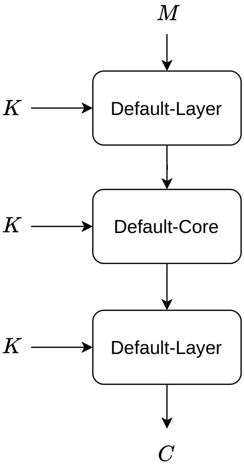
More Vulnerabilities of Linear Structure Sbox-Based Ciphers Reveal Their Inability to Resist DFA

Amit Jana[1], Anup Kumar Kundu[1], Goutam Paul[1]

[1]Indian Statistical Institute, Kolkata

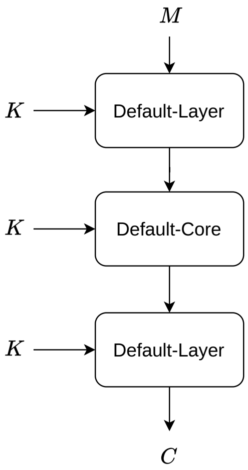
Asiacrypt
Kolkata, India
December, 2024

Default [BBB+21]



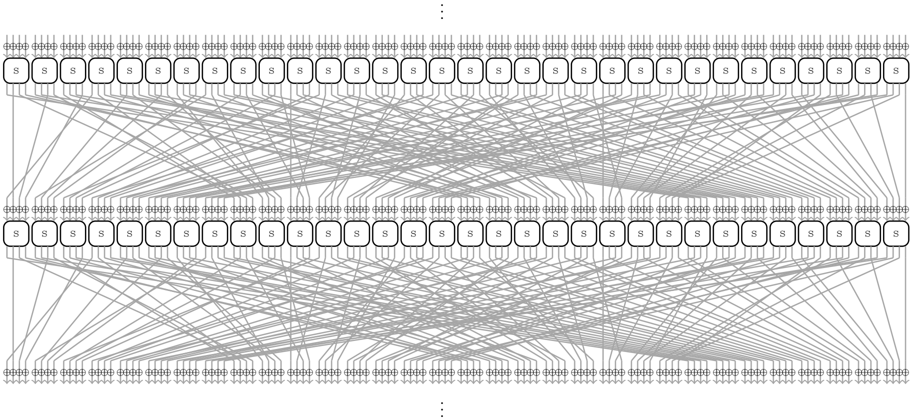
- First attempt to design a DFA immune cipher at an algorithmic level.
- Both Default-Layer and Default-Core follow the SPN structure.
- Default-Layer uses Linear Structured (LS) SBox.
- Default-Core uses non-linear SBox.
- Designers initiate the cipher as GIFT [BPP+17] like Structure.

Default [BBB+21]



- First attempt to design a DFA immune cipher at an algorithmic level.
- Both Default-Layer and Default-Core follow the SPN structure.
- Default-Layer uses Linear Structured (LS) SBox.
- Default-Core uses non-linear SBox.
- Designers initiate the cipher as GIFT [BPP+17] like Structure.
- **Initial Version (Simple Key Schedule)** [BBB+21]: Same key (128-bit) is used in each round of Default-Layer.
- **Modified Version (Rotating Key Schedule)** [BBB+21]: 4 independent keys are used in the Default-Layer.

GIFT-like Structure



Ciphers

Default

- Both state and key size are of 128 bits.
- Cipher has total of 80 rounds, 28 DEFAULT-LAYER and 24 DEFAULT-CORE.
- Each round has SBox (4-bit), permutation (bit), add round constant, and add round key layer.

Ciphers

Default

- Both state and key size are of 128 bits.
- Cipher has total of 80 rounds, 28 DEFAULT-LAYER and 24 DEFAULT-CORE.
- Each round has SBox (4-bit), permutation (bit), add round constant, and add round key layer.

Baksheesh [BBC+ 23]

- Baksheesh also follows GIFT-like structure.
- Both state and key size are of 128 bits.
- It has 35 rounds.
- Each round has SBox (4-bit), permutation layer (bit), add round constant layer, and add round key layer.
- For each round key k_i, next round key, $k_{j+1} \leftarrow k_j \ggg 1$.

Sbox Property

Linear Structure (LS)

For $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$, an element $a \in \mathbb{F}_2^n$ is called a linear structure of F, if for some constant $c \in \mathbb{F}_2^n$ and $\forall x \in \mathbb{F}_2^n$,

$$F(x) \oplus F(x \oplus a) = c.$$

Sbox Property

Linear Structure (LS)

For $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$, an element $a \in \mathbb{F}_2^n$ is called a linear structure of F, if for some constant $c \in \mathbb{F}_2^n$ and $\forall x \in \mathbb{F}_2^n$,

$$F(x) \oplus F(x \oplus a) = c.$$

- Default has 3-LS and Baksheesh has 1-LS SBox.
- For Default, DFA reduces each nibble keyspace from 2^4 to 2^2 at most, i.e. a total search complexity of $4^{32} = 2^{64}$.
- For Baksheesh, DFA reduces each keybits of SBox nibble can reduce from 2^4 to 2^1 at most, i.e. a total search complexity of 2^{32} for each round.

DDT

- **Default-Layer:**

$x:$	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
$S(x):$	0	3	7	e	d	4	a	9	c	f	1	8	b	2	6	5

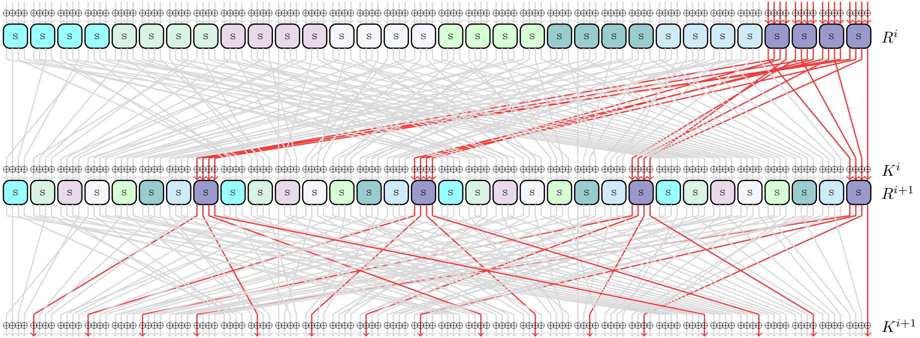
	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	16															
1				8						8						
2								8						8		
3					8										8	
4								8						8		
5					8										8	
6											16					
7				8						8						
8							8						8			
9																16
a		8										8				
b			8						8							
c		8										8				
d			8			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

- **Baksheesh:**

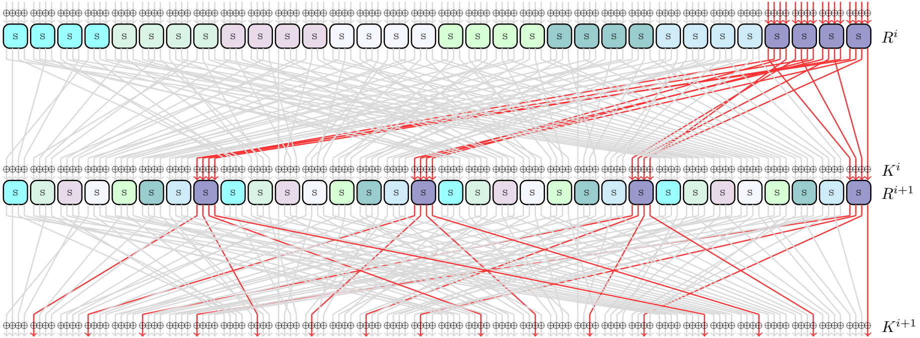
$x:$	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
$S(x):$	3	0	6	d	b	5	8	e	c	f	9	2	4	a	7	1

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

QR Structure of GIFT Permutation



QR Structure of GIFT Permutation



Advantages

- Keyspace of two consequitive rounds can be splitted according to keyspace of the QR groups.
- Do not need to guess the whole round key at once for key recovery.

Attack Procedure

Attack History

Nageler *et al.* [NDE22]

- They first showed a DFA for all key schedules by combining information through rounds.
- They expanded their DFA by inducing bit-flip faults across multiple rounds to further reduce the keyspace.
- Their strategy involved injecting differences at certain rounds and exploring all possible differential paths through subsequent rounds based on the DDT.
- For the simple key schedule, the key space reduced to around 2^{20} using 16 faults.
- We verified that, the key space can not be reduced to 1 by injecting more than 16 faults.

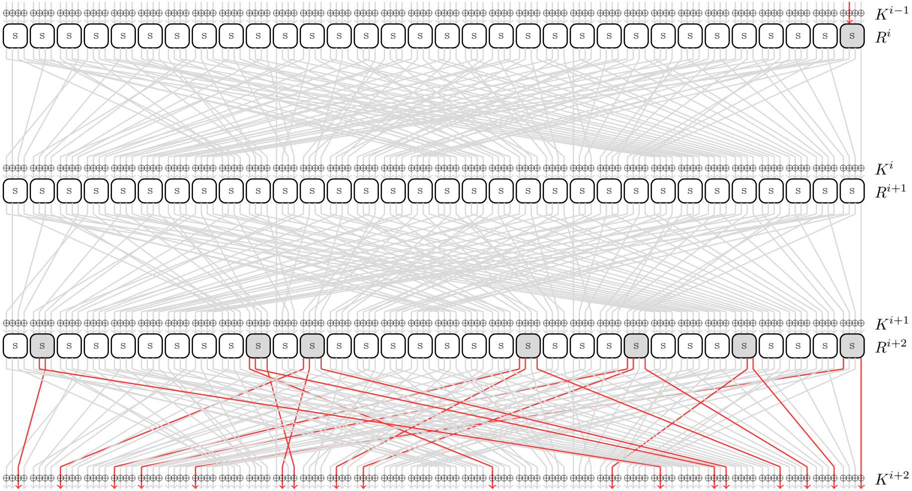
Dey *et al.* [DPR+21]

- Apply DFA on simple key schedule to reduce the key space to 2^{16} using 112 faults.
- Their attack can not be applied on the rotating key schedule.

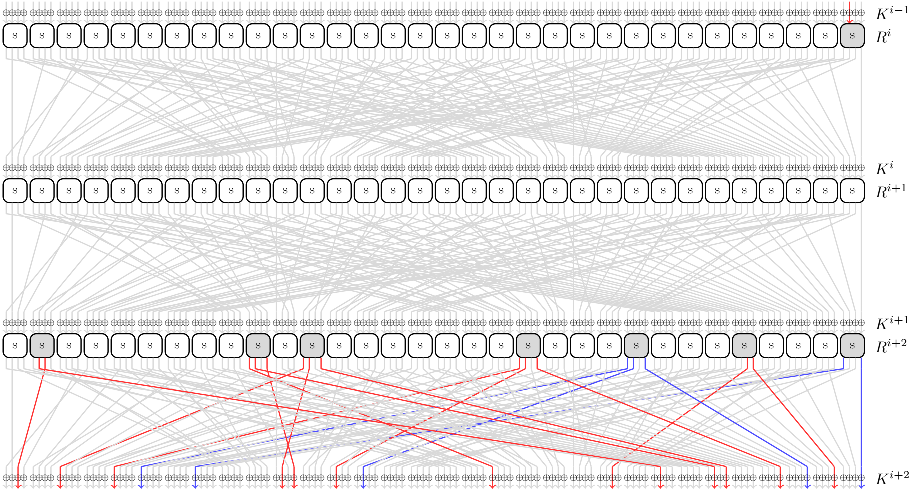
Our Contributions

- Novel technique to compute intermediate differential trails uniquely (due to fault).
- Leads to reduce the key space faster.
- Significantly reduce the number of faults when faults are injected at 5th last round.
- For GIFT-like permutation, we devise an algorithm to compute unique trail upto 5 rounds using GIFT QR structure.

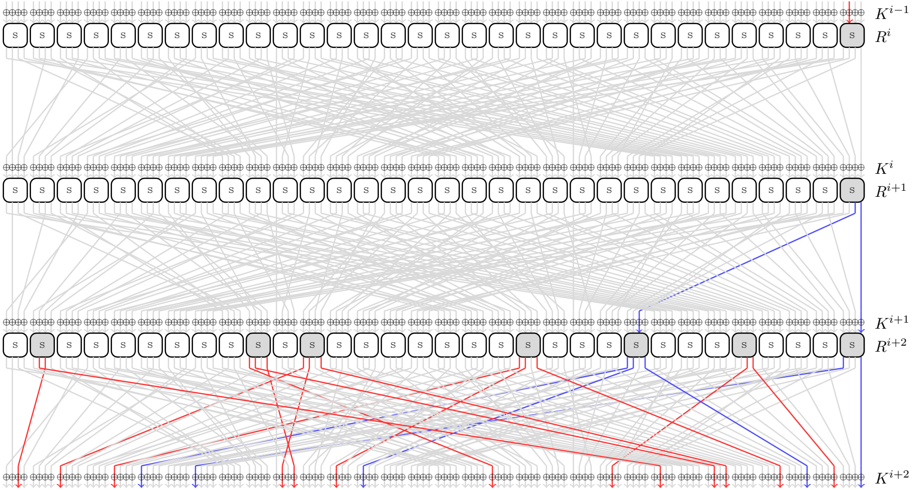
Finding Unique Trail for 3 Rounds



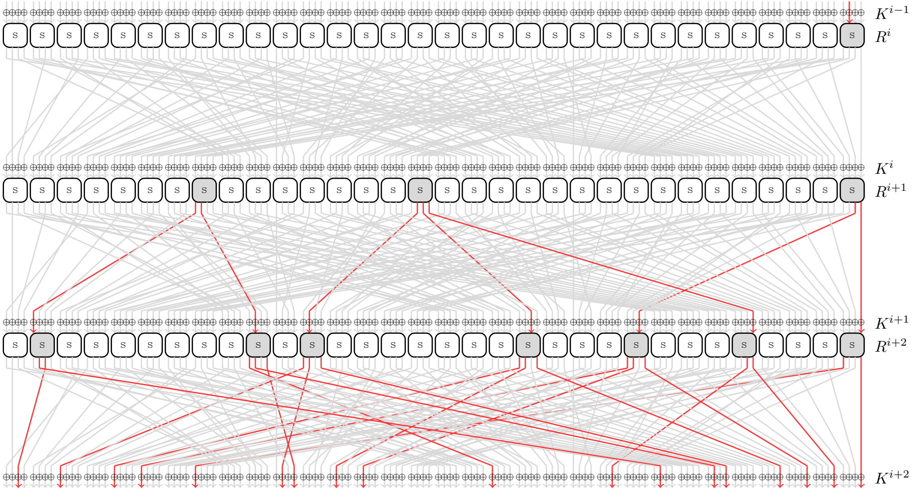
Finding Unique Trail for 3 Rounds



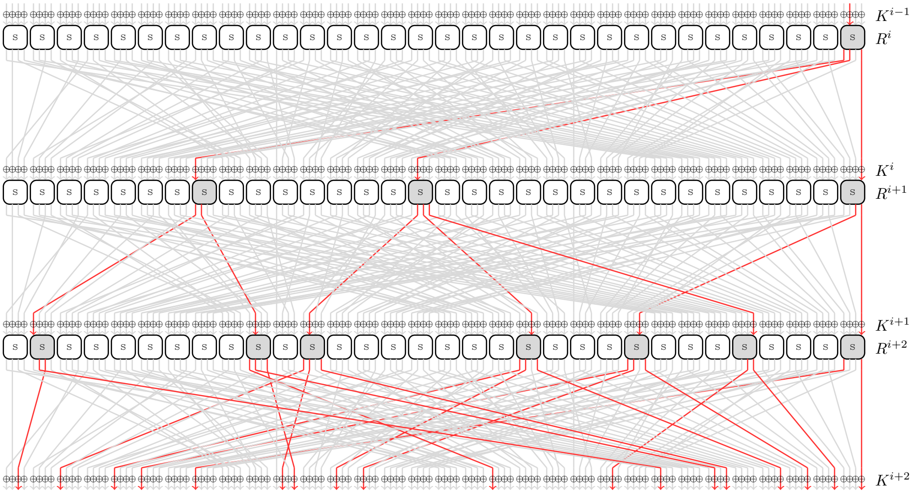
Finding Unique Trail for 3 Rounds



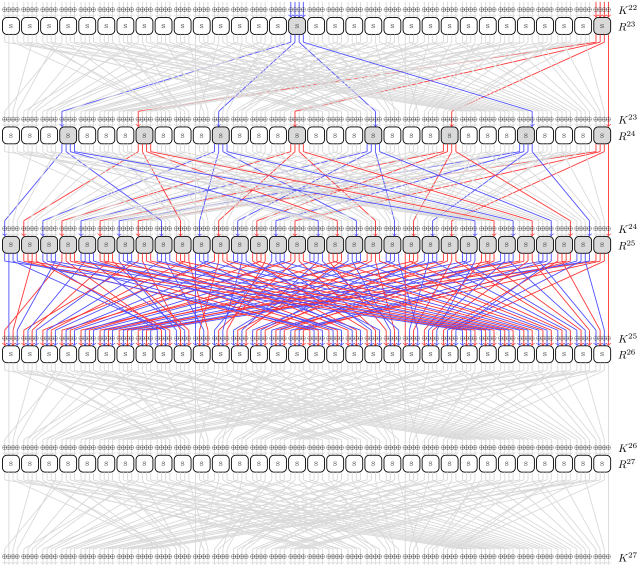
Finding Unique Trail for 3 Rounds



Finding Unique Trail for 3 Rounds

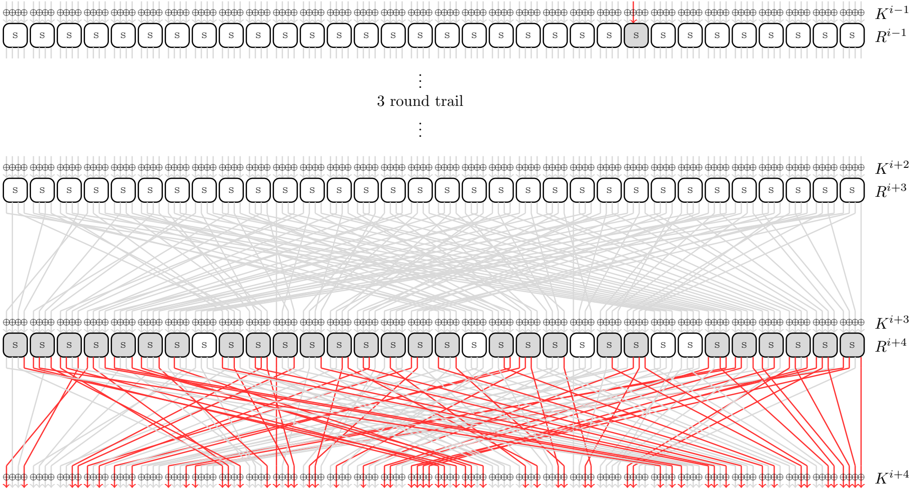


Finding Unique Trail for 5 Rounds

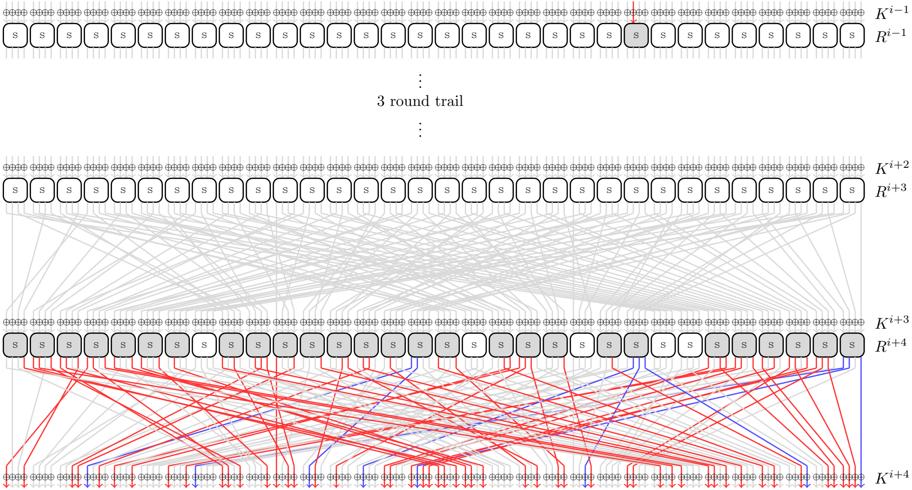


- Nibble 0-15 spread into fixed 64 bits after 3 rounds (shown in color red).
- Nibble 16-31 spread into other fixed 64 bits after 3 rounds (shown in color blue).

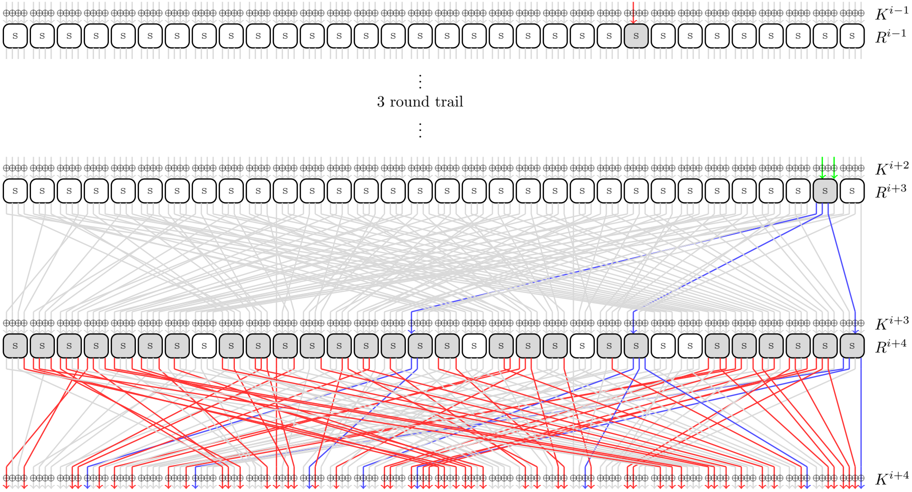
Finding Unique Trail for 5 Rounds



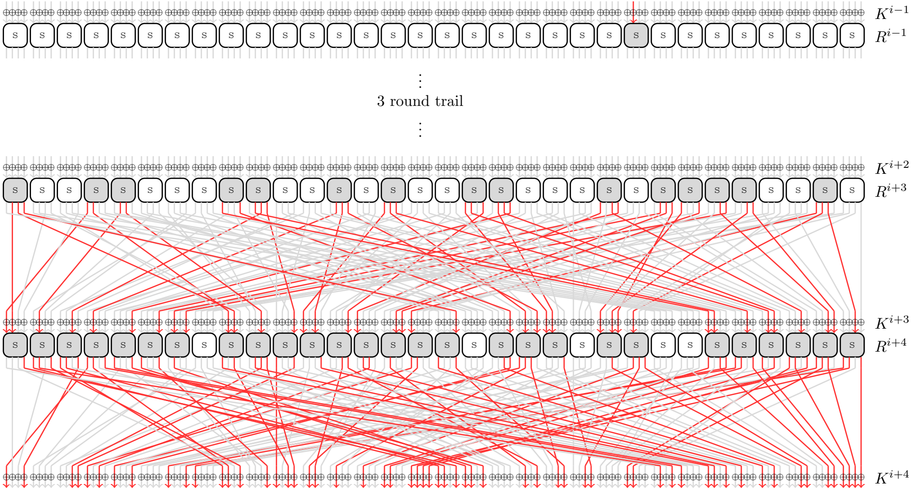
Finding Unique Trail for 5 Rounds



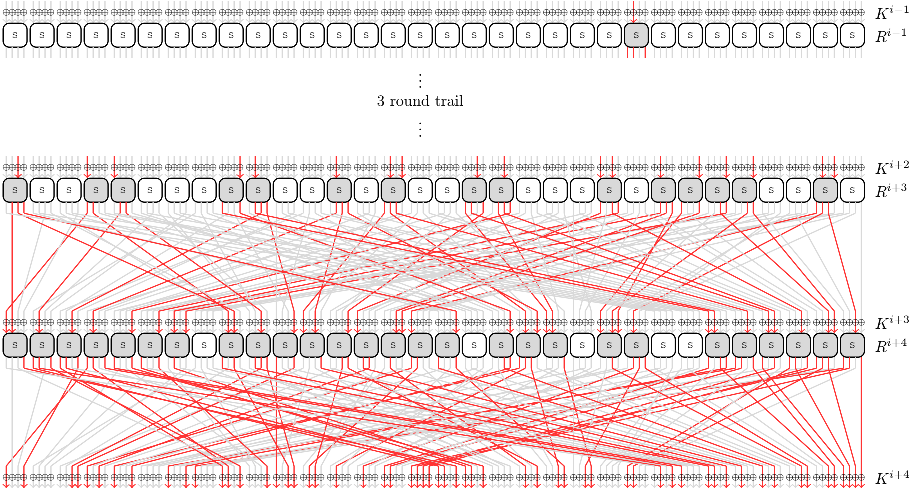
Finding Unique Trail for 5 Rounds



Finding Unique Trail for 5 Rounds



Finding Unique Trail for 5 Rounds



Key Recovery Attack

Attack Procedure

- Reduce key space of each nibble in R^{i+4} for each input-output difference from ciphertext.

Key Recovery Attack

Attack Procedure

- Reduce key space of each nibble in R^{i+4} for each input-output difference from ciphertext.
- Combine keyspace of R^{i+4} for each quotient group of R^{i+3}.

Key Recovery Attack

Attack Procedure

- Reduce key space of each nibble in R^{i+4} for each input-output difference from ciphertext.
- Combine keyspace of R^{i+4} for each quotient group of R^{i+3}.
- Combine keyspace of all even (odd) nibbles of R^{i+4} to filter all least (most) significant 16 nibbles.

Key Recovery Attack

Attack Procedure

- Reduce key space of each nibble in R^{i+4} for each input-output difference from ciphertext.
- Combine keyspace of R^{i+4} for each quotient group of R^{i+3}.
- Combine keyspace of all even (odd) nibbles of R^{i+4} to filter all least (most) significant 16 nibbles.
- Combine the whole reduced key space in R^{i+4}, we filter 4th and 5th last round.

Key Recovery Attack

Attack Procedure

- Reduce key space of each nibble in R^{i+4} for each input-output difference from ciphertext.
- Combine keyspace of R^{i+4} for each quotient group of R^{i+3}.
- Combine keyspace of all even (odd) nibbles of R^{i+4} to filter all least (most) significant 16 nibbles.
- Combine the whole reduced key space in R^{i+4}, we filter 4th and 5th last round.

Cipher	Attack Strategy	Results	
		Number of Faults	Reduced Keyspace
Default	Faults at the Second-to-Last Round	64	2^{32}
		48	2^{39}
		32	2^{46}
	Faults at the Third-to-Last Round	32	$2^{0.2}$
		28	2^{7}
		24	2^{14}
	Faults at the Fourth-to-Last Round	16	1
		12	1
		8	2^{7}
	Faults at the Fifth-to-Last Round	8	1
		6	1
		5	**1**

Attack on Rotating Key Schedule

Attack on Rotating Key Schedule

Equivalent Key Schedule

- Four independent round keys k_0, k_1, k_2, k_3 are used in four consequitive rounds of Default-Layer.

Attack on Rotating Key Schedule

Equivalent Key Schedule

- Four independent round keys k_0, k_1, k_2, k_3 are used in four consequitive rounds of Default-Layer.
- Default has 3 LS, $L(S) = \{0, 6, 9, f\}$. So, $\exists$ input-output difference (α, β) s.t. $Pr[\alpha \to \beta] = 1$.
- For any $\alpha \in L(S)$, $\exists\ \beta \in L(S^{-1}) = \{0, 5, a, f\}$ s.t. $S(x \oplus \alpha) = S(x) \oplus S(\alpha) = S(x) \oplus \beta$, $\forall x \in \mathcal{F}_2^4$.
- Define $L(S, S^{-1}) = \{(\alpha, \beta) : S(x \oplus \alpha) = S(x) \oplus \beta\} = \{(0, 0), (6, a), (9, f), (f, 5)\}$.

Attack on Rotating Key Schedule

Equivalent Key Schedule

- Four independent round keys k_0, k_1, k_2, k_3 are used in four consequitive rounds of Default-Layer.
- Default has 3 LS, $L(S) = \{0, 6, 9, f\}$. So, $\exists$ input-output difference (α, β) s.t. $Pr[\alpha \rightarrow \beta] = 1$.
- For any $\alpha \in L(S)$, $\exists\ \beta \in L(S^{-1}) = \{0, 5, a, f\}$ s.t. $S(x \oplus \alpha) = S(x) \oplus S(\alpha) = S(x) \oplus \beta$, $\forall x \in \mathcal{F}_2^4$.
- Define $L(S, S^{-1}) = \{(\alpha, \beta) : S(x \oplus \alpha) = S(x) \oplus \beta\} = \{(0, 0), (6, a), (9, f), (f, 5)\}$.
- Take the toy cipher, $y = S(x \oplus k_0) \oplus k_1$.
- If (k_0, k_1) be the actual key, then for any $(\alpha, \beta) \in L(S, S^{-1})$, $(\hat{k_0}, \hat{k_1}) = (k_0 \oplus \alpha, k_1 \oplus \beta)$ will also be an *equivalent key*.
- For a round key pair (k_0, k_1), $\exists\ 2^{64}$ (for 32 SBoxes in a round) such equivalent keys, which satisfies the same input-output difference.

k_0	$1a5f01b35ef5deea60361f4df591c654$
k_1	$5a66c55f3847aed30250237855542a124$
k_2	$85cb6b4f87f44ed160d20d713c86144f$
k_3	$84c302e5cb1539af59d623e9acdae09d$

(a) Original Keys

$\hat{k}_0$	$7c3967d53893b88c0650792b93f7a032$
$\hat{k}_1$	$96aa0993f48b621fce9cefb4998e6de8$
$\hat{k}_2$	$4907a7834b38821dac1ec1bdf04ad883$
$\hat{k}_3$	$2e69a84f61bf9305f37c894306704a37$

(b) Equivalent Keys

Attack on Rotating Key Schedule

Attack Strategy

- **Keyspace → Equivalent Keyspace**
  - Give 32 faults in the 5th last round to generate at least 2 distinct input-output differences at last, 2nd, 3rd and 4th last round.
  - Compute $\hat{k_3}$, $\hat{k_2}$, $\hat{k_1}$, $\hat{k_0}$ using 5 round trail for the last 4 rounds.
  - This reduces each $\hat{k_i}$ keyspace to 2^{64}, for $i \in \{0, 1, 2, 3\}$.

Attack on Rotating Key Schedule

Attack Strategy

- **Keyspace → Equivalent Keyspace**
  - Give 32 faults in the 5th last round to generate at least 2 distinct input-output differences at last, 2nd, 3rd and 4th last round.
  - Compute $\hat{k_3}$, $\hat{k_2}$, $\hat{k_1}$, $\hat{k_0}$ using 5 round trail for the last 4 rounds.
  - This reduces each $\hat{k_i}$ keyspace to 2^{64}, for $i \in \{0, 1, 2, 3\}$.
- **Equivalent Keyspace → An Equivalent Key**
  - Give 4 faults at 10th last round.
  - Use $\hat{k_i}$ for $i \in \{0, 1, 2, 3\}$, to recover the unique trail for 10 rounds.
  - Use key recovery procedure for the simple key schedule and $\hat{k_i}$, for $i \in \{1, 2, 3\}$ to reduce k_0 to unique one.

Attack on Rotating Key Schedule

Attack Strategy

- **Keyspace → Equivalent Keyspace**
  - Give 32 faults in the 5th last round to generate at least 2 distinct input-output differences at last, 2nd, 3rd and 4th last round.
  - Compute $\hat{k_3}$, $\hat{k_2}$, $\hat{k_1}$, $\hat{k_0}$ using 5 round trail for the last 4 rounds.
  - This reduces each $\hat{k_i}$ keyspace to 2^{64}, for $i \in \{0, 1, 2, 3\}$.
- **Equivalent Keyspace → An Equivalent Key**
  - Give 4 faults at 10th last round.
  - Use $\hat{k_i}$ for $i \in \{0, 1, 2, 3\}$, to recover the unique trail for 10 rounds.
  - Use key recovery procedure for the simple key schedule and $\hat{k_i}$, for $i \in \{1, 2, 3\}$ to reduce k_0 to unique one.
- **An Equivalent Key → Original Key**
  - Inject faults at Default-Core and recover its original key with less number of faults.

Attack on Rotating Key Schedule

Attack Strategy

- **Keyspace → Equivalent Keyspace**
  - Give 32 faults in the 5th last round to generate at least 2 distinct input-output differences at last, 2nd, 3rd and 4th last round.
  - Compute $\hat{k_3}$, $\hat{k_2}$, $\hat{k_1}$, $\hat{k_0}$ using 5 round trail for the last 4 rounds.
  - This reduces each $\hat{k_i}$ keyspace to 2^{64}, for $i \in \{0, 1, 2, 3\}$.
- **Equivalent Keyspace → An Equivalent Key**
  - Give 4 faults at 10th last round.
  - Use $\hat{k_i}$ for $i \in \{0, 1, 2, 3\}$, to recover the unique trail for 10 rounds.
  - Use key recovery procedure for the simple key schedule and $\hat{k_i}$, for $i \in \{1, 2, 3\}$ to reduce k_0 to unique one.
- **An Equivalent Key → Original Key**
  - Inject faults at Default-Core and recover its original key with less number of faults.
- **Results**
  - 36 faults are needed to reduce the keyspace of Default-Layer.
  - For Default-Core, 32 faults are required to recover the key uniquely after giving faults at 2nd last round.

SDFA

SDFA

Idea

- Combines DFA and SFA using bit-set faults.
- Objective is to identify the common nibble values that passes through both DFA and SFA.

SDFA

Idea

- Combines DFA and SFA using bit-set faults.
- Objective is to identify the common nibble values that passes through both DFA and SFA.

Example

- Consider the input-output difference $2 \rightarrow 7$ for the bit-set $u_1 = 1$ in an Sbox.
- Using DFA, $\mathcal{D} = \{0, 5, a, f, 2, 7, 8, d\}$.

SDFA

Idea

- Combines DFA and SFA using bit-set faults.
- Objective is to identify the common nibble values that passes through both DFA and SFA.

Example

- Consider the input-output difference $2 \rightarrow 7$ for the bit-set $u_1 = 1$ in an Sbox.
- Using DFA, $\mathcal{D} = \{0, 5, a, f, 2, 7, 8, d\}$.
- Using SFA, $\mathcal{I} = \{1, 5, 6, 7, 8, 9, a, e\}$.
- Hence, using SDFA, $\mathcal{Z} = \mathcal{D} \cap \mathcal{I} = \{5, a, 7, 8\}$.

SDFA

Idea

- Combines DFA and SFA using bit-set faults.
- Objective is to identify the common nibble values that passes through both DFA and SFA.

Example

- Consider the input-output difference $2 \rightarrow 7$ for the bit-set $u_1 = 1$ in an Sbox.
- Using DFA, $\mathcal{D} = \{0, 5, a, f, 2, 7, 8, d\}$.
- Using SFA, $\mathcal{I} = \{1, 5, 6, 7, 8, 9, a, e\}$.
- Hence, using SDFA, $\mathcal{Z} = \mathcal{D} \cap \mathcal{I} = \{5, a, 7, 8\}$.

Results

- For Default, [64, 128] bit set faults are required to reduce the keyspace to unique one.

Results on Baksheesh

- Similar attack can be adapted to Baksheesh.
- For Baksheesh, minimum of two faults in each nibble are needed to reduce the key nibble to one.
- In the worst case, for baksheesh, 128 bit-set faults are needed for unique key recovery.

Results on Baksheesh

- Similar attack can be adapted to Baksheesh.
- For Baksheesh, minimum of two faults in each nibble are needed to reduce the key nibble to one.
- In the worst case, for baksheesh, 128 bit-set faults are needed for unique key recovery.
- **Results:**

Cipher	Attack Strategy	Results	
		Number of Faults	Reduced Keyspace
Baksheesh	Faults at the Second-to-Last Round	48	1
		40	1
		32	2^{32}
	Faults at the Third-to-Last Round	16	1
		12	**1**
		10	2

Summary of Our Results

Cipher	Key Schedule	Relevant Works	Attack Strategy	Results: # of Faults	Results: Keyspace	References
DEFAULT	Simple	Nageler *et al.*	Enc-Dec IC-DFA	16	2^{39}	[NDE22, Section 6.1]
			Multi-round IC-DFA	16	2^{20}	[NDE22, Section 6.2]
		This Work	Second-to-Last Round Attack	64	2^{32}	**Section 3.1.2**
			Third-to-Last Round Attack	34	1	**Section 3.1.3**
			Fourth-to-Last Round Attack	16	1	**Section 3.1.4**
			Fifth-to-Last Round Attack	5	1	**Section 3.1.5**
			SDFA	$[64, 128]$	1	**Section 4.2**
	Rotating	Nageler *et al.*	Generic NK-DFA	$1728 + x$	1	[NDE22, Section 4.3]
			Enc-Dec IC-NK-DFA	$288 + x$	2^{32}	[NDE22, Section 5.1]
			Multi-round IC-NK-DFA	$(84 \pm 15) + x$	1	[NDE22, Section 5.2, 6.3]
		This Work	Third-to-Last Round Attack	$96 + x$	1	**Section 3.2.2.1**
			Fourth-to-Last Round Attack	$48 + x$	1	**Section 3.2.2.2**
			Fifth-to-Last Round Attack	$36 + x$	1	**Section 3.2.2.3**
			SDFA	$[64, 128]$	1	**Section 4.3**
BAKSHEESH	Rotating	This Work	Second-to-Last Round Attack	40	1	**Section 5.1.2**
			Third-to-Last Round Attack	12	1	**Section 5.1.3**
			SDFA	128	1	**Section 5.2**

*x represents the number of faults to retrieve the key at the Default-Core. We verified that 32 bit-faults at the second-to-last round in Default-Core achieve unique key recovery.

Conclusion

- Our approach involves constructing deterministic differential trails spanning up to 5 rounds for Default-Layer and 3 rounds for Baksheesh.
- For the simple key schedule, we demonstrate that approximately 5 bit-flip faults are sufficient to uniquely recover the key of DEFAULT.
- For rotating key schedule, we show that approximately 36 bit-flip faults are required to recover the equivalent key of DEFAULT-LAYER.
- We introduce a novel fault attack technique called SDFA, which combines both SFA and DFA.
- We apply our proposed DFA attack on BAKSHEESH, and efficiently recovered its master key uniquely.
- This computes unique 3 rounds trail for the cipher by using 12 faults only.
- Finally, Our findings underscore the difficulty in achieving DFA protection for linear-structured SBox-based ciphers.


thank you
danke
謝謝
ngiyabonga
teşekkür ederim
gracias
merci
obrigado
bedankt
dziękuję
спасибо
Баярлалаа
рахмат
go raibh maith agat
mochchakkeram
sukriya
kop khun krap
terima kasih
grazie
arigatô
dakujem
takk
tapadh leat
мерси
감사합니다
ありがとう
dank je
hvala
mauruuru
sagolun
taiku
faafetai lava
blagodaram
misaotra
matondo
paldies
grazzi
mahalo
хвала
asante
manana
obrigada
murakoze
tenki
chokrane
trugarez
shukriya
dhanyavadagalu
diolch
xiexie
rahmet
tanemirt
najis tuke
kam sah hamnida
didi madloba
mèsi
sobodi
dêkuji
enkosi
bayarlalaa
gracie
nanni
nandri
kiitos
dankie
dhanyavad
köszönöm
vinaka
спасибі
mersi
kia ora
barka
welalin
tack
spas
dankon
akun
aciu
chnorakaloutioun
gratias ago
gracies
sulpay
djiere dieuf
tau
дякую
mamnun
merce
ευχαριστώ