Singularization: A New Approach to Designing Block Ciphers for Resource-Constrained Devices

Gilles Macario-Rat[1][0000−0002−6156−8295] and Mihail-Iulian Plesa[2][0000−0001−5954−7199]

[1] Orange, Chatillon, France
gilles.macariorat@orange.com
[2] Orange Services, Bucharest, Romania
mihail.plesa@orange.com

Abstract. Running traditional symmetric encryption algorithms, such as AES, on resource-constrained devices presents significant challenges due to the limited computational resources available. A common bottleneck in these algorithms is the number of rounds, which is typically determined through cryptanalysis efforts. In this paper, we introduce a novel framework for designing block ciphers, termed Singularization. This framework is based on a generic Feistel network with dynamically generated pseudorandom functions (PRFs). We demonstrate that Singularization may enable the design of symmetric ciphers with fewer rounds without compromising security. This is evidenced by a case study of a 6-round DES, which is vulnerable to differential cryptanalysis attacks. By redesigning DES using our framework, we mitigate this vulnerability, suggesting that it is possible to achieve almost the same level of security as a full-round DES with a reduced number of rounds.

Keywords: Cryptography · Moving Target Defense · Block Cipher

1 Introduction

Conventional algorithms, such as AES, introduce significant overhead when executed on resource-constrained devices [2]. To address this issue, lightweight cryptography has been developed with the objective of designing new ciphers that can be efficiently implemented on these devices [3]. A critical factor in symmetric algorithms is the number of rounds, which directly impacts the computational resources required by the cipher. This parameter is typically established following extensive cryptanalysis to ensure that potential attacks are effectively mitigated. Notably, two prevalent attacks on symmetric ciphers are linear cryptanalysis and differential cryptanalysis [4]. For instance, in the case of AES, the authors determined the number of rounds by considering various cryptanalysis techniques, including linear, differential, truncated differential, and integral cryptanalysis [8]. The number of rounds was set to the maximum number of rounds for which any such attack is applicable, plus an additional security margin. Naturally, by exploring various methods to mitigate these attacks, the efficiency of the cipher can be directly improved by reducing the number of rounds.

Moving Target Defense (MTD) is a recently proposed philosophy for improving systems security in general. The motivation behind MTD is that the static nature of some systems, e.g., hardware, well-known IP addresses, etc., gives an attacker time to find and try different ways to counterattack the security measures put in place. MTD tries to mitigate this by reducing the attacker's time window. Although numerous MTD approaches exist for network security, such as IP shuffling, port hopping, and programming language diversity [7], there are relatively few MTD applications in cryptographic systems. One such example is switching between multiple cryptosystems to reduce the success probability of brute force attacks, particularly on resource-constrained devices [6]. However, this approach can be difficult to implement on legacy systems and requires the deployment of numerous cryptosystems to enhance security, which is challenging for resource-constrained devices.

In this paper, we propose Singularization, a novel framework for designing block ciphers based on MTD principles. The core concept of our method is to employ a dynamically generated pseudorandom function (PRF) for each round of a Feistel network. Previously used to improve security applets on legacy SIM cards, Singularization now generalizes to entire families of symmetric cryptographic algorithms [9].

2 Related works

The concept of incorporating dynamic elements into the structure of a cipher is well-established, though these elements are typically associated with the input data rather than the algorithm itself. Similar Feistel-based methods generally employ a limited number of pseudorandom functions (PRFs), such as 2 or 3, which are combined in a predetermined manner during each execution of the algorithm. Intuitively, we say that a cipher has a static structure if the algorithms remain the same at different runs. Another way of thinking of this is that the cipher can be implemented without any control structures such as `IF-THEN` statements.

Skipjack is a block cipher designed as an unbalanced Feistel network [15]. The cipher executes distinct operations in even and odd rounds. During even-numbered rounds, it employs a substitution-permutation network referred to as an A-cycle. In odd-numbered rounds, it utilizes a different substitution-permutation network known as a B-cycle. Despite these variations in operations based on the round number, the overall algorithm remains static, meaning that the cipher consistently follows the same sequence of steps in each execution, with the specific steps differing between even and odd rounds.

CAST-256 is another Feistel network that segments the input into multiple blocks [1]. The cipher runs 48 rounds, which are executed into three groups of 16 rounds each. For every group, the cipher performs different operations. As in the case of Skipjack[15], although changes are performed in the steps taken at different rounds, overall, its structure is static, i.e., the algorithm runs the same steps regardless of the key or the input.

Blowfish is a Feistel network featuring a variable key length, the closest idea to our approach [14]. The cipher initializes the S-boxes in a key-dependent manner, causing the S-boxes to change with each execution based on the key. This approach is more dynamic than those used in CAST-256 [1] and Skipjack [15], although it remains static according to our informal definition. While the internal state of the cipher is altered using different tables generated by the key, the operations performed on this data remain consistent. For instance, consider the $\mathsf{F-Function}$ used in each round. Although the four S-boxes change according to the key, the round's output is generated solely through XOR operations. Consequently, the algorithm can be implemented without any control structures once the S-boxes are instantiated.

In contrast to previous methods, our framework generates ciphers that dynamically produce the pseudorandom function (PRF) used in each Feistel round based on a secret key. It is important to note that the PRFs for each round differ from an algorithmic perspective; i.e., different keys will result in different encryption algorithms. This approach aims to ensure the secrecy of both the encryption key and the encryption algorithm. Consequently, an attacker must first identify the encryption algorithm (i.e., determine which PRF is used in each round) before attempting an attack. The remainder of the paper is structured as follows: Section 3 outlines the general framework of Singularization. Section 4 presents a case study on a reduced 6-round DES, demonstrating how redesigning the cipher within our framework renders differential cryptanalysis as complex as a direct brute-force attack on the key. Finally, Section 5 provides conclusions and suggests directions for future research.

3 Cipher structure

Notations.

1. $S_0||S_1$ denotes the concatenation between the binary strings S_0 and S_1.
2. $S_0 \oplus S_1$ denotes the result of the bitwise XOR operation between the binary strings S_0 and S_1 of equal lengths.
3. $\textsc{Split}\,(S)$ denotes the procedure that receives as input a binary sequence S of $2n$ bits and returns two binary strings containing the most and least significant n bits of S:

$$S_L, S_R \leftarrow \textsc{Split}\,(S)$$

where $S_L||S_R = S$.

General structure. A cipher instantiated by Singularization is based on the structure of a Feistel network [11]. Following the MTD philosophy, we sought to increase the complexity of a possible attack by using, for each round, a pseudorandom function chosen uniformly at random from a set of distinct pseudorandom functions. Let $F_c : \{0,1\}^{k_c} \times \{0,1\}^n \to \{0,1\}^n$, $c \in \mathbb{N}$, be a PRF with a

key of length k_c bits and a block of length n bits. We consider the following set of N_s PRFs:

$$\mathbf{F} = \{F_1, F_2, \ldots, F_{N_s}\}$$

We denote by $\mathbf{F}_i$, the i^{th} PRF in the set, i.e., F_i. Note that all PRFs in the set $\mathbf{F}$ are different. In theory, two PRFs can be considered distinct if they use different keys. However, our design goes beyond merely changing the key from one round to another; it involves altering the underlying algorithms that implements the PRFs. In this sense, we define two PRFs as distinct as follows:

Definition 1. *Let $F_x : \{0,1\}^{k_x} \times \{0,1\}^n \to \{0,1\}^n$ and $F_y : \{0,1\}^{k_y} \times \{0,1\}^n \to \{0,1\}^n$ be two pseudorandom functions with key lengths of k_x and k_y bits respectively. Both functions act over binary strings of length n. We called F_x and F_y ϵ-different, denoted by $F_x \neq_\epsilon F_y$, if:*

$$Pr\left[F_x\left(k,m\right) = F_y\left(k,m\right)\right] < \epsilon \; \forall \left(k,m\right) \in \{0,1\}^{k_c} \times \{0,1\}^n$$

The definition aims to capture the intuition that two PRFs are considered distinct if their underlying computations differ. We formally define two computations as different if the probability of returning the same outputs for the same inputs is below a certain threshold, denoted by ϵ. At each round i of the cipher, we select a PRF, F_i, uniformly at random from the set $\mathbf{F}$. The input to the cipher is $2n$ bits in length, divided into two parts: the left part and the right part, each consisting of n bits. The left part represents the most significant bits of the input, while the right part represents the least significant bits. Similarly, the output of the cipher is $2n$ bits long and is divided in the same manner as the input. The total number of rounds is denoted by N_r.

The round operations. We use two types of keys in a single round:

1. The *round key*, RK_i, represents the key used by the PRF chosen in round i, $1 \leq i \leq N_r$.
2. The *selection key*, SK_i, represents the index of the selected PRF from the set $\mathbf{F}$ in round i, $\mathrm{SK}_i \in \{1, 2, \ldots, N_s\}$.

We call the state any intermediate result of the cipher. The state S_i represents the output of round i. The operations performed in a single round are described by RoundFunction in Algorithm 1. The RoundFunction receives as inputs the current state, S_i, the round key, RK_i and the selection key SK_i based on which the PRF of the round is determined. The inverse of a round function is described by InverseRoundFunction in Algorithm 2.

Encryption and decryption. The encryption and decryption procedures of the cipher, Encrypt and Decrypt, are described in Algorithms 3 and 4 respectively. The encryption/decryption algorithm receives the plaintext/ciphertext as input, the set of the round keys, and the set of the selection keys.

Algorithm 1 Round operations

function RoundFunction(S_i, RK_i, SK_i)
    $S_{L_i}, S_{R_i} \leftarrow \textsc{Split}(S_i)$
    $S_{L_{i+1}} \leftarrow S_{R_i}$
    $S_{R_{i+1}} \leftarrow \mathbf{F}_{SK_i}(\mathrm{RK}_i, S_{R_i}) \oplus S_{L_i}$
    return $S_{L_{i+1}} || S_{R_{i+1}}$
end function

Algorithm 2 Inverse round operations

function InverseRoundFunction(S_i, RK_i, SK_i)
    $S_{L_i}, S_{R_i} \leftarrow \textsc{Split}(S_i)$
    $S_{R_{i-1}} \leftarrow S_{L_i}$
    $S_{L_{i-1}} \leftarrow \mathbf{F}_{SK_i}(\mathrm{RK}_i, S_{L_i}) \oplus S_{R_i}$
    return $S_{L_{i-1}} || S_{R_{i-1}}$
end function

Algorithm 3 Encryption

function Encrypt(P, $\{RK_1, RK_2, \ldots, RK_{N_r}\}$, $\{SK_1, SK_2, \ldots, SK_{N_r}\}$, N_r)
    $S_1 \leftarrow \textsc{RoundFunction}(P, \mathrm{RK}_1, \mathrm{SK}_1)$
    for i = 2 **to** $\mathrm{N_r}$ **do**
        $S_i \leftarrow \textsc{RoundFunction}(S_{i-1}, \mathrm{RK}_i, \mathrm{SK}_i)$
        $i \leftarrow i + 1$
    end for
    return S_{N_r}
end function

Algorithm 4 Decryption

function Decrypt(C, $\{RK_1, RK_2, \ldots, RK_{N_r}\}$, $\{SK_1, SK_2, \ldots, SK_{N_r}\}$, N_r)
    $S_{N_r} \leftarrow C$
    for i = $\mathrm{N_r} - 1$ **to** 0 **do**
        $S_i \leftarrow \textsc{InverseRoundFunction}(S_{i+1}, \mathrm{RK}_{i+1}, \mathrm{SK}_{i+1})$
        $i \leftarrow i - 1$
    end for
    return S_0
end function

Design rationale. We designed this cipher following the moving target defense strategy [7]. There are four main ideas of our design:

1. Our cipher is based on a Feistel network to facilitate theoretical analysis and formal proofs based on previous works [11].
2. At every round, we chose the PRF uniformly at random from a set of PRFs following the idea of changing between multiple cryptosystems proposed in [6]. This will change the cryptographic-related code that is called each round.
3. We set the number of rounds, N_r to be at least four to facilitate the theoretical analysis of our cipher in the context of Luby-Rackoff theorem [12].

A generic cipher designed with Singularization is illustrated in Figure 1.

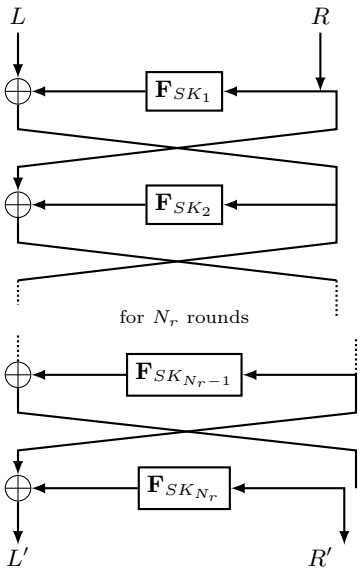

Fig. 1. The cipher structure

4 Case study

In this section, we present a case study on 6-round DES [13]. We illustrate how this variant of DES is vulnerable to differential cryptanalysis and how the attack is mitigated by redesigning the cipher in the proposed framework [10, 5]. We denote by P^i the difference between two inputs of round i, defined as the bitwise XOR of these inputs. Correspondingly, C^i represents the difference between the outputs of round i. Additionally, P_L^i and P_R^i denote the most significant and least significant bits of P^i, respectively, while C_L^i and C_R^i denote the most significant and least significant bits of C^i, respectively. The notation $(a_0, a_1, a_2, a_3)_{\mathrm{x}}$ represents an array of 4 bytes in hexadecimal format.

4.1 Differential cryptanalysis

Differential cryptanalysis aims to exploit instances where a specific difference between two plaintexts likely results in a particular difference between the corresponding ciphertexts. In an ideally secure cipher, given a plaintext difference $P = P' \oplus P''$, the probability of obtaining the corresponding ciphertext difference $C = C' \oplus C''$ is $\frac{1}{2^n}$, where n represents the block size. Since some of the S-boxes do not produce an equally distributed output, the difference between two inputs of the S-box will not result in an equally distributed difference in the corresponding outputs. A difference distribution table captures how various input differences generate specific output differences. The element on row P and column C counts how many input pairs (P', P'') with $P' \oplus P'' = P$ produces a pair of outputs (C', C'') with $C' \oplus C'' = C$. If an S-box does not have equally distributed outputs, then we say it is vulnerable to differential cryptanalysis. The propagation of a specific input difference through the rounds of the cipher to a particular output difference is referred to as differential characteristics. Assuming there are vulnerable S-boxes in each round, an input difference of P^i in the i^{th} round will, with high probability, produce a specific difference C^i between the round outputs. The sequence $P^1 \rightarrow P^2 \rightarrow \cdots \rightarrow P^{N_r}$ represents the differential characteristic, where N_r is the number of rounds in the cipher. Knowing P^{N_r}, the input difference of the last round enables the attacker to recover part of the last round key. The target partial subkey refers to all bits of the last round key influenced by vulnerable S-boxes. If the attacker has access to a chosen plaintext oracle, the attack proceeds as follows:

1. Determine the differential characteristics of the cipher: $P^1 \rightarrow P^2 \rightarrow \ldots, \rightarrow P^{N_r}$.
2. Generate a pair of plaintexts, (P', P'') at difference P^1, i.e., $P' \oplus P'' = P^1$.
3. Encrypt the plaintexts into the ciphertexts (C', C'').
4. For each ciphertext form the pair (C', C''), determine the inputs of the last round for each possible target partial subkey.
5. If the difference between the inputs of the last round is equal to the difference expected from the differential characteristics, i.e., P^{N_r}, increment a counter for the current target partial subkey.
6. Determine the correct partial subkey as the one with the highest counter and brute-force the rest of the remaining bits of the last round key.

Table 1. S_1 difference distribution table

In \ Out	0x0	0x4	0x5	0x8
0x03	14	10	6	6
0x0E	0	6	6	6
0x24	12	2	2	14
0x30	0	12	6	8

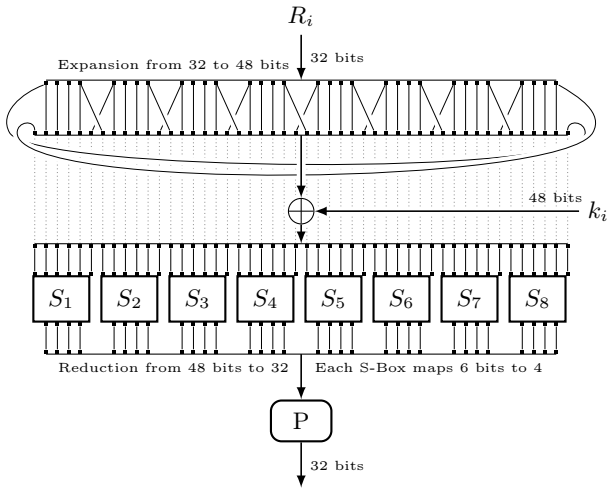

Fig. 2. Round function F

4.2 DES structure and differential cryptanalysis

The particular structure of 6-round DES is depicted in Fig. 1. DES uses a 56-bit key with a block size of 64.

The round function, F, is a substitution-permutation network with the following structure:

1. The PRF receives as input a block of 32 bits and a round key of 48 bits.
2. Applies an expansion function on the block input and computes the bitwise XOR between the result and the round key. The expansion function extends the 32 input bits to 64.
3. Runs the result through a set of 8 S-boxes and applies the round permutation.

Each S-box has a 6-bit input and a 4-bit output; therefore, the input and output of the PRF are both 32 bits. The PRF for a round is illustrated in Fig. 2. Given that an S-box has a 6-bit input and a 4-bit output, the difference distribution table will contain 64 rows and 16 columns. Table 1 presents a subset of the difference distribution table for the S_1 substitution. If S_1 produced equally distributed outputs, each element of the table would be equal to 4. However, this is not the case for the S_1 substitution. For example, given an input difference of 0x03, instead of having a probability of $\frac{4}{64}$ for the output 0x0, we observe a probability of $\frac{14}{64}$.

For DES, two differential characteristics for the first three rounds are described in Tables 2 and 3. Consider, for example, the first differential characteristic. In this scenario, the input difference to the fourth round PRF is $(40\ 08\ 00\ 00)_{\mathrm{x}}$. After expansion, five S-boxes (S_2, S_5, S_6, S_7, S_8) receive zero difference inputs and thus return zero difference outputs. The attacker is interested in determining the output difference of the last round PRF (since this

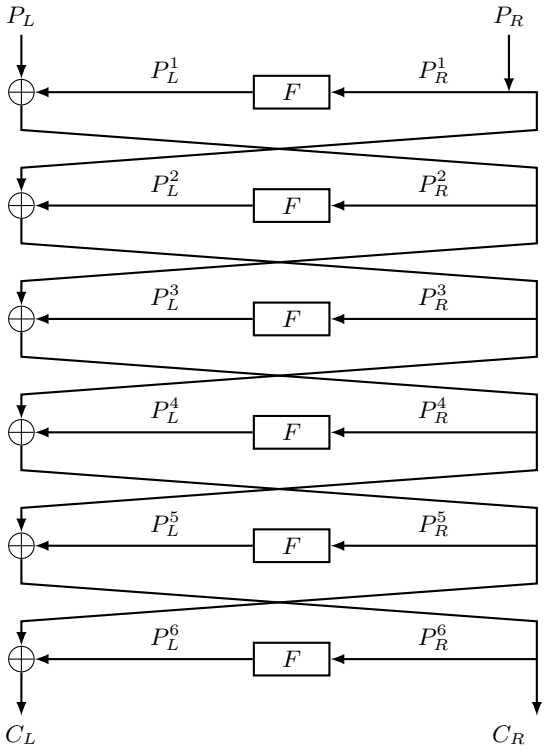

Fig. 3. 6-round DES

PRF applies the last round key), i.e., P_L^6. Since the attacker knows the output difference of the cipher, C_L, they can compute P_L^6 as

$$P_L^6 = C_L \oplus P_R^5 \tag{1}$$

On the other hand, P_R^5 is computed from the output difference of the fourth round PRF, P_L^4, and the input difference of the third round PRF, P_R^3, which is known from the differential characteristics:

$$P_R^5 = P_L^4 \oplus P_R^3 \tag{2}$$

Table 2. First differential characteristics

i	P_L^i	P_R^i	$Probability$
1	$(40\ 08\ 00\ 00)_x$	$(04\ 08\ 00\ 00)_x$	1
2	$(04\ 00\ 00\ 00)_x$	$(40\ 08\ 00\ 00)_x$	1/4
3	$(00\ 00\ 00\ 00)_x$	$(00\ 00\ 00\ 00)_x$	1
4	$(04\ 00\ 00\ 00)_x$	$(40\ 08\ 00\ 00)_x$	1/4

Table 3. Second differential characteristics

i	P_L^i	P_R^i	$Probability$
1	$(00\ 20\ 00\ 08)_x$	$(00\ 00\ 04\ 00)_x$	1
2	$(00\ 00\ 04\ 00)_x$	$(00\ 20\ 00\ 08)_x$	1/4
3	$(00\ 00\ 00\ 00)_x$	$(00\ 00\ 00\ 00)_x$	1
4	$(00\ 00\ 04\ 00)_x$	$(00\ 20\ 00\ 08)_x$	1/4

Since five S-boxes from the fourth round have zero output difference, 20 out 32 bits of P_L^4 are zero. Tracing back the result, the attacker can compute 20 bits of P_L^6 as:

$$P_L^6 = C_L \oplus P_R^3 \tag{3}$$

The remaining 12 bits can be easily brute-forced. At this point, the attacker knows the input of the last round PRF, $P_R^6 = C_R$, and, due to the vulnerable S-boxes, also knows the output of this PRF, P_L^6. Consequently, the attacker can brute-force the corresponding bits from the target partial subkey for each vulnerable S-box. The main idea is that, unlike a brute-force attack on the 30 bits of the target partial subkey, the attacker determines the corresponding bits from the round key independently for each of the five vulnerable S-boxes. This reduces the number of trials from 2^{30} to 5×2^6.

The attacker proceeds similarly for the second differential characteristic, with the vulnerable S-boxes S_1, S_2, S_4, S_5, and S_6. Since three S-boxes are common between the two differential characteristics (S_2, S_5, S_6), the attacker can determine 42 bits of the encryption key using differential analysis and brute-force only 14 bits.

4.3 DES redesign

Redesigning the cipher using our framework mitigates the differential cryptanalysis attack by making its complexity comparable to that of a direct brute-force attack on the encryption key. One root cause of the attack is the static nature of the PRF used in each round.

Using our framework, we design the unique PRF at each round with the following specifications:

1. The PRF receives as input a block of 32 bits and a master round key of 48 bits.
2. In addition to the round key of 48 bits, a selection key of 8 bits is also generated.
3. Applies the expansion function on the 32-bit input block to get a 48-bit result.
4. Splits the result of the expansion function and the round key into 6-bit words.
5. Combines the w^{th} word from the expansion result with the w^{th} word from the round key in the following key:

(a) If the w^{th} bit from the selection key is 0, the combination result is the bitwise XOR between the word from the expansion result and the word from the round key.
(b) If the w^{th} bit from the selection key is 1, the combination result is the addition modulo 64 between the word from the expansion result and the word from the round key.

6. Runs the result through the permuted set of S-boxes and applies the round permutation.

Within the context of our framework, the set $\mathbf{F}$ comprises 256 PRFs. The PRF $\mathbf{F}_{SK}$ partitions the input and the round key into arrays of words and combines them by performing either bitwise XOR or addition modulo 64 at the word level. The selection key, SK, determines the specific operations to be applied.

Using the redesigned round function with a PRF chosen uniformly at random for each round, the attacker must determine how the result of the expansion is combined with the key at each round. The differential cryptanalysis attack is successful in the initial DES construction because the mixing between the round key and the input preserves the input difference:

$$(P' \oplus K) \oplus (P'' \oplus K) = P' \oplus P'' \tag{4}$$

From another perspective, for the differential cryptanalysis attack to be successful, the attacker must know the input difference for each S-box. The input for an S-box is the result of mixing the round input, which is known to the attacker, with the round key, which is unknown to the attacker. If the mixing is performed using bitwise XOR, then according to (4), the input differences for the S-boxes are the same as the input difference before mixing. This means that even if the attacker does not know the round key, they can still compute the input difference for the S-boxes. After the redesign, although (4) remains valid if bitwise XOR is replaced by modular addition, the attacker must determine for each word whether the input difference is the result of bitwise XOR or modular addition. In the original DES construction, the attacker must perform 5×2^6 trials to determine the bits of the target partial subkey. In the new variant constructed on our framework, the attacker must perform each of the 5×2^6 trials for each possible mixing operation at each round. Since there are 2^8 possible mixing operations per round (with 8 words to be mixed and two possible operations for each pair of words) and the cipher has six rounds, there are 2^{48} possible mixing operations in total. Consequently, the total number of trials becomes $5 \times 2^6 \times 2^{48} = 5 \times 2^{54}$. This number is comparable to the number of trials required for a brute-force attack on the entire DES key, which is 2^{56}.

5 Conclusions and further directions of research

In this paper, we proposed Singularization, a novel framework for constructing block ciphers inspired by the principles of MTD. Our construction utilizes a

Feistel network where the PRF for each round is randomly selected from a set of PRFs. We examined differential cryptanalysis on a reduced 6-round DES. We demonstrated that redesigning the cipher within this framework reduces the attack's effectiveness to that of an almost direct brute-force attack on the encryption key.

An important next step in advancing the Singularization framework is to generalize the formal protection guarantees by investigating various types of cryptanalysis attacks without limiting the analysis to a specific cipher.

Future research directions include exploring the application of this framework to redesign other encryption algorithms. This could result in ciphers with a reduced number of rounds, thereby enhancing the efficiency of encryption algorithms in terms of running time and energy consumption, which is particularly important for resource-constrained devices.

Another avenue for research is to conduct a comprehensive security analysis of the framework using other types of attacks, such as linear cryptanalysis or side-channel attacks.

References

1. Adams, C., Gilchrist, J.: The cast-256 encryption algorithm. RFC 2612 pp. 1–19 (1999), https://dblp.org/rec/journals/rfc/rfc2612
2. Alluhaidan, A.S.D., Prabu, P.: End-to-end encryption in resource-constrained iot device. IEEE Access **11**, 70040–70051 (2023)
3. Bhagat, V., Kumar, S., Gupta, S.K., Chaube, M.K.: Lightweight cryptographic algorithms based on different model architectures: A systematic review and futuristic applications. Concurrency and Computation: Practice and Experience **35**(1), e7425 (2023)
4. Biham, E.: Differential Cryptanalysis, pp. 332–336. Springer US, Boston, MA (2011)
5. Biham, E., Shamir, A.: Differential cryptanalysis of des-like cryptosystems. Journal of CRYPTOLOGY **4**, 3–72 (1991)
6. Casola, V., De Benedictis, A., Albanese, M.: A moving target defense approach for protecting resource-constrained distributed devices. In: 2013 IEEE 14th International Conference on Information Reuse & Integration (IRI). pp. 22–29. IEEE (2013)
7. Cho, J.H., Sharma, D.P., Alavizadeh, H., Yoon, S., Ben-Asher, N., Moore, T.J., Kim, D.S., Lim, H., Nelson, F.F.: Toward proactive, adaptive defense: A survey on moving target defense. IEEE Communications Surveys & Tutorials **22**(1), 709–745 (2020)
8. Daemen, J., Rijmen, V.: Aes proposal: Rijndael (1999)
9. Gaber, C., Macariot-Rat, G., David, S., Wary, J.P., Cuaboz, A.: Position paper: Strengthening applets on legacy sim cards with singularization, a new moving target defense strategy. In: International Conference on Mobile, Secure, and Programmable Networking. pp. 71–74. Springer (2023)
10. Heys, H.M.: A tutorial on linear and differential cryptanalysis. Cryptologia **26**(3), 189–221 (2002)
11. Hoang, V.T., Rogaway, P.: On generalized feistel networks. In: Annual Cryptology Conference. pp. 613–630. Springer (2010)
12. Luby, M., Rackoff, C.: How to construct pseudorandom permutations from pseudorandom functions. SIAM Journal on Computing **17**(2), 373–386 (1988)
13. Pub, F.: Data encryption standard (des). FIPS PUB pp. 46–3 (1999)
14. Schneier, B.: Description of a new variable-length key, 64-bit block cipher (blowfish). In: International Workshop on Fast Software Encryption. pp. 191–204. Springer (1993)
15. of Standards, N.I., (NIST), T.: Skipjack and kea algorithm specifications. Tech. rep. (May 1998), https://csrc.nist.gov/Presentations/1998/Skipjack-and-KEA-Algorithm-Specifications