上海交通大學
SHANGHAI JIAO TONG UNIVERSITY

Differential Fault Analysis on Lightweight Blockciphers with Statistical Cryptanalysis Techniques

Dawu Gu, Juanru Li, Sheng Li, Zheng Guo, Junrong Liu

Shanghai Jiao Tong University

FDTC 2012, 9 September 2012

密码与计算机安全实验室
LoCCS Lab of Cryptology and Computer Security

Outlines

- Fault Analysis Review and General Princinple
- PRESENT and PRINTcipher Specification
- Attack Setup and Details
- Simulation Result
- Conclusion

Fault Analysis

- Fault Analysis was proposed and developed by
  - D. Boneh, R. DeMillo, and R. Lipton, "On the importance of checking cryptographic protocols for faults"
  - E. Biham and A. Shamir, "Differential fault analysis of secret key cryptosystems," CRYPTO'97.
  - et al
- Using some pairs of correct and faulty ciphertexts to recover the secret key

General DFA Principles

- Guess and determine
- An equation or equations involve correct and faulty ciphertexts and partial round keys

$$f\,(\mathrm{C}, \mathrm{C}^{*}, \mathrm{rk}) = Consts$$

  - right key guess always passes the test
  - Wrong key guesses fail with great probability

    - Correctness

General DFA Principles

- Combining divide and conquer
- Each equation involves partial round keys within exhaustive search

$$f\,(\mathrm{C}, \mathrm{C}^{*}, \mathrm{rk}) = Consts$$

- Efficiency

上海交通大學
SHANGHAI JIAO TONG UNIVERSITY

New Challenges

- Countermeasures
  - More robust hardware to make the injection harder
  - Compute the last few rounds twice and check the integrity
- Research goal
  - Less fault injections
  - Earlier injection rounds
  - More practical fault model

→ More sufficient diffusion

→ There doesn't exist clear equations with required properties

Our Attack Principles

Solutions

- Adjust considering the vaule of $f\,(C, C^*, rk)$ to the distribution of $f\,(C, C^*, rk)$
- Distribution is a statistical concepts
  - More faults needed
- Methods to evaluate the similarity of distribution

PRESENT

a 31-round SPN block cipher with 64 bits block size and supports 80/128 bits key. (CHES 2007)

Algorithm 1: PRESENT

Input: $u_1, K_1 - K_{32}$

Output: u_{32}

for i = *1 to 31* **do**

  addRoundKey(u_i, K_i)

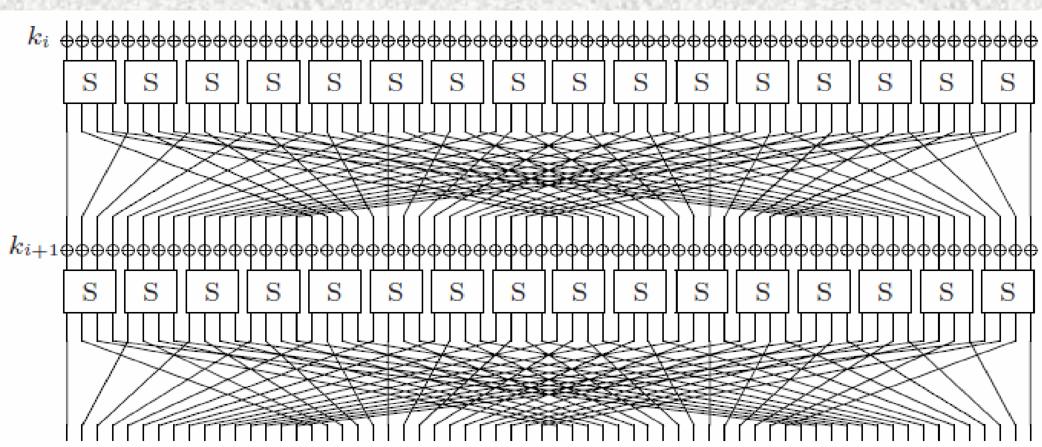
  sBoxlayer(u_i)

  permutationLayer(u_i)

end

$addRoundKey(u_{32}, K_{32})$

return u_{32}



PRESENT

a0	a1	a4	a5	a8	a9	a12	a13
a2	a3	a6	a7	a10	a11	a14	a15
a16	a17	a20	a21	a24	a25	a28	a29
a18	a19	a22	a23	a26	a27	a30	a31
a32	a33	a36	a37	a40	a41	a44	a45
a34	a35	a38	a39	a42	a43	a46	a47
a48	a49	a52	a53	a56	a57	a60	a61
a50	a51	a54	a55	a58	a59	a62	a63

$\oplus$

rk0	rk1	rk4	rk5	rk8	rk9	rk12	rk13
rk2	rk3	rk6	rk7	rk10	rk11	rk14	rk15
rk16	rk17	rk20	rk21	rk24	rk25	rk28	rk29
rk18	rk19	rk22	rk23	rk26	rk27	rk30	rk31
rk32	rk33	rk36	rk37	rk40	rk41	rk44	rk45
rk34	rk35	rk38	rk39	rk42	rk43	rk46	rk47
rk48	rk49	rk52	rk53	rk56	rk57	rk60	rk61
rk50	rk51	rk54	rk55	rk58	rk59	rk62	rk63

→

b0	b1	b4	b5	b8	b9	b12	b13
b2	b3	b6	b7	b10	b11	b14	b15
b16	b17	b20	b21	b24	b25	b28	b29
b18	b19	b22	b23	b26	b27	b30	b31
b32	b33	b36	b37	b40	b41	b44	b45
b34	b35	b38	b39	b42	b43	b46	b47
b48	b49	b52	b53	b56	b57	b60	b61
b50	b51	b54	b55	b58	b59	b62	b63

Add RoundKey

S-box

c0	c1	c4	c5	c8	c9	c12	c13
c2	c3	c6	c7	c10	c11	c14	c15
c16	c17	c20	c21	c24	c25	c28	c29
c18	c19	c22	c23	c26	c27	c30	c31
c32	c33	c36	c37	c40	c41	c44	c45
c34	c35	c38	c39	c42	c43	c46	c47
c48	c49	c52	c53	c56	c57	c60	c61
c50	c51	c54	c55	c58	c59	c62	c63

Bit-Permutation

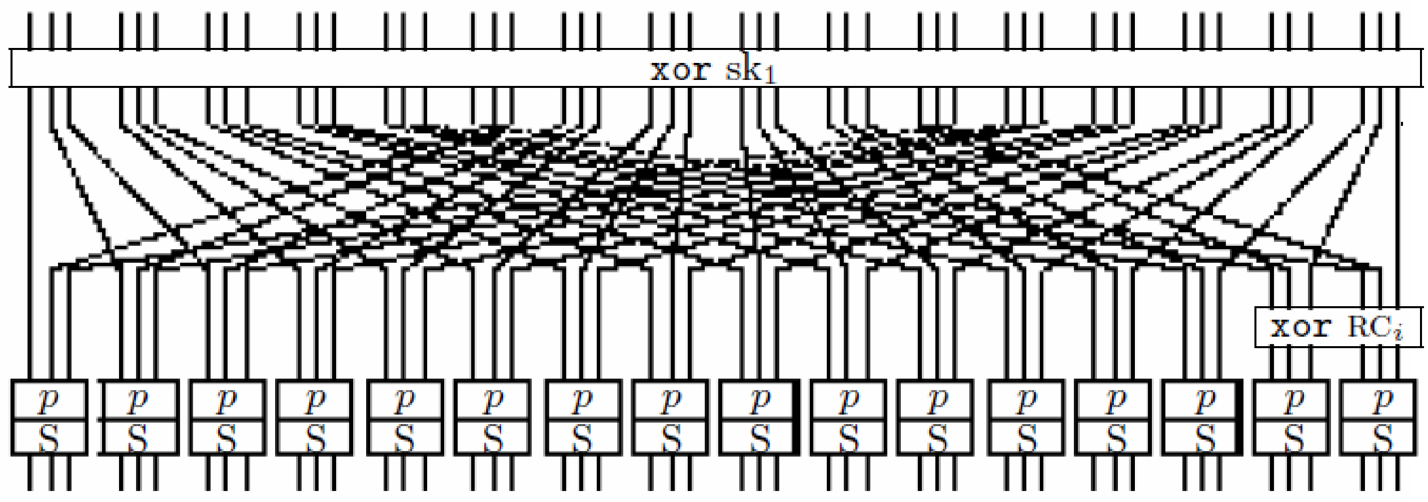
c0	c4	c16	c20	c32	c36	c48	c52
c8	c12	c24	c28	c40	c44	c56	c60
c1	c5	c17	c21	c33	c37	c49	c53
c9	c13	c25	c29	c41	c45	c57	c61
c2	c6	c18	c22	c34	c38	c50	c54
c10	c14	c26	c30	c42	c46	c58	c62
c3	c7	c19	c23	c35	c39	c51	c55
c11	c15	c27	c31	c43	c47	c59	c63

PRINTcipher

a 48/96-round SPN block cipher with 48/96 bits block size and supports 80/160 bits key.(CHES 2010)

Algorithm 2: PRINTCIPHER

Input: $u_1, K_1 - Kr$
Output: u_r
for i = *1 to r* **do**
  addRoundKey(u_i, K_i)
  linearDiffusion(u_i)
  xorRoundCounter(u_i)
  keyedPermutation(u_i)
  sBoxlayer(u_i)
end
return u_r



Previous Results

PRESENT-80/PRESENT-128

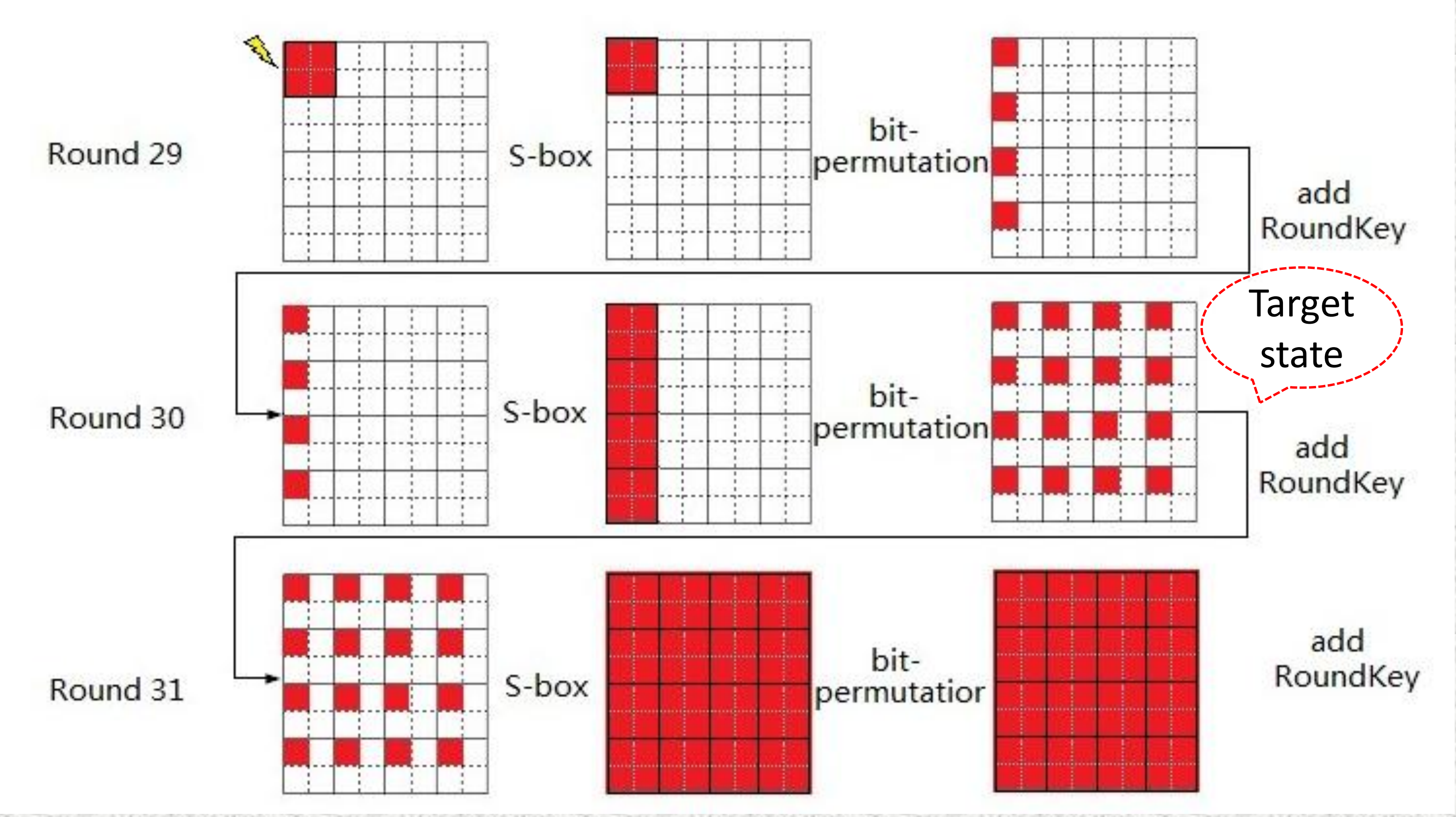
	Round	Numbers	Complex	Fault model
J. Li et al	r-1th	40-50/-	2^{16}/-	1 nibble fault on encryption
G. Wang et al	30th and 31st round key	64/-	2^{29}/-	1 nibble fault on key schedule
X. Zhao et al	r-2th	8/16	$2^{14.7}/2^{21.1}$	1 nibble fault on encryption

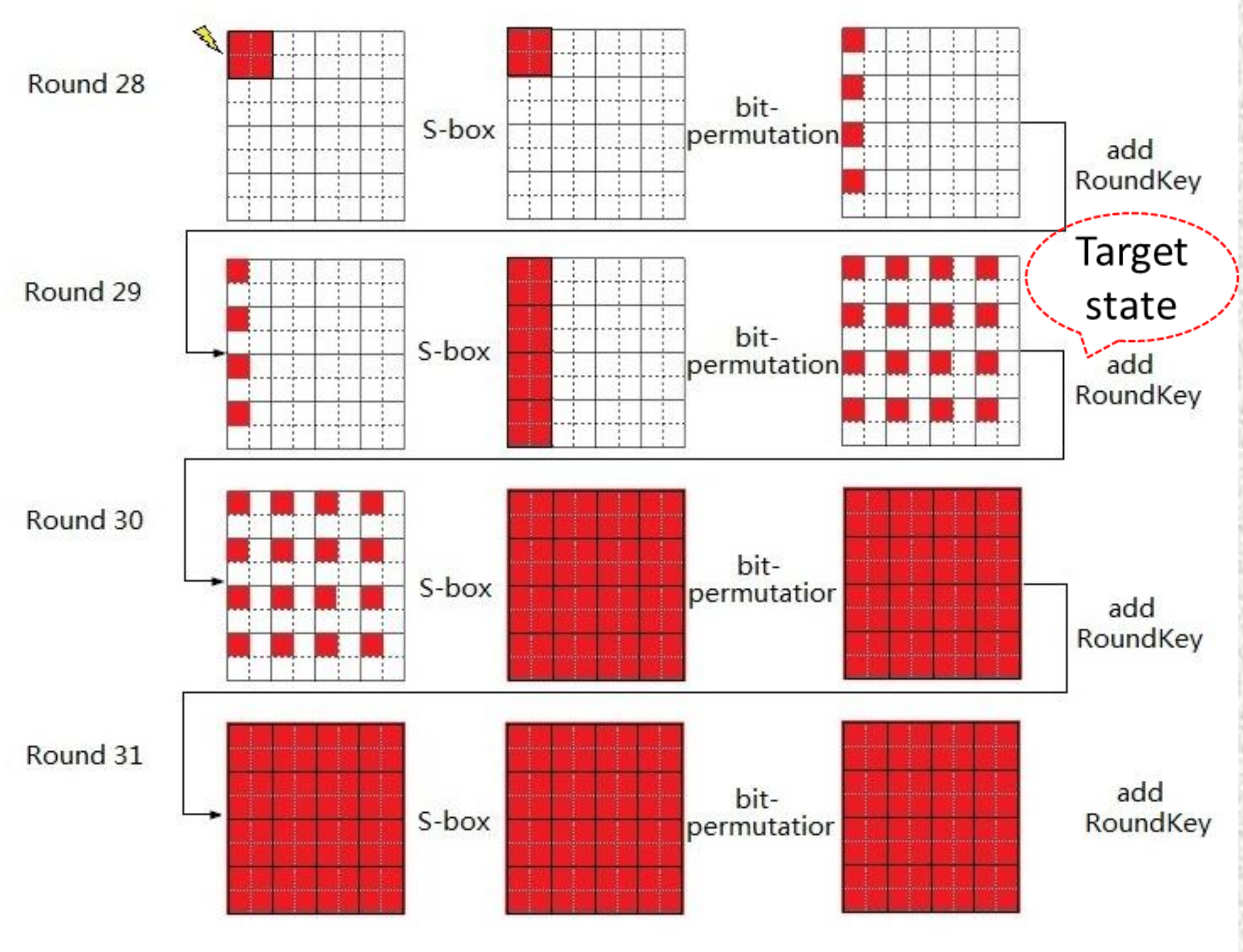
PRINTcipher-48/PRINTcipher-96

	Round	Numbers	Complex	Fault model
X. Zhao et al	r-2th	12/24	$2^{13.7}/2^{22.8}$	1 nibble fault on encryption
X. Zhao et al	r-3th	-/8	$-/2^{18.7}$	1 nibble fault on encryption

密码与计算机安全实验室
Lab of Cryptology and Computer Security

Previous Fault Analysis



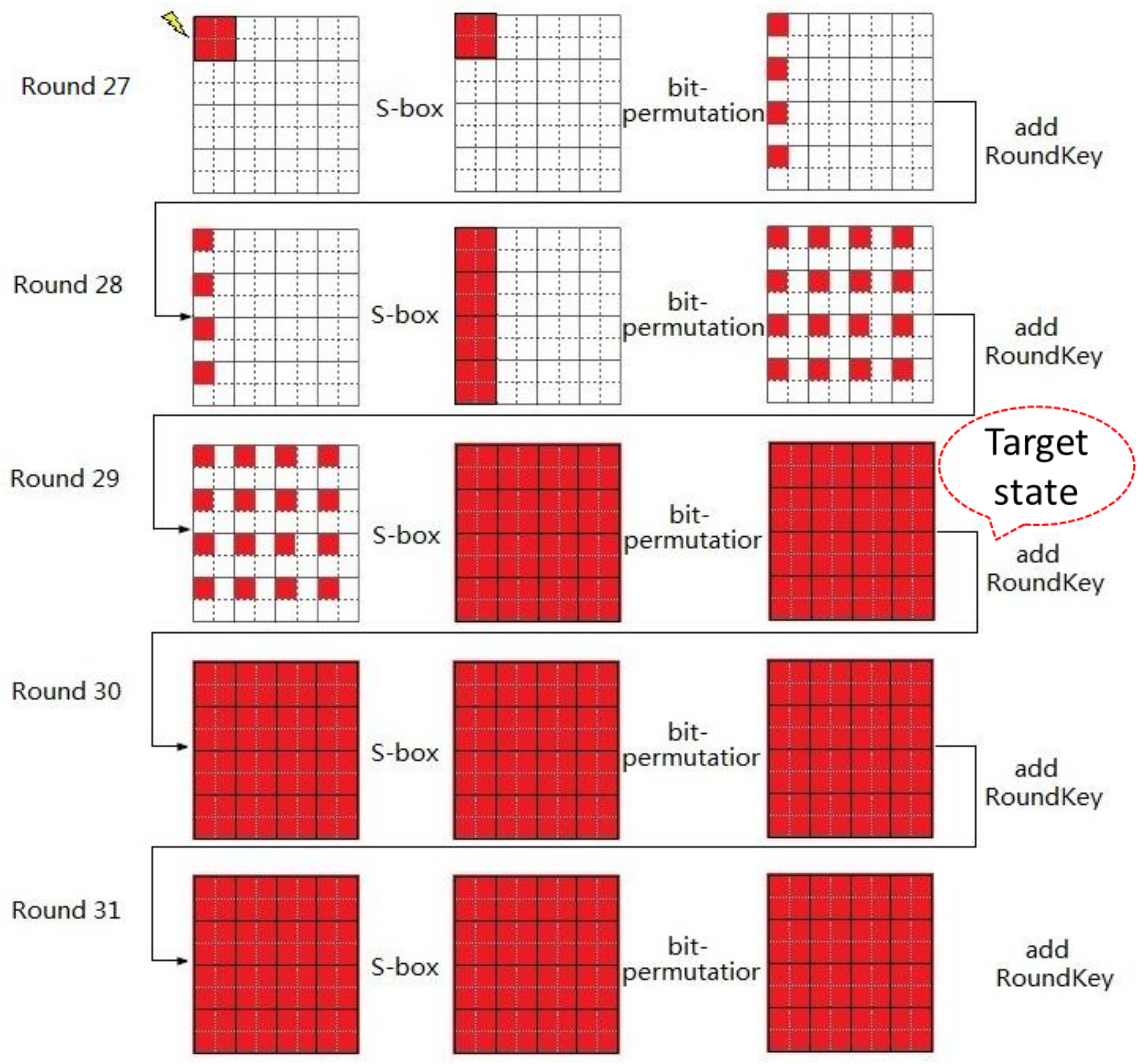


Earlier Round Fault Injection

上海交通大學
SHANGHAI JIAO TONG UNIVERSITY

Earlier
Earlier Round
Fault
Injection

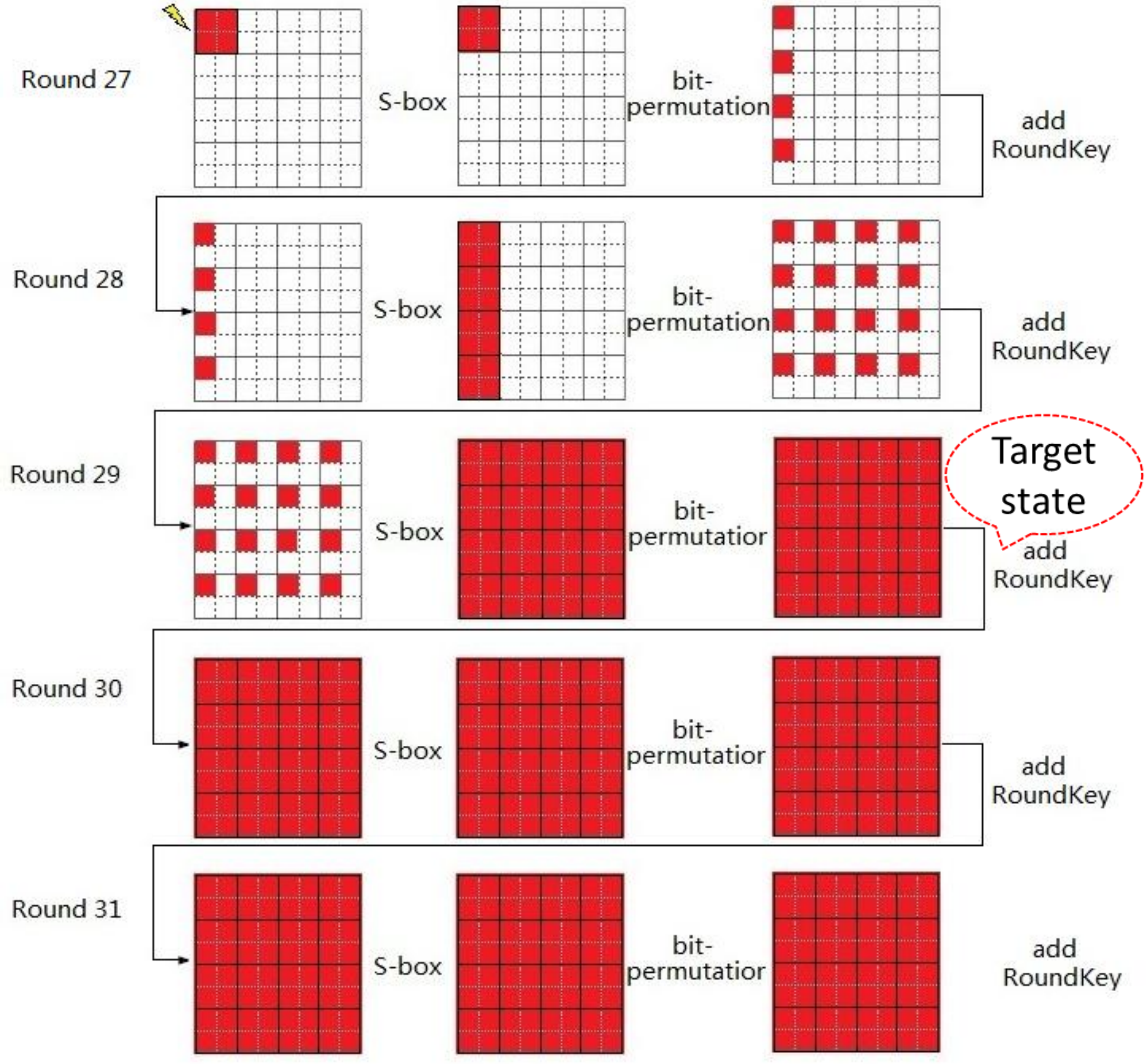
No exact
relation in
target state



密码与计算机安全实验室
Lab of Cryptology and Computer Security

上海交通大学
SHANGHAI JIAO TONG UNIVERSITY

In target state each bit has probability to be affected, but the probability is different.



Attack Details

Single Random S-box Fault Model

- Only one S-box corrupted
- The faulty S-box and faulty value is unknown and uniformly distributed
- For ciphers considered 4-bit/3-bit fault

Multi S-boxes Fault Model

- Multiple S-boxes corrupted
- The faulty S-boxes and faulty values are unknown and uniformly distributed

Attack Details

- Collect correct and faulty ciphertext pairs
- For each group of key guess partial decrypt the ciphertext pairs to get the differences at target state
- Use distinguisher to eliminate the wrong keys till only one candidate left or the practical level
- Use key schedule to recover the master key

上海交通大學
SHANGHAI JIAO TONG UNIVERSITY

Attack Details

- Build fault-based distinguisher

  $d\left(F(\mathrm{C}, \mathrm{C}^*, \mathrm{rk})\right)$ is maximal or mimimal

  - Due to the slow diffusion of bit-permutation and Wrong Key Randomization Hypothesis
- the difference distribution is non-uniform even on a subset of the penultimate or antepenultimate internal state
  - We focus on the difference for each S-box bits just before penultimate round

密码与计算机安全实验室
Lab of Cryptology and Computer Security

Attack Details

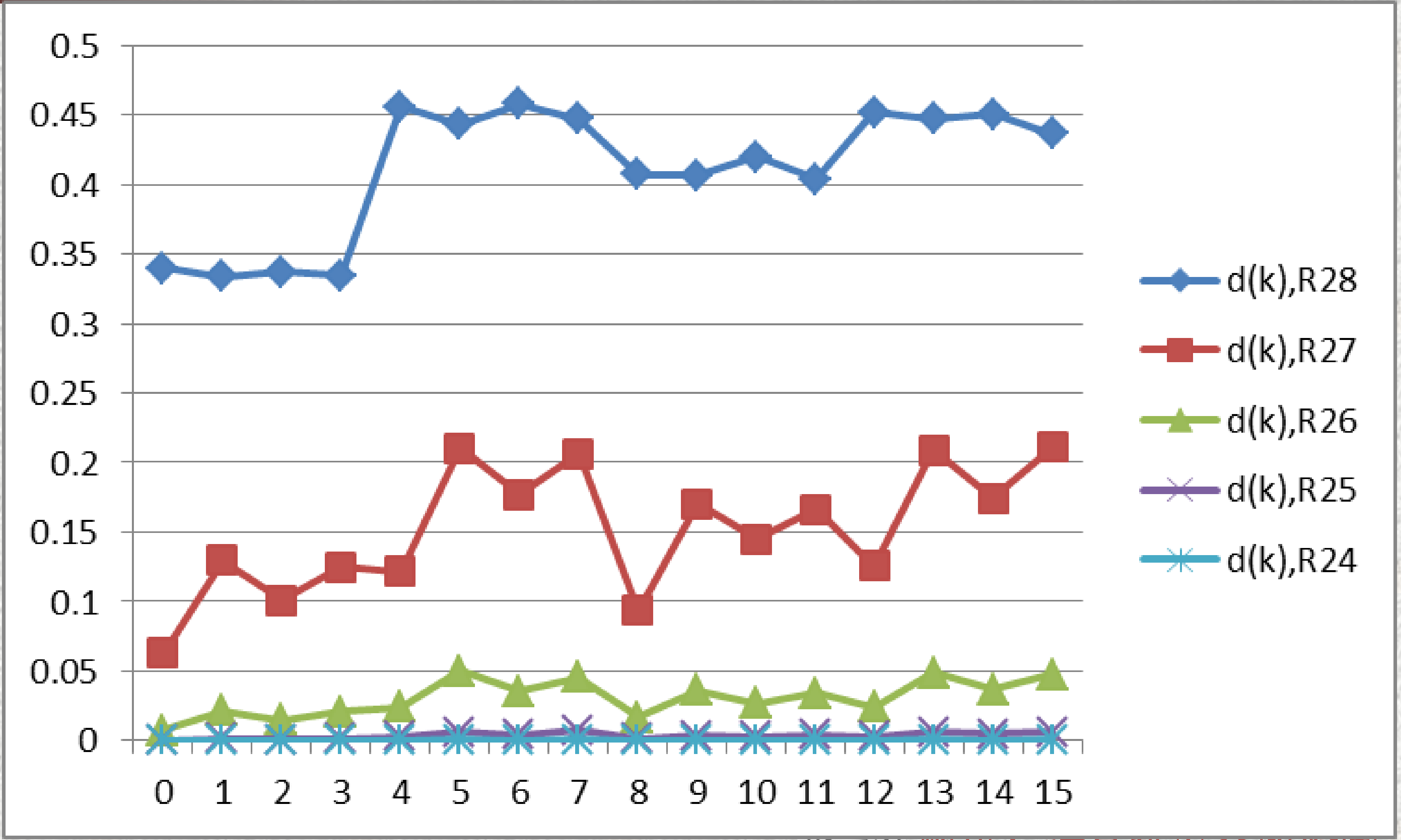
- Squared Euclidean Imbalance (SEI) distinguisher
  - Exact knowledge about the fault propagation and theoretical calculation of the distribution is hard
  - Don’t require exact distribution and simplicity consideration

$$d(k) = \sum_{\delta=0}^{2^m-1} \left(\frac{\#\{n; g_i(C_n, C^*, rk) = \delta\}}{N} - \frac{1}{2^m} \right)^2$$

Simulation Results

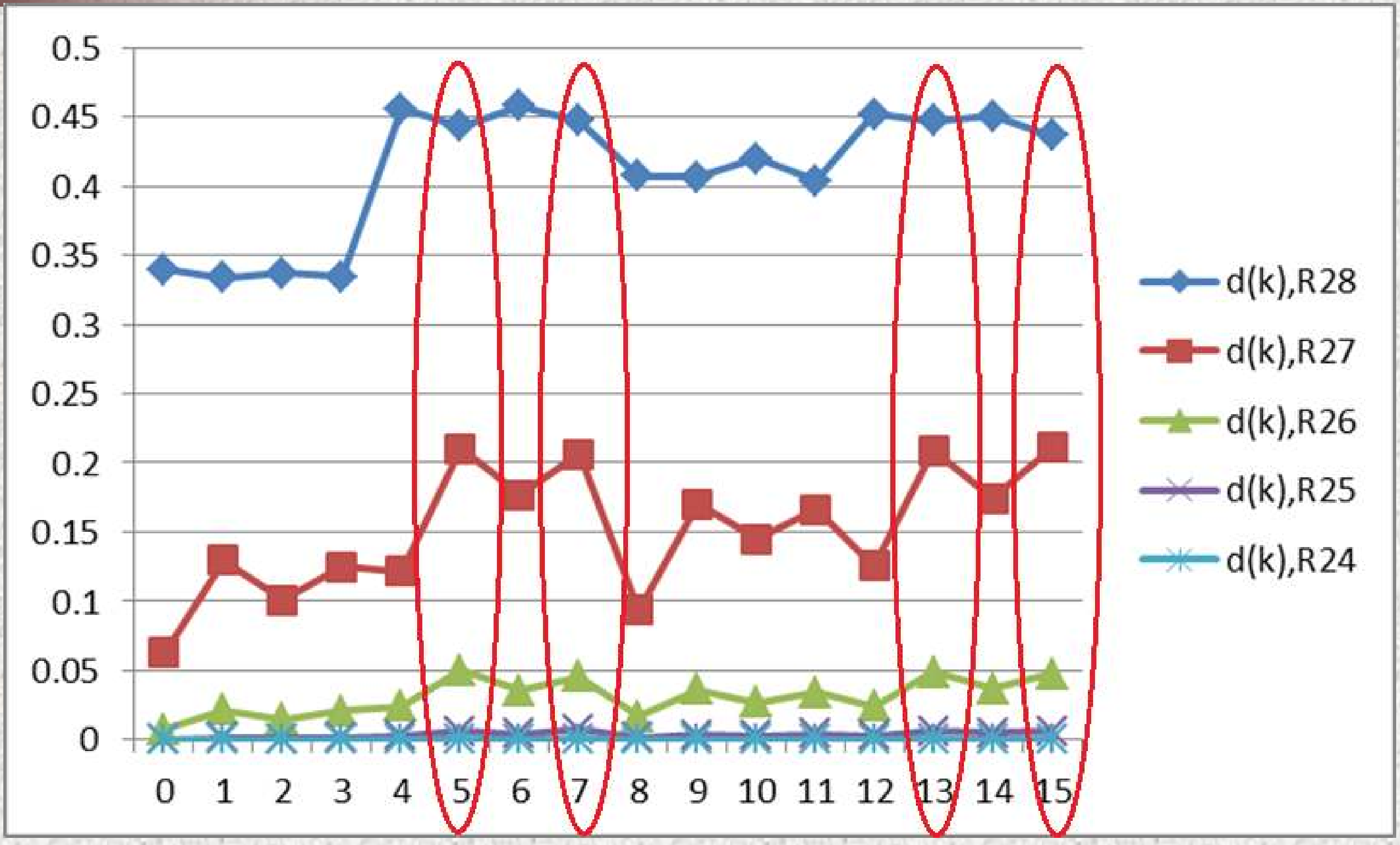
- Test 10 000 pairs of random ciphertext pairs and calculate their SEI as threshold
  - about 0.0001-0.0005
- Do fault injection simulation and calculate $d(k)$ using SEI on each nibble before penultimate round
- Complete key recover simulation

Simulation Result



上海交通大學 SHANGHAI JIAO TONG UNIVERSITY

Simulation Result



*Different fault model leads to different distribution

密码与计算机安全实验室 LoCCS Lab of Cryptology and Computer Security

Simulation Result

- Key recover simulation result
  - The correct key gives a significant high SEI value (about 0.006)
  - the average SEI is about 0.0001-0.0003 for wrong keys
  - The most significant is only about 0.0006 for wrong keys
- guess four group of each 16+4 sub-key bits
- The attack complexity is about $4 \cdot 2^{16+4} \cdot 10000 \cdot 2 = 2^{36.3}$ partial decryption.

Simulation Result

Fault injection before Round 25-28								
nibble(i)	0	1	2	3	4	5	6	7
$d(k), R_{28}$	0.1686	0.1542	0.1650	0.1538	0.2563	0.2434	0.2532	0.2409
$d(k), R_{27}$	0.0145	0.0333	0.0238	0.0334	0.0350	0.0743	0.0548	0.0691
$d(k), R_{26}$	0.0007	0.0024	0.0014	0.0024	0.0040	0.0105	0.0066	0.0103
$d(k), R_{25}$	0.0001	0.0002	0.0001	0.0002	0.0002	0.0010	0.0005	0.0008
nibble(i)	8	9	10	11	12	13	14	15
$d(k), R_{28}$	0.2224	0.1996	0.2171	0.2105	0.2553	0.2374	0.2433	0.2425
$d(k), R_{27}$	0.0232	0.0519	0.0407	0.0544	0.0371	0.0728	0.0549	0.0732
$d(k), R_{26}$	0.0023	0.0064	0.0031	0.0064	0.0042	0.0096	0.0054	0.0104
$d(k), R_{25}$	0.0003	0.0005	0.0004	0.0004	0.0003	0.0009	0.0005	0.0007

Table II

$d(k)$ FOR PRESENT DISTINGUISHER: 2 S-BOXES FAULT MODEL

上海交通大學 SHANGHAI JIAO TONG UNIVERSITY

Simulation Result

Fault injection before Round 25-28								
nibble(i)	0	1	2	3	4	5	6	7
$d(k)$								
$d(k)$								
$d(k)$								
$d(k)$								
nibb								
$d(k)$								
$d(k)$								
$d(k)$								
$d(k)$								

$d($

Fault injection before Round 26-28								
nibble(i)	0	1	2	3	4	5	6	7
$d(k), R_{28}$	0.0879	0.0822	0.0806	0.0841	0.1480	0.1385	0.1416	0.1458
$d(k), R_{27}$	0.0042	0.0121	0.0077	0.0110	0.0147	0.0316	0.0239	0.0315
$d(k), R_{26}$	0.0001	0.0003	0.0002	0.0003	0.0008	0.0033	0.0015	0.0033
nibble(i)	8	9	10	11	12	13	14	15
$d(k), R_{28}$	0.1202	0.1146	0.1154	0.1179	0.1507	0.1411	0.1388	0.1415
$d(k), R_{27}$	0.0092	0.0246	0.0138	0.0216	0.0149	0.0329	0.0226	0.0318
$d(k), R_{26}$	0.0007	0.0014	0.0006	0.0015	0.0008	0.0028	0.0016	0.0031

Table III

$d(k)$ FOR PRESENT DISTINGUISHER: 3 S-BOXES FAULT MODEL

密码与计算机安全实验室 Lab of Cryptology and Computer Security

上海交通大學
SHANGHAI JIAO TONG UNIVERSITY

Simulation Result

Fault injection before Round 25-28								
nibble(i)	0	1	2	3	4	5	6	7

Fault injection before Round 26-28								
nibble(i)	0	1	2	3	4	5	6	7

Fault injection before Round 26-28								
nibble(i)	0	1	2	3	4	5	6	7
$d(k), R_{28}$	0.0506	0.0449	0.0444	0.0431	0.0958	0.0873	0.0857	0.0868
$d(k), R_{27}$	0.0016	0.0049	0.0029	0.0045	0.0066	0.0172	0.0118	0.0159
$d(k), R_{26}$	0.0001	0.0003	0.0002	0.0003	0.0003	0.0011	0.0007	0.0011
nibble(i)	8	9	10	11	12	13	14	15
$d(k), R_{28}$	0.0749	0.0696	0.0698	0.0684	0.0959	0.0839	0.0842	0.0885
$d(k), R_{27}$	0.0040	0.0105	0.0070	0.0107	0.0062	0.0160	0.0107	0.0172
$d(k), R_{26}$	0.0002	0.0006	0.0004	0.0007	0.0002	0.0013	0.0006	0.0014

Table IV
$d(k)$ FOR PRESENT DISTINGUISHER: 4 S-BOXES FAULT MODEL

LoCCS 密码与计算机安全实验室
Lab of Cryptology and Computer Security

上海交通大學
SHANGHAI JIAO TONG UNIVERSITY

Simulation Result

PRESENT Multi S-boxes Fault Attack	
Fault S-boxes Number	Valid Attack
1	5 fault propagation + 2 partial decryption
2	4 fault propagation + 2 partial decryption
3	3 fault propagation + 2 partial decryption
4	2 fault propagation + 2 partial decryption

LoCCS 密码与计算机安全实验室
Lab of Cryptology and Computer Security

SHANGHAI JIAO TONG UNIVERSITY

Simulation Result

- Attack against PRINTcipher-48
  - almost the same as the process against PRESENT
- Differences
  - PRINTcipher uses the key-dependent permutation
- Not make attack more complex
  - the distribution keeps biased on each S-box even with 4 different secret permutation

Simulation Result

Fault injection before Round 39-43								
nibble(i)	0	1	2	3	4	5	6	7
$d(k), R_{43}$	0.2767	0.2819	0.2830	0.2746	0.2777	0.2706	0.2772	0.2759
$d(k), R_{42}$	0.1049	0.1083	0.1086	0.0966	0.0944	0.1035	0.1041	0.1013
$d(k), R_{41}$	0.0273	0.0314	0.0286	0.0253	0.0265	0.0256	0.0277	0.0237
$d(k), R_{40}$	0.0072	0.0049	0.0051	0.0061	0.0053	0.0052	0.0053	0.0041
$d(k), R_{39}$	0.0008	0.0012	0.0011	0.0006	0.0007	0.0011	0.0008	0.0010
nibble(i)	8	9	10	11	12	13	14	15
$d(k), R_{43}$	0.2738	0.2658	0.2680	0.2835	0.2661	0.2734	0.2777	0.2723
$d(k), R_{42}$	0.0987	0.0957	0.1045	0.1091	0.0946	0.0985	0.1041	0.1027
$d(k), R_{41}$	0.0261	0.0257	0.0251	0.0267	0.0257	0.0247	0.0264	0.0268
$d(k), R_{40}$	0.0046	0.0058	0.0055	0.0051	0.0049	0.0051	0.0045	0.0060
$d(k), R_{39}$	0.0008	0.0009	0.0005	0.0008	0.0009	0.0010	0.0008	0.0009

Table V

$d(k)$ FOR PRINTCIPHER DISTINGUISHER: SINGLE S-BOX FAULT MODEL

上海交通大學
SHANGHAI JIAO TONG UNIVERSITY

Simulation Result

Fault injection before Round 39-43								
nibble(i)	0	1	2	3	4	5	6	7
$d(k)$,								
$d(k)$,								
$d(k)$,								
$d(k)$,								
$d(k)$,								
nibb								
$d(k)$,								
$d(k)$,								
$d(k)$,								
$d(k)$,								
$d(k)$,								

$d(k$

Fault injection before Round 43-40								
nibble(i)	0	1	2	3	4	5	6	7
$d(k), R_{43}$	0.1094	0.0909	0.1014	0.1019	0.1005	0.0992	0.0984	0.1026
$d(k), R_{42}$	0.0261	0.0279	0.0246	0.0233	0.0221	0.0220	0.0221	0.0217
$d(k), R_{41}$	0.0045	0.0031	0.0040	0.0030	0.0028	0.0028	0.0034	0.0033
$d(k), R_{40}$	0.0003	0.0007	0.0003	0.0004	0.0009	0.0004	0.0004	0.0004
nibble(i)	8	9	10	11	12	13	14	15
$d(k), R_{43}$	0.0990	0.1030	0.1044	0.1093	0.1022	0.1034	0.0942	0.0912
$d(k), R_{42}$	0.0194	0.0190	0.0208	0.0222	0.0227	0.0215	0.0225	0.0233
$d(k), R_{41}$	0.0041	0.0030	0.0026	0.0039	0.0028	0.0032	0.0029	0.0024
$d(k), R_{40}$	0.0006	0.0005	0.0002	0.0004	0.0003	0.0005	0.0005	0.0004

Table VI
$d(k)$ FOR PRINTCIPHER DISTINGUISHER: 2 S-BOXES FAULT MODEL

上海交通大學
SHANGHAI JIAO TONG UNIVERSITY

Simulation Result

Fault injection before Round 39-43								
nibble(i)	0	1	2	3	4	5	6	7
$d(k)$,								
$d(k)$,								
$d(k)$,								
$d(k)$,								
$d(k)$,								
nibb								
$d(k)$,								
$d(k)$,								
$d(k)$,								
$d(k)$,								
$d(k)$,								

$d(k$

Fault injection before Round 43-40								
nibble(i)	0	1	2	3	4	5	6	7
$d(k), R_{43}$								
$d(k), R_{42}$								
$d(k), R_{41}$								
$d(k), R_{40}$								
nibble(i)								
$d(k), R_{43}$								
$d(k), R_{42}$								
$d(k), R_{41}$								
$d(k), R_{40}$								

$d(k)$ FOR I

Fault injection before Round 43-41								
nibble(i)	0	1	2	3	4	5	6	7
$d(k), R_{43}$	0.0443	0.0405	0.0422	0.0453	0.0426	0.0438	0.0405	0.0403
$d(k), R_{42}$	0.0064	0.0069	0.0068	0.0047	0.0049	0.0066	0.0060	0.0059
$d(k), R_{41}$	0.0007	0.0005	0.0009	0.0009	0.0008	0.0010	0.0008	0.0007
nibble(i)	8	9	10	11	12	13	14	15
$d(k), R_{43}$	0.0405	0.0383	0.0400	0.0429	0.0366	0.0402	0.0370	0.0334
$d(k), R_{42}$	0.0066	0.0055	0.0052	0.0052	0.0066	0.0052	0.0059	0.0058
$d(k), R_{41}$	0.0008	0.0008	0.0006	0.0006	0.0007	0.0005	0.0005	0.0007

Table VII
$d(k)$ FOR PRINTCIPHER DISTINGUISHER: 3 S-BOXES FAULT MODEL

密码与计算机安全实验室
Lab of Cryptology and Computer Security

上海交通大學
SHANGHAI JIAO TONG UNIVERSITY

Simulation Result

PRINTcipher-48 Multi S-boxes Fault Attack	
Fault S-boxes Number	Valid Attack
1	7 fault propagation + 2 partial decryption
2	6 fault propagation + 2 partial decryption
3	5 fault propagation + 2 partial decryption

- The attack complexity is about $5 \cdot 2^{25} \cdot 2^{11} \cdot 2 = 2^{39}$ partial decryption

Conclusion

- Differential Fault Analysis with Statistical Cryptanalysis Techniques
- Used in the lightweight block cipher with bit-permutation
- Threaten the middle rounds of the ciphers
- Useful to Multi S-boxes Fault Model
- simulation source code at https://bitbucket.org/RomanGol/faultattack

Questions?

Thank You!