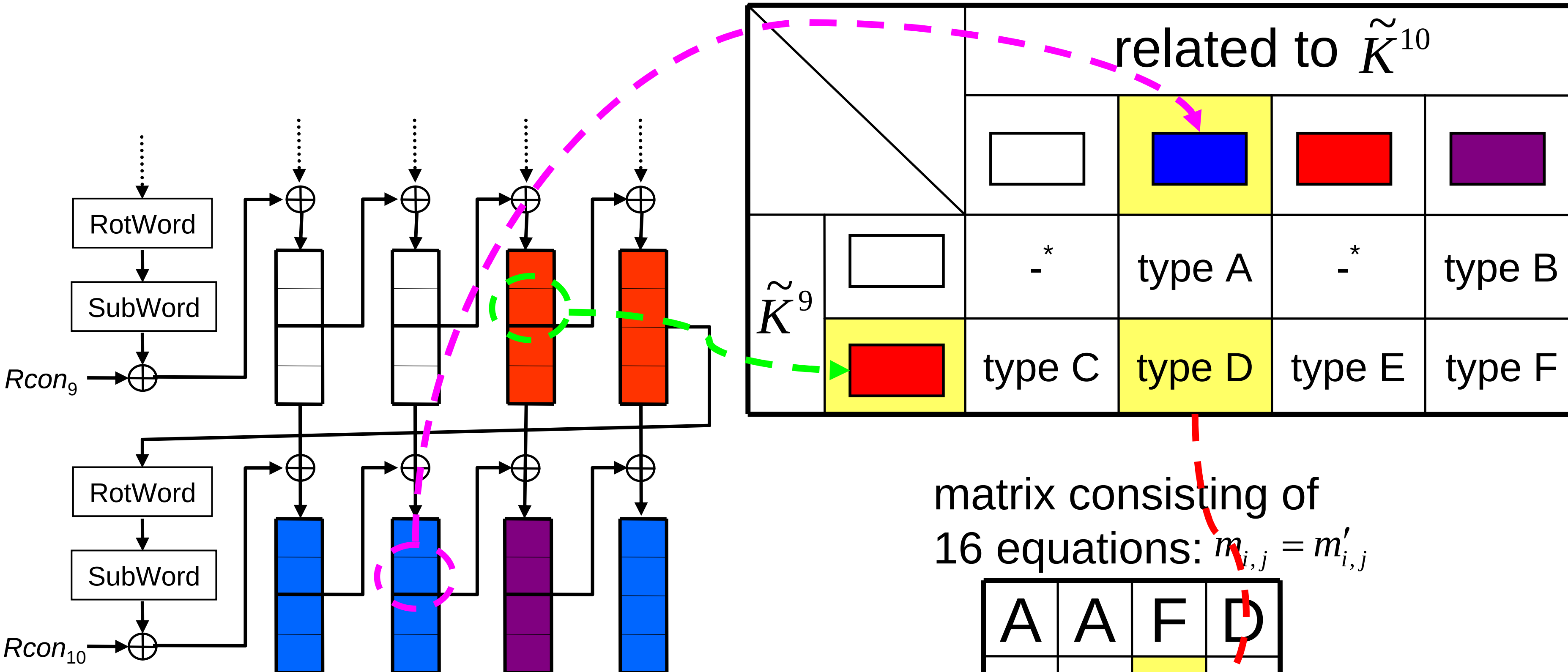


Classification : 8 patterns

- Each byte of $\left\{ \begin{array}{c} m' \\ \text{equation } m = m' \end{array} \right\}$ can be classified into 8 patterns

		related to $\tilde{K}^{10}$			*Not used in analysis
					
$\tilde{K}^{9}$		-*	type A	-*	type B
		type C	type D	type E	type F

m=m’ assigned to one of 8 patterns



matrix consisting of
16 equations: $m_{i,j} = m'_{i,j}$

A	A	F	D
A	A	D	F
B	A	D	D
A	B	D	D

Our idea

- 16 equations of $m_{i,j} = m'_{i,j}$ are classified into 8 patterns
- Some types are related
- Attack utilizes position of types and known values during the attack

matrix consisting of
16 equations: $m_{i,j} = m'_{i,j}$

A	A	F	D
A	A	D	F
B	A	D	D
A	B	D	D

related

related

Proposed 7 attack rules

- General expression of equation : $m_{i,j} = m'_{i,j}$

$$K_{i,j} \oplus S^{-1}\left[Q_{i,j} \oplus S\left[K_{i+1(\mathrm{mod}\,4),3}\right] \oplus y_{i,j}\right] = \tilde{K}_{i,j} \oplus S^{-1}\left[\tilde{Q}_{i,j} \oplus S\left[\tilde{K}_{i+1(\mathrm{mod}\,4),3}\right] \oplus \tilde{y}_{i,j}\right]$$

- In the case of type A byte on (i, j):

$$\cancel{K_{i,j}} \oplus S^{-1}\left[\cancel{Q_{i,j}} \oplus S\left[K_{i+1(\mathrm{mod}4),3}\right] \oplus y_{i,j}\right] = \cancel{K_{i,j}} \oplus S^{-1}\left[\cancel{Q_{i,j}} \oplus S\left[K_{i+1(\mathrm{mod}4),3} \oplus \varepsilon_{i+1(\mathrm{mod}4),j}\right] \oplus \tilde{y}_{i,j}\right]$$

$$S[K_{i+1(\mathrm{mod}4),3}] \oplus y_{i,j} = S[K_{i+1(\mathrm{mod}4),3} \oplus \varepsilon_{i+1(\mathrm{mod}4),j}] \oplus \tilde{y}_{i,j}$$

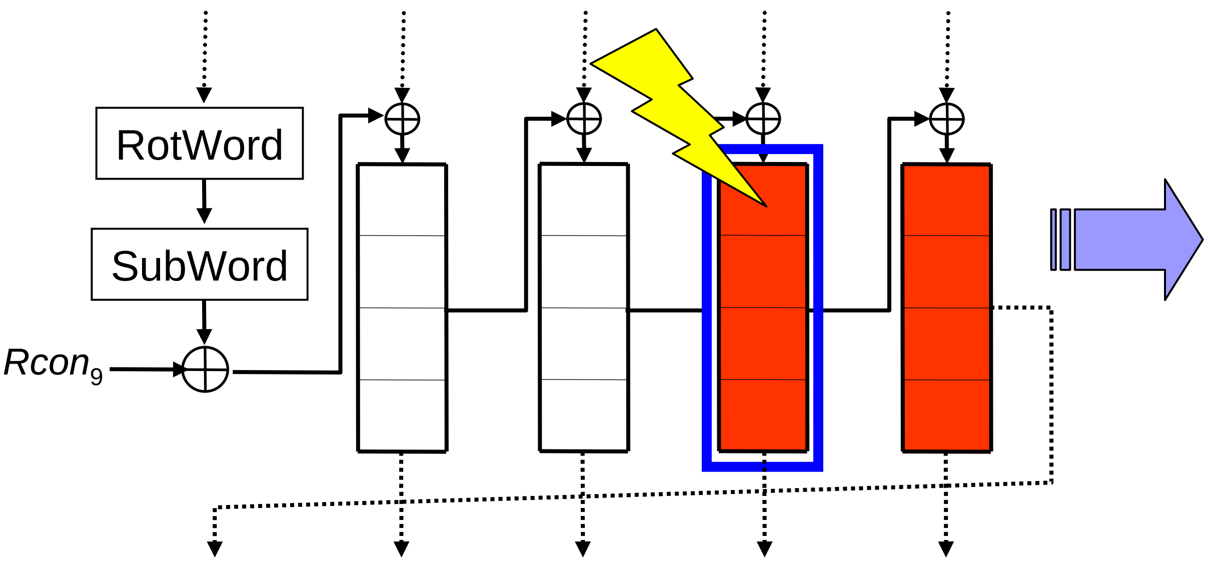
attack rule.2

if we know $\varepsilon_{i+1(\mathrm{mod}\,4),j}$ below type A, we can obtain $K_{i+1(\mathrm{mod}\,4),3}$ in the most right byte of the row below type A. We have to use 2 pairs of correct and faulty ciphertexts to determine $K_{i+1(\mathrm{mod}\,4),3}$.

- Motivation
- Our results
- Analysis on DFA mechanism
- Our attack
- Conclusions

Our attack with one fault injection

9th round



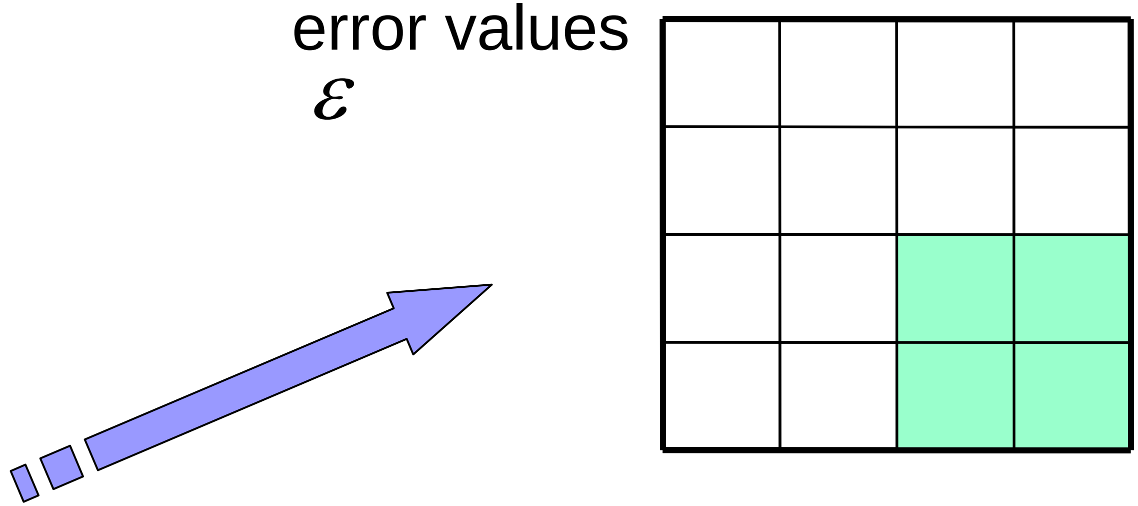
matrix consisting of
16 equations: $m_{i,j} = m'_{i,j}$

A	A	F	D
A	A	D	F
B	A	D	D
A	B	D	D

Attack procedure

error values

ε



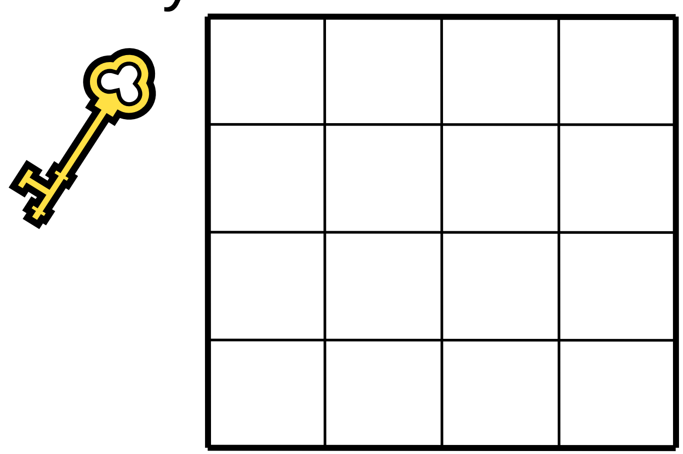
apply to rule.1

matrix consisting of
16 equations: $m_{i,j} = m'_{i,j}$

A	A	F	D
A	A	D	F
B	A	D	D
A	B	D	D

9^{th} round key

K

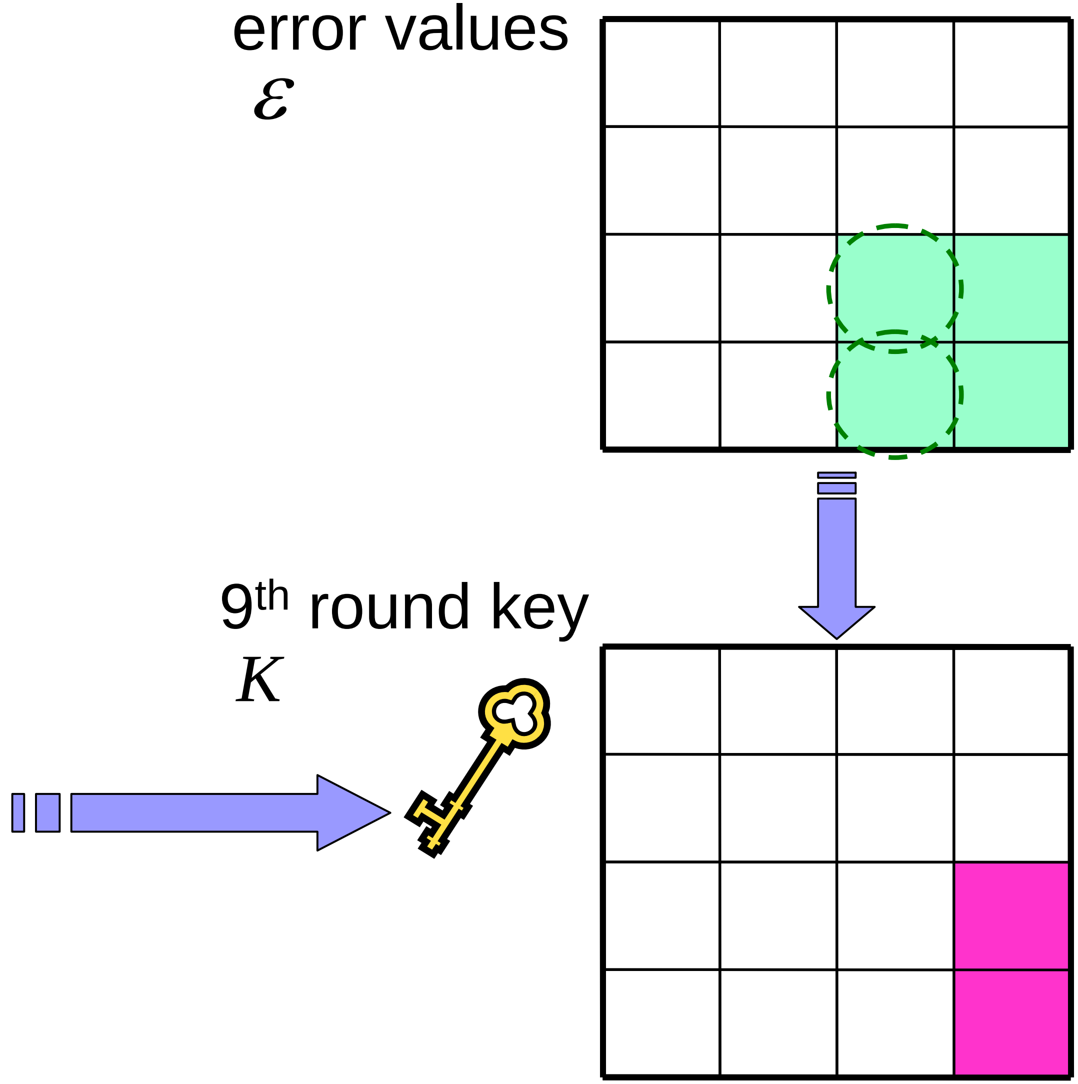


Attack procedure

apply to rule.2

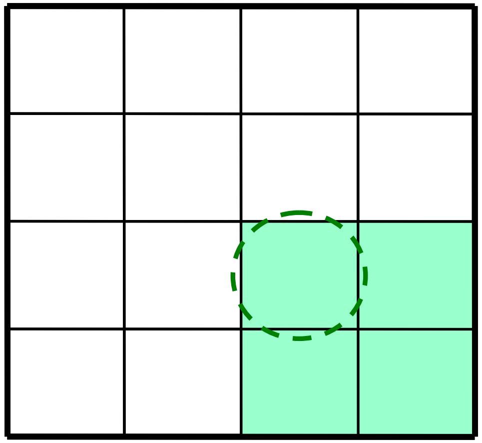
matrix consisting of
16 equations: $m_{i,j} = m'_{i,j}$

A	A	F	D
A	A	D	F
B	A	D	D
A	B	D	D



Attack procedure

error values
ε

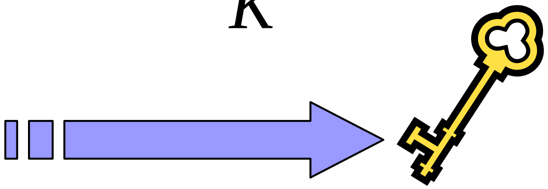


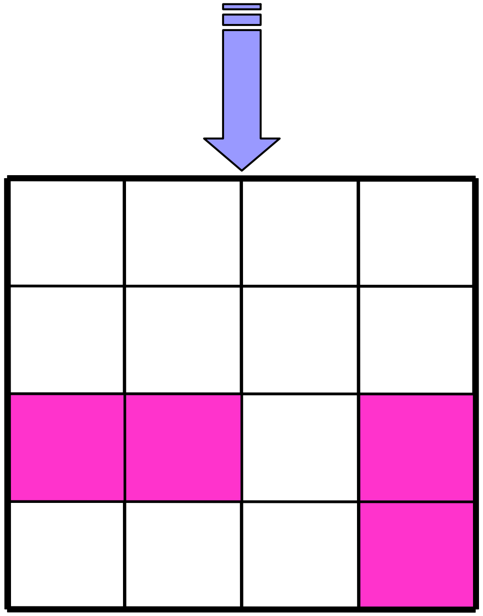
apply rule.3 and rule.5

matrix consisting of
16 equations: $m_{i,j} = m'_{i,j}$

A	A	F	D
A	A	D	F
B	A	D	D
A	B	D	D

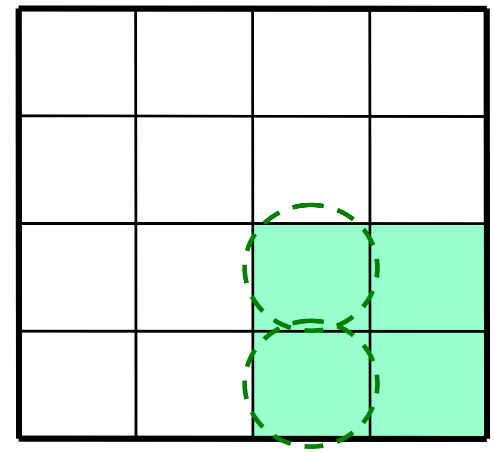
9th round key
K





Attack procedure

error values
ε

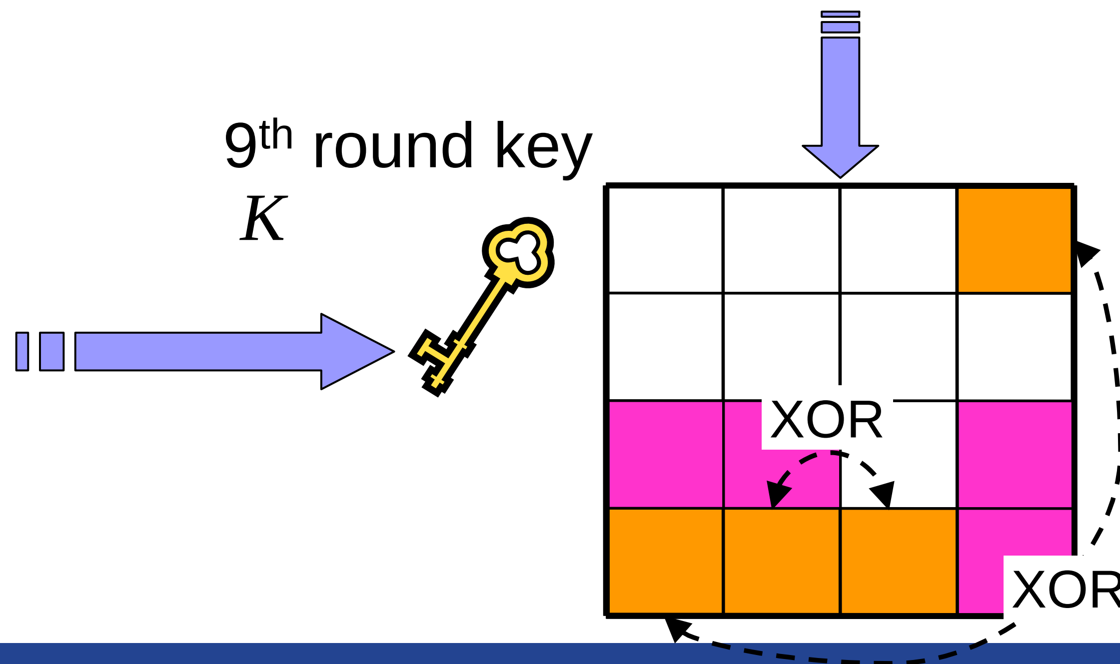


apply rule.3 and rule.5

matrix consisting of
16 equations: $m_{i,j} = m'_{i,j}$

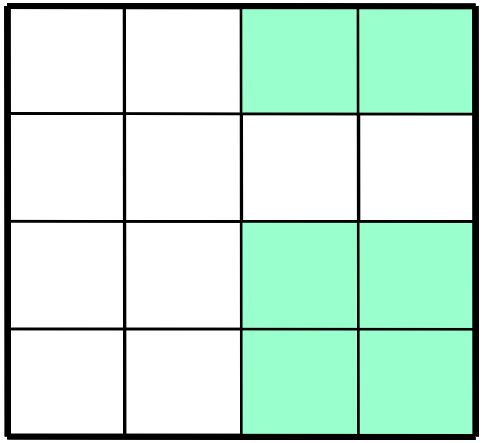
A	A	F	D
A	A	D	F
B	A	D	D
A	B	D	D

9^{th} round key
K



Attack procedure

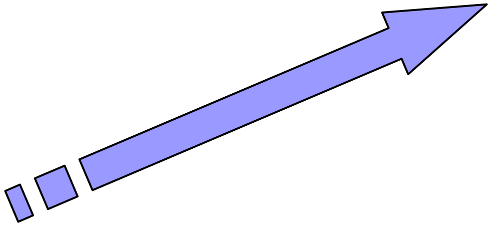
error values ε



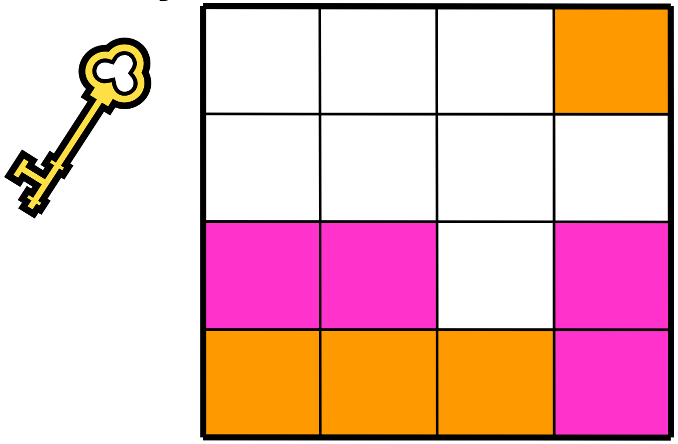
apply rule.1

matrix consisting of 16 equations: $m_{i,j} = m'_{i,j}$

A	A	F	D
A	A	D	F
B	A	D	D
A	B	D	D



9th round key K

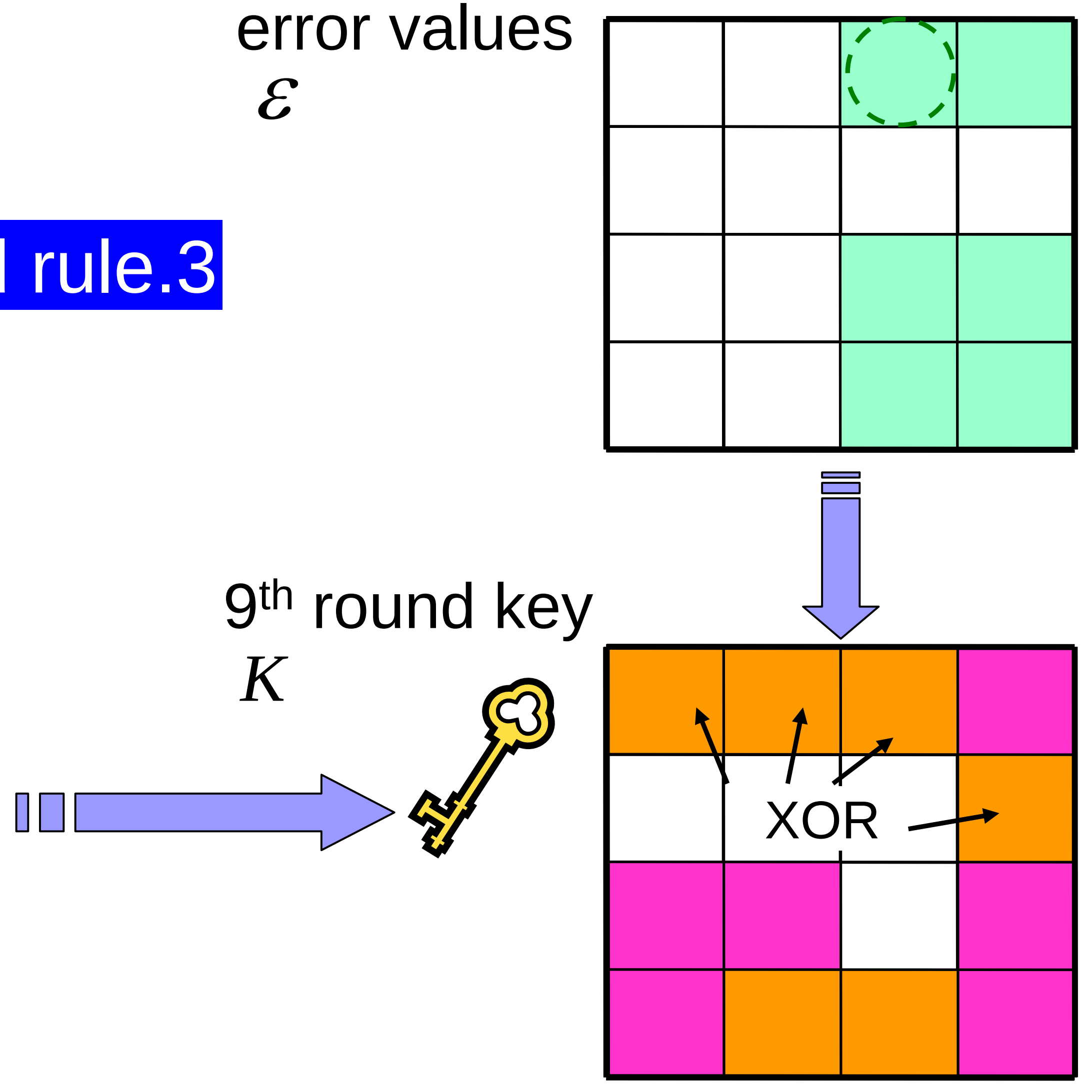


Attack procedure

apply rule.2 and rule.3

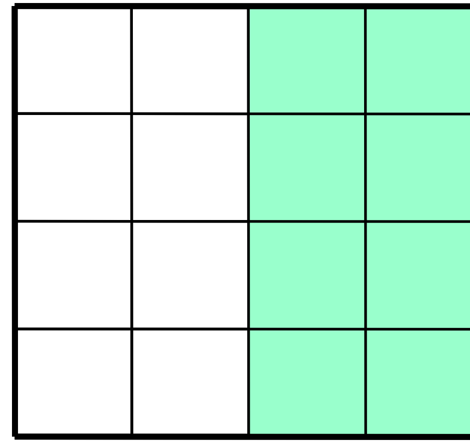
matrix consisting of
16 equations: $m_{i,j} = m'_{i,j}$

A	A	F	D
A	A	D	F
B	A	D	D
A	B	D	D



Attack procedure

error values
ε

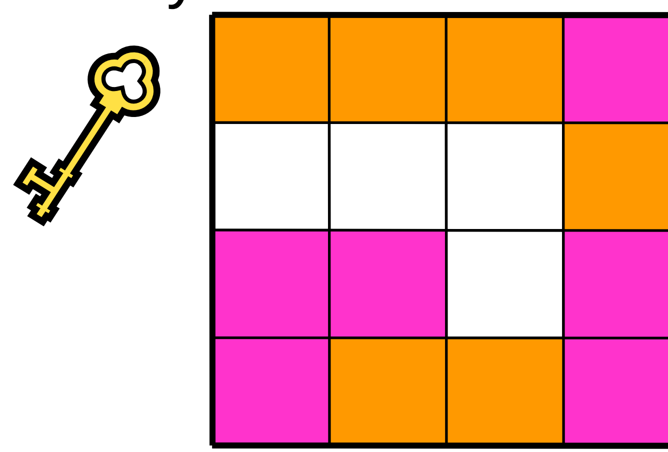


apply rule.2

matrix consisting of
16 equations: $m_{i,j} = m'_{i,j}$

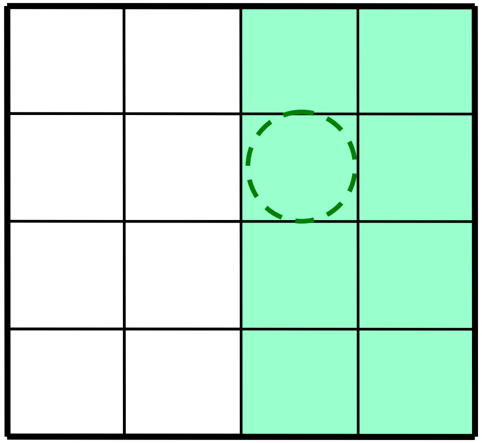
A	A	F	D
A	A	D	F
B	A	D	D
A	B	D	D

9th round key
K



Attack procedure

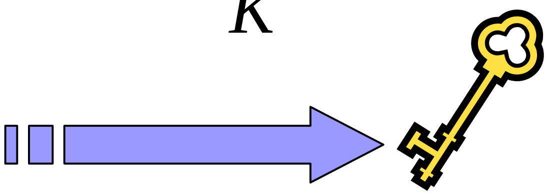
error values $\mathcal{E}$



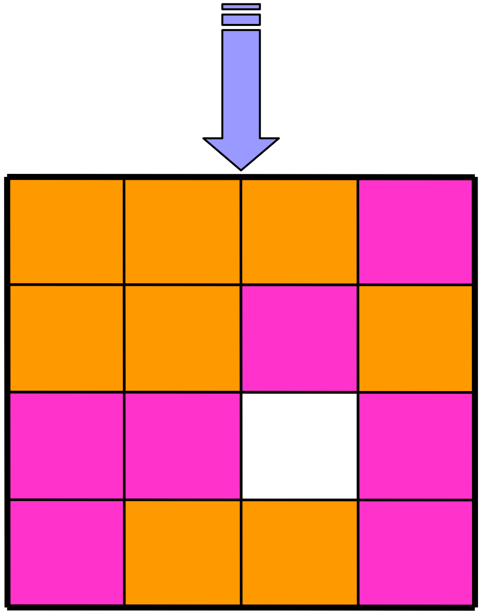
apply rule.3 and rule.5

matrix consisting of
16 equations: $m_{i,j} = m'_{i,j}$

A	A	F	D
A	A	D	F
B	A	D	D
A	B	D	D

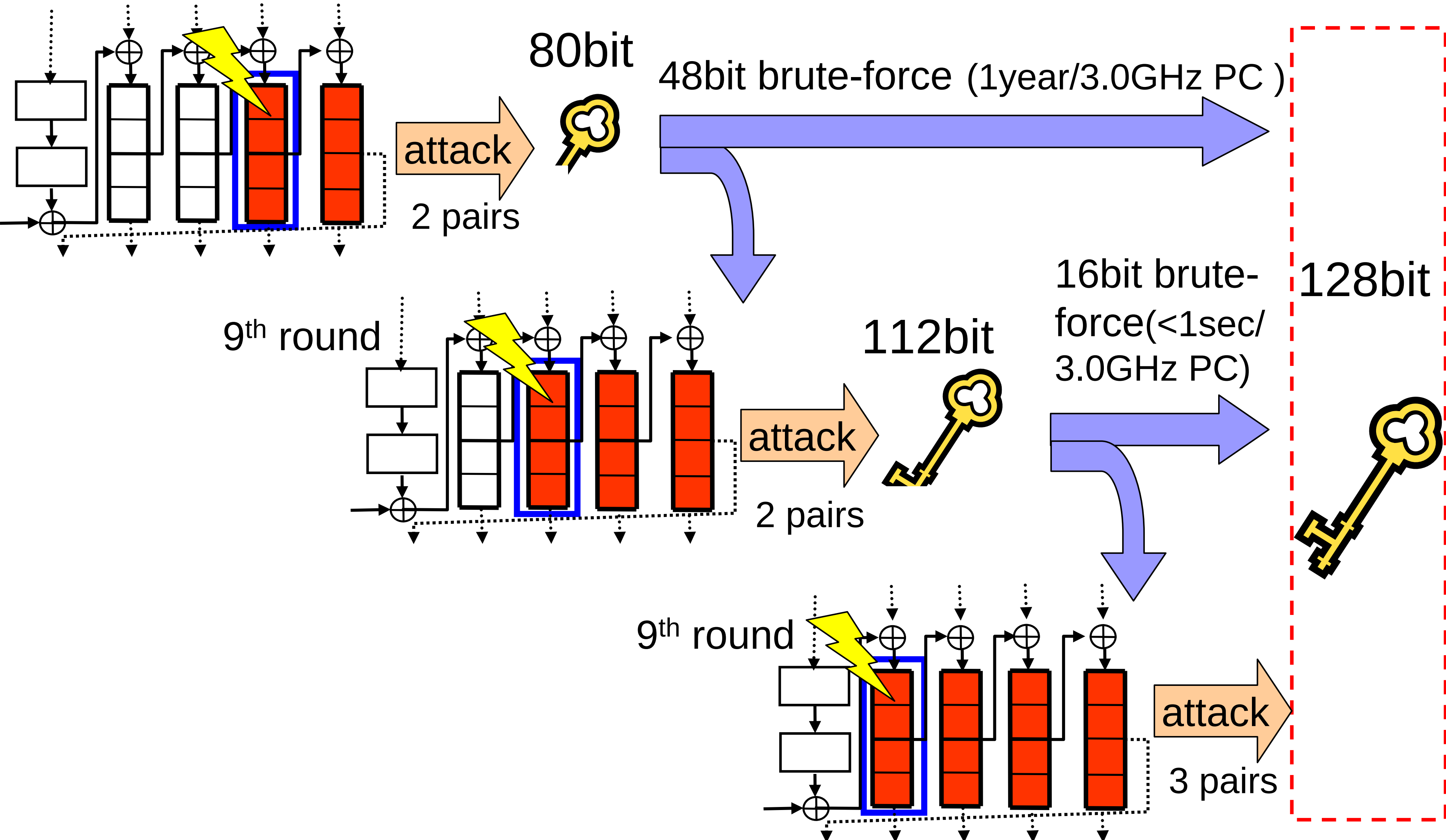


9th round key K

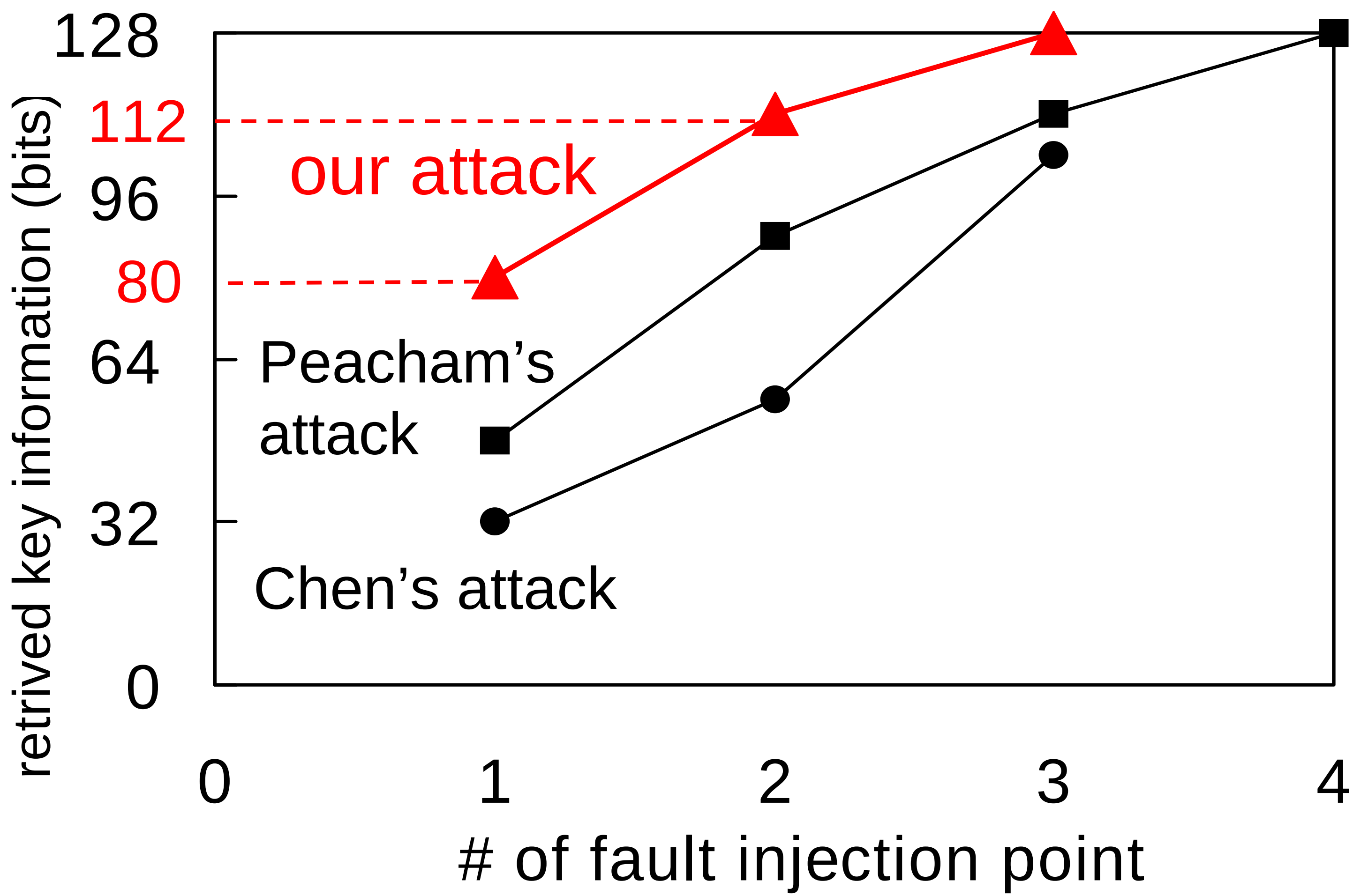


We can obtain information equivalent to 80 bits of key

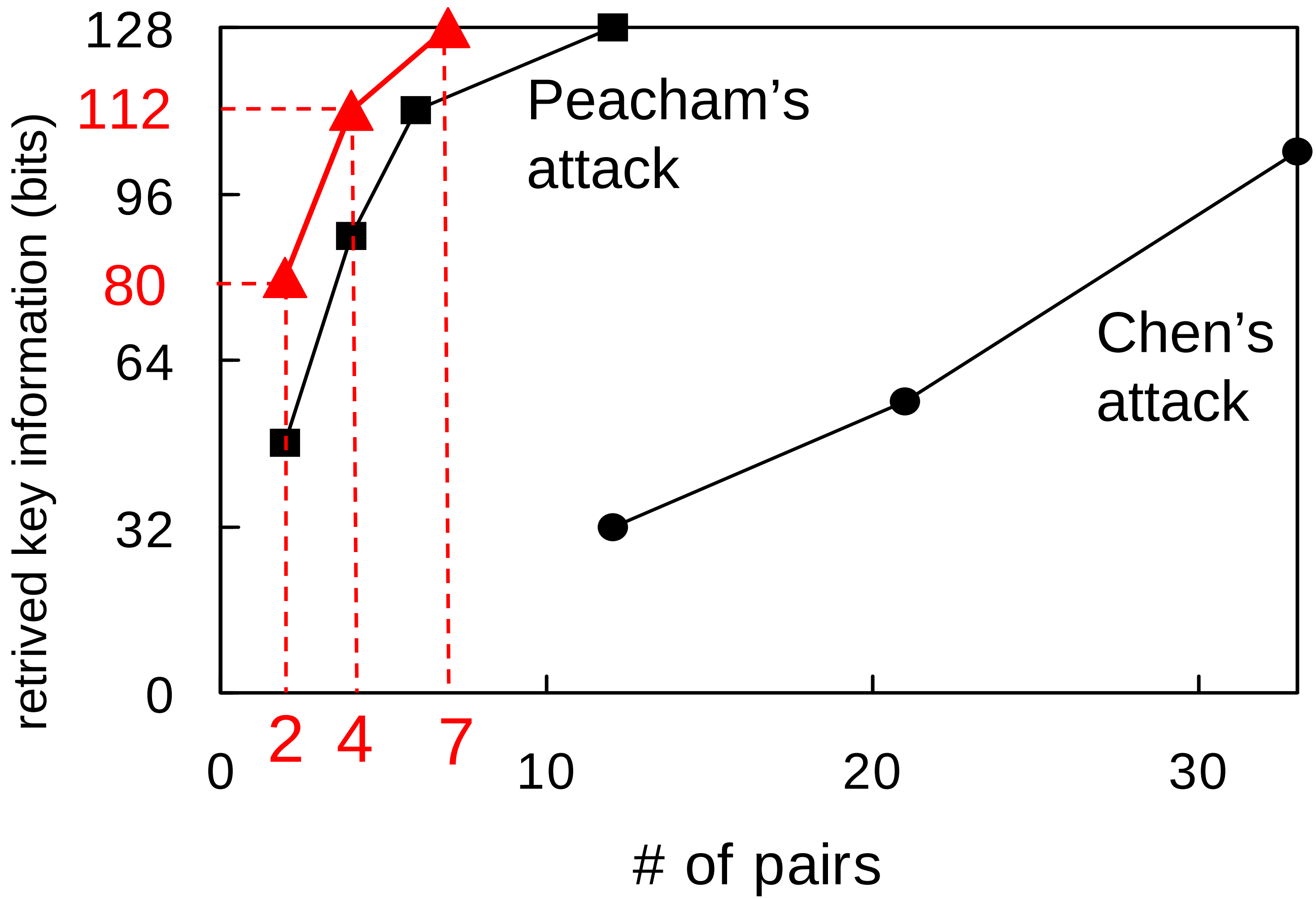
How to retrieve a complete key



Comparison to existing attacks



Comparison to existing attacks



- Motivation
- Our results
- Analysis on DFA mechanism
- Our attack
- Conclusions

Conclusions

- Analysis of DFA mechanism
  - We found that DFA against the AES key schedule can be clearly represented, when seen from two sides,
    - how each key byte is affected by fault injection
    - position of each type affected by fault injection
  - We proposed how to get the complete key with the position of types read from simple expressions and attack rules.

- efficient attack
  - It is much more efficient.
    - 2-pairs needed with 48-bit brute-force search
    - 7-pairs needed without brute-force search

Thank you very much for your attention !!