International Journal for Multidisciplinary Research (IJFMR)
E-ISSN: 2582-2160 ● Website: www.ijfmr.com ● Email: editor@ijfmr.com

Distinguisher for Substitution-Permutation Network (Spn)

Phadtare Vikas S[1], Prof. Patil Rahul K[2]

[1]Student, Department of Electronics and Telecommunication Baramati, India
[2]Lecturer, Department of Electronics and Telecommunication Baramati, India

Abstract:
This paper provides a detailed overview of differential cryptanalysis, with particular emphasis on its application to Substitution-Permutation Networks (SPNs). Differential cryptanalysis is a powerful attack that exploits patterns in the differences between plaintext pairs and the corresponding ciphertext pairs to reveal information about the secret key. The paper outlines the fundamental concepts of SPN structure, including the role of substitution and permutation layers, and demonstrates how differential characteristics can be leveraged to trace the propagation of differences through the cipher. By analysing the effect of these differences over multiple rounds, it becomes possible to weaken the cipher's security and reduce the search space for potential key candidates.

Keywords: Substitution-Permutation Networks (SPNs), Differential, cipher.

INTRODUCTION

The distinguisher of an SPN (Substitution-Permutation Network) cipher is crucial because it serves as a measure of the cipher's strength and security. In cryptography, the ideal goal is for a cipher to produce output that is indistinguishable from a truly random permutation, ensuring that attackers cannot detect any patterns or weaknesses. A distinguisher, however, identifies non-random behavior in the cipher's output, which could be leveraged to gain insights into the cipher's structure or even break it. The existence of a distinguisher suggests that the cipher might not be as secure as intended, highlighting potential vulnerabilities that could be exploited in cryptographic attacks.

In the context of SPN ciphers, where the design aims to create a strong diffusion and con-fusion of input data, the existence of a distinguisher indicates a flaw in the design's ability to achieve these goals. This can prompt further investigation and refinement of the cipher's structure, leading to improved versions that are more resistant to cryptanalysis. Understanding and addressing distinguishers is essential for cryptographers, as it helps ensure that the ciphers they develop can withstand sophisticated attacks and provide robust security in real-world applications.

Differential cryptanalysis is a crucial technique used to distinguish and analyse ciphers by examining how differences in input pairs influence the resulting differences in output pairs after encryption. This method involves selecting specific input differences and observing how they propagate through the rounds of the cipher. By identifying patterns or predictable behaviors in the output differences, cryptanalysts can detect deviations from what would be expected in a truly random permutation. These deviations serve as distinguishers, indicating that the cipher may exhibit non-random characteristics that could be analysed further.

This analysis is particularly effective against certain ciphers, especially those based on Substitution-Permutation Networks (SPNs). If a cipher shows consistent or predictable behavior when processing specific input differences, it suggests that the cipher might not be fully randomizing the data as intended. Differential cryptanalysis thus plays a key role in evaluating the security of ciphers by revealing potential weaknesses in their design, particularly in their ability to diffuse and obscure input data effectively. Understanding how differential cryptanalysis can distinguish a cipher is essential for developing and refining cryptographic algorithms that resist such analytical techniques.

Grassi et al. introduced subspace trail cryptanalysis as an extension of invariant subspaces, which they used to develop the first five-round distinguisher for AES. Although this method is broadly applicable, it has so far only been used for AES and PRINCE. A key challenge hindering wider adoption of this technique has been the lack of a general analysis algorithm.

In this work, we aim to provide efficient and general algorithms that enable the computation of the provably optimal subspace trails for any substitution-permutation cipher.

LITERATURE SURVEY

Differential Cryptanalysis of the Data Encryption Standard [8]. In this paper, a differential attack on DES(Data encryption System) involves examining how specific differences in input pairs affect the differences in the resulting ciphertexts. By analysing how these differences propagate through DES's rounds, particularly through its Sboxes and Feistel function, attackers can identify patterns in the output. These patterns reveal deviations from random behavior, and a distinguisher can be found. Grassi et al., Subspace Trail Cryptanalysis and its Applications to AES [4] The paper introduces Subspace Trail Cryptanalysis, a new method for analysing block ciphers like AES. This technique focuses on identifying trails in a specific subspace of the cipher's state space, enabling a more efficient analysis of its security. The method is used to assess the resilience of AES against various attacks by exploring the propagation of certain patterns (trails) through its rounds. The findings highlight potential weaknesses in AES and offer insights into how these vulnerabilities could be exploited in practice. The paper demonstrates the practical applications of this technique, showing its effectiveness in enhancing cryptanalytic methods. The paper [5] breaks PRINT cipher using the Invariant Subspace Attack by exploiting structural weaknesses. It enables full key recovery more efficiently than brute force. The paper [2] introduces the Nonlinear Invariant Attack, a new cryptanalytic technique. It is applied successfully to break full SCREAM, iSCREAM, and Midori64 under certain weak-key assumptions. The attack leverages nonlinear invariant properties that persist through the cipher rounds. These results raise concerns about the use of involutive and low-algebraic-degree components in cipher design. The paper [9] analyzes the AEAD cipher ASCON using truncated, impossible, and improbable differential techniques. It improves understanding of ASCON's differential properties without threatening its full-round security. The authors explore differential trails in the permutation to find distinguishers. These results guide future design and analysis of sponge-based ciphers. The paper [6] studies how undisturbed bits in S-boxes relate to other cryptographic properties like differential uniformity and nonlinearity. It shows that undisturbed bits can signal structural weaknesses in S-box design. The authors establish bounds and relationships between these properties. The results help in evaluating and designing stronger S-boxes.

PROPOSED SYSTEM ARCHITECTURE

We are planning to provide an input difference to the SPN cipher along with sufficient plaintext, and then

we will determine the difference set after one round. We will identify the subspace generated by this difference set and repeat the procedure for several rounds. We will stop when the dimension of the subspace becomes full. And then we will distinguish the cipher.

Preliminaries

By we $\mathbb{f}_2$ denote the finite field with two elements and by $\mathbb{f}_2^n$ the ndimensional vector space over $\mathbb{f}_2$.

Definition[Derivative]: Let F : $\mathbb{f}_2^n \to \mathbb{f}_2^n$ The derivative of F in direction α is defined as

$$\Delta_\alpha(F(x)) = F(x) + F(x + \alpha).$$

Definition- [S-box layer]: Let F : $\mathbb{f}_2^n \to \mathbb{f}_2^n$ be an S-box. Then S-box layer F^k is the parallel application of F for k times

$$F^k: (\mathbb{f}_2^n)^k \to (\mathbb{f}_2^n)^k$$

$$F^k (x_1, x_2, \cdots x_k) = (F(x_1), F(x_2), \cdots F(x_k)).$$

Definition[Subspace Trail]: Let F : $\mathbb{f}_2^n \to \mathbb{f}_2^n$. Linear subspaces U, V $\subseteq \mathbb{f}_2^n$ are called a(one round) subspace trail,

$$\forall a : \exists b : F(U + a) \subseteq V + b$$

We denote this by U → V .

An r + 1 -tuple of subspaces (U_1, ..., $Ur_{+1)}$ is called a subspace trail (over r rounds),

$$\text{if } U_i \to U_{i+1} \forall i \in \{1, 2, \cdots r + 1\}.$$

SOME MATHEMATICAL RESULT

Lemma :

Given U → V be a subspace trail. Then

$$\forall u \in U : Im(\Delta u(F)) \subseteq V$$

Moreover, for any subspace U $\subseteq \mathbb{f}_2^n$ it holds that

$$U \to \text{span } (\cup_{u \in U} Im(\Delta u(F)))$$

Proof. Let u ∈ U. Because U → V is a subspace trail for F, for any x ∈ $\mathbb{f}_2^n$ both, x and x+u, are in a coset U +x of U.

Due to the subspace trail, they get mapped to a coset of V : F(x), F(x + u) ∈ V +b.

Therefore, their sum is again in V: F(x) + F(x + u) ∈ V.

Computing A Trail for a Given Input Difference

A starting point for finding subspace trails is: Given an initial subspace, how to compute the resulting trail? One approach is based on Lemma 1. In order to compute V , we have to compute the images of the derivatives of F in direction U. To speed this up, we can exploit two facts. First, when choosing x ∈ $\mathbb{f}_2^n$, assuming a random behavior, it is sufficient to take slightly more than n many x to compute the subspace spanned by the image. Second, we do not need to compute the image of every element in U; instead it is

enough to take a basis of U, see the following lemma.

Given $U \subseteq \mathbb{F}_2^n$ and a basis $b_1, ..., b_k$ of U, then span $(\cup_{u \in U} Im(\Delta_u(F)) = span (\cup_{0<i<k+1} Im(\Delta_{bi}(F))$.

Proof. It is clear that the set on the right side is a subset of the set on the left side of the equation. Thus, we are left with showing that the left side is a subset of the right side. Moreover, as we consider the linear span on both sides, it suffices to show that any

$v \in \cup_{u \in U} Im(\Delta_u(F)$ is contained in span $(\cup_{0<i<k+1} Im(\Delta_{bi}(F))$.

We will prove this by induction over the dimension of U. The case dim(U) = 1 is trivial. Now assume that the statement is correct for any U' of dimension smaller than k. We consider

$$v \in Im(\Delta u(F))$$

for $u \in U$. That is, there exist an element $x \in \mathbb{F}_2^n$ such that

$$v = \Delta u(F)(x) = F(x) + F(x + u).$$

As the bi form a basis of U, we can express u as a linear combination of the bi , that is

$$u = \sum_{i=1}^{k} \lambda i \, bi$$

for suitable $\lambda i \in F_2$. Thus

$$v = F(x) + F(x + \sum_{i=1}^{k} \lambda ibi)$$

By defining $x' = x + \lambda_k b_k$ we get

$$v = F(x) + F(x + \sum_{i=1}^{k} \lambda ibi)$$

$$v = F(x' + \lambda_k b_k) + F(x' + \sum_{i=1}^{k} \lambda ibi)$$

$$v = F(x' + \lambda_k b_k) + F(x') + F(x') + F(x' + \sum_{i=1}^{k} \lambda ibi)$$

$$= \lambda_k \Delta b_k (F)(x') + \lambda' \Delta u'(F)(x')$$

Where $\lambda' = \bigvee_1^{k-1} \lambda i$ $u' = \sum_{i=1}^{k-1} \lambda ibi$

Thus $v \in$ span $(Im(\Delta_{b1} (F)) \cup Im(\Delta_{u'}(F)))$,

and the lemma follows by induction as u′ is contained in a (k − 1) dimensional subspace

$$U' = \text{span } \{b_1, \cdots b_k\}$$

Assembling the above observations, we get the recursive Algorithm 1 to compute the optimal subspace trail for a given starting subspace U

Algorithm

Given: A nonlinear, bijective function F: $\mathbb{F}_2^n \rightarrow \mathbb{F}_2^n$ and a subspace U

1. function Compute Trail (F, U)
2. if dimension(U) = n then
   return the U
3. V ← ϕ
4. for bi basis vectors of U do
5. for enough x ∈ random $\mathbb{F}_2^n$ do
6. V ← V ∪△b_i (F)(x)
7. V ← span (V)
8. return U + Find Trail (F, V)

Details of Some SPN Ciphers

a. Present

Present is a lightweight block cipher proposed by Bogdanov et al. The block size and key size are 64 and 80 bits, respectively. The round function consists of three operations, namely AddRoundKey, sBoxlayer and pLayer which are described as follows.

1. AddRoundKey: This step XORs a 64-bit subkey and a round constant to the state.
2. sBoxlayer: It applies the 4-bit PRESENT S-box in parallel to 16 4-bit inputs. The 4-bit S-box of present in hexadecimal notation is given in Table 1.

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

Table 1: present S-box

3. pLayer: It takes a 64-bit state as an input and applies the PRESENT bitwise permutation P to obtain a 64-bit state. The bitwise permutation P maps i-th bit of state to the bit position P(i) as shown below in Table 2.

i	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$P(i)$	0	16	32	48	1	17	33	49	2	18	34	50	3	19	35	51
i	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
$P(i)$	4	20	36	52	5	21	37	53	6	22	38	54	7	23	39	55
i	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47
$P(i)$	8	24	40	56	9	25	41	57	10	26	42	58	11	27	43	59
i	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
$P(i)$	12	28	44	60	13	29	45	61	14	30	46	62	15	31	47	63

Table 2: present Permutation

b. GIFT

GIFT is a lightweight block cipher proposed by Banik et al. at CHES'17. It has two variants, namely GIFT-n where n ∈ {64, 128} is the block size in bits and the key size is 128 bits for both versions. The round function consists of three operations, namely Subcells, Permbits, and AddRoundKey which are described as follows.

1. Subcells: It applies the 4-bit GIFT S-box in parallel to n/4 many 4-bit inputs. The 4-bit S-box of GIFT in hexadecimal notation is given in Table 3.

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

Table 3: Gift S-box

2. Permbits: It takes an n-bit state as an input and applies the GIFT bitwise permutation Pn to obtain an n-bit state. GIFT-n uses a bitwise permutation Pn. It maps i-th bit of the state to bit position Pn(i), ∀i ∈ {0, 1, . . . , n−1}. For n = 64 and 128, the permutations P_{64} and P_{128} are given in Table 4 and Table 5.

i	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$P_{64}(i)$	0	17	34	51	48	1	18	35	32	49	2	19	16	33	50	3
i	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
$P_{64}(i)$	4	21	38	55	52	5	22	39	36	53	6	23	20	37	54	7
i	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47
$P_{64}(i)$	8	25	42	59	56	9	26	43	40	57	10	27	24	41	58	11
i	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
$P_{64}(i)$	12	29	46	63	60	13	30	47	44	61	14	31	28	45	62	15

Table 4: Gift-64 Permutation

i	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$P_{128}(i)$	0	33	66	99	96	1	34	67	64	97	2	35	32	65	98	3
i	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
$P_{128}(i)$	4	37	70	103	100	5	38	71	68	101	6	39	36	69	102	7
i	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47
$P_{128}(i)$	8	41	74	107	104	9	42	75	72	105	10	43	40	73	106	11
i	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
$P_{128}(i)$	12	45	78	111	108	13	46	79	76	109	14	47	44	77	110	15
i	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79
$P_{128}(i)$	16	49	82	115	112	17	50	83	80	113	18	51	48	81	114	19
i	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95
$P_{128}(i)$	20	53	86	119	116	21	54	87	84	117	22	55	52	85	118	23
i	96	97	98	99	100	101	102	103	104	105	106	107	108	109	110	111
$P_{128}(i)$	24	57	90	123	120	25	58	91	88	121	26	59	56	89	122	27
i	112	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127
$P_{128}(i)$	28	61	94	127	124	29	62	95	92	125	30	63	60	93	126	31

Table 5: Gift-128 Permutation

3. AddRoundKey: This step XORs a n/2-bit subkey to the state followed by a round constant addition.
   - Number of Rounds: 28 for Gift-64 and 40 for Gift-128.

c. DEFAULT

The details the 128-bit version of proposed DFA protecting layer (DEFAULT-LAYER). It can be used to encrypt 128-bit block.

1. SubCells : It uses the 4-bit S SBox.
   The 4-bit S-box of DEFAULT in hexadecimal notation is given in Table 6.

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

Table 6: DEFAULT S-box

This SBox is applied to every nibble of the state.

2. PermBits: The bit-permutation is the same as the permutation P128 in GIFT-128.
3. Rounds : 16

Application

In the table below, we have mentioned the number of rounds covered by the algorithm. It provides information on how many rounds of the cipher are distinguishable.

1. **Application for Present**
   Here, we provided a subspace trail for the present cipher with the following subspace.
   0x1: 3 - 15 - 63 - 64
   we observed that for the subspace 0x1, it covers 3 rounds of the block cipher, meaning this cipher is distinguishable up to 3 rounds. For the other subspaces, we obtained coverage for less than 3 rounds, so we did not consider them.
2. **Application for Gift-64**
   0x1: 3 - 15 - 63 - 64
   Gift-128
   Ox1: 4 - 15 - 60 - 128
3. **Application for DEFAULT**
   0x1: 2 - 5 - 15 - 42 - 74 - 106 – 128

Ciphers	Rounds
Gift-64	3
Gift - 128	3
Present	3
DEFAULT	6

Conclusion

In this work, we consider a subspace generated by a given input difference. Using a sufficient number of plaintexts and a fixed difference, we identify the corresponding differential trail. We repeat this process for several rounds, and once we obtain the full dimension of the subspace, we stop. At that point, we say that the cipher is distinguishable. In this study, we apply this method to the PRESENT, GIFT-64, and GIFT-128, Default block ciphers, and we provide their corresponding trails.

References

1. Anubhab Baksi and Anubhab Baksi. Default: Cipher-level resistance against differential fault attack. Classical and Physical Security of Symmetric Key Cryptographic Algorithms, pages 177–216, 2022.
2. Jung Hee Cheon and Tsuyoshi Takagi. Advances in Cryptology–ASIACRYPT 2016: 22nd International Conference on the Theory and Application of Cryptology and Information Security, Hanoi, Vietnam, December 4-8, 2016, Proceedings, Part II, volume 10032. Springer, 2016.

3. Wieland Fischer and Naofumi Homma. Cryptographic Hardware and Embedded Systems–CHES 2017: 19th International Conference, Taipei, Taiwan, September 25-28, 2017, Proceedings, volume 10529. Springer, 2017.
4. Lorenzo Grassi, Christian Rechberger, and Sondre Rønjom. Subspace trail cryptanalysis and its applications to aes. Cryptology ePrint Archive, 2016.
5. Gregor Leander, Mohamed Ahmed Abdelraheem, Hoda AlKhzaimi, and Erik Zenner. A cryptanalysis of printcipher: The invariant subspace attack. In Advances in Cryptology–CRYPTO 2011: 31st Annual Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2011. Proceedings 31, pages 206–221. Springer, 2011.
6. Rusydi H Makarim and Cihangir Tezcan. Relating undisturbed bits to other properties of substitution boxes. In International workshop on lightweight cryptography for security and privacy, pages 109–125. Springer, 2014.
7. Pascal Paillier. Cryptographic Hardware and Embedded Systems-CHES 2007: 9th International Workshop, Vienna, Austria, September 10-13, 2007, Proceedings, volume 4727. Springer Science & Business Media, 2007.
8. Data Encryption Standard et al. Data encryption standard. Federal Information Processing Standards Publication, 112(3), 1999.
9. Cihangir Tezcan. Truncated, impossible, and improbable differential analysis of ascon. In International Conference on Information Systems Security and Privacy, volume 2, pages 325– 332. SciTePress, 2016.