The ISC Int'l Journal of Information Security

ISeCure

October 2025, Volume 17, Number 3 (pp. 1–12)

http://www.isecure-journal.org

Presented at the ISCISC'2025 in Tehran, Iran.

Cryptanalysis of Reduced-Round GFRX-64 **

Javad Alizadeh [1,*], Bahman Madadi [2]

[1] *Fath Center, Faculty and Research Center of Computer, Imam Hossein University, Tehran, Iran.*

[2] *Faculty and Research Center of Computer, Imam Hossein University, Tehran, Iran.*

ARTICLE INFO.

Keywords:
Lightweight block cipher, GFRX, Neural distinguisher, SAT/SMT-based cryptanalysis, Differential cryptanalysis, Linear cryptanalysis.

Type:

doi:

ABSTRACT

In 2023, Zhang *et al.* introduced the lightweight block cipher family GFRX-b/k, offering various versions with different block (b) and key (k) lengths. Due to the similarity of the GFRX's round function to that of the SIMON, the designers referenced the cryptanalysis conducted on the SIMON-32 and claimed that the GFRX-64/128, with higher than 19 and 13 rounds, is resistant to differential and linear cryptanalysis, respectively. In this paper, we examine the differential and linear cryptanalysis of GFRX-64/96 and GFRX-64/128. We first introduce baseline neural distinguishers for up to 7 rounds of the GFRX-64/96. Subsequently, we extend a 6-round neural distinguisher by adding 2 rounds to perform a key recovery attack, achieving an 8-round key rank analysis through a deep learning-based approach. Furthermore, we conduct an automated cryptanalysis of GFRX-64 using a SAT/SMT-based framework, identifying an 11-round differential distinguisher with a probability of 2^{-62}, a 15-round linear distinguisher with a correlation of 2^{-30}, and a 17-round linear hull with a correlation of $2^{-31.61}$. These results indicate that reducing the differential and linear cryptanalysis of the GFRX block cipher to the differential and linear cryptanalysis of the SIMON block cipher cannot yield accurate results or bounds. To the best of our knowledge, this work represents the first third-party cryptanalysis of the GFRX block cipher, offering new insights into its security.

1 Introduction

Cryptography is a tool used to provide some main components of information security, such as confidentiality and integrity. Block ciphers are a fundamental component of modern cryptography and have a critical role in securing data. One can see the application of the block cipher in industrial systems, computer networks, or Internet of Thing (IoT)-based applications. Some systems and applications, such as IoT sensors and embedded systems, have been developed in recent years, have constraints in their resources and memory usage. This leads to the development of design and cryptanalysis of lightweight block ciphers.

In recent years, several lightweight block ciphers,

* Corresponding author.

**The ISCISC'2025 program committee effort is highly acknowledged for reviewing this paper.

Email addresses: JAlizadeh@ihu.ac.ir, BahmanMadadi@ihu.ac.ir

ISSN: 2008-2045

such as SIMON and SPECK [1], SAND-2 [2], ARADI [3], GFSPX [4], QLW [5], LiteEncrypt [6], and so on, have been introduced. From the perspective of the components and transformations used in the design of these block ciphers, they can be divided into two categories: block ciphers with substitution boxes (S-boxes) and block ciphers without substitution boxes. The second category of block ciphers employs basic operations such as modular addition, bitwise AND, rotation, and XOR. These block ciphers are referred to as ARX (Addition-Rotation-XOR) or AND-RX (AND-Rotation-XOR) block ciphers. In addition to the components used for designing a block cipher, block ciphers can also be categorized as Feistel (generalized Feistel) and non-Feistel.

While the block ciphers should be efficient, they should also provide security against known attacks. Differential [7] and linear [8] cryptanalysis are two important attacks that have been widely discussed and applied to various block ciphers. Advancements in these attacks have led to other attacks such as impossible differential [9], zero correlation [10], integral [11], boomerang [12], differential-linear [13], and so on. As a few examples in the field of cryptanalysis of lightweight block ciphers, one can consider the boomerang attack on Midori [14], integral cryptanalysis of SAND [15], differential fault analysis of SKINNY [16] and CRAFT [17], differential cryptanalysis of SAND-2 [18], and linear cryptanalysis of SIMON [19].

Before 2012 and the paper by Mouha *et al.* [20], cryptanalysis of block ciphers was done manually, usually based on an ad-hoc approach. In 2012, Mouha et al. explained how differential and linear cryptanalysis can be performed using Mixed-Integer Linear Programming (MILP). This work leads to the automation of the cryptanalysis of block ciphers. In addition to developments in MILP-based automatic cryptanalysis methods, cryptanalysts have also developed other automatic methods, including constraint programming (CP)-based automatic cryptanalysis methods [21] and also methods based on the boolean satisfiability problem (SAT) or satisfiability modulo theories (SMT) [22].

Due to the importance of automatic cryptanalysis methods in increasing the speed and accuracy of cryptanalysis of block ciphers, these methods have always been of interest. With the advancements in machine learning applications, it has also been utilized in the differential analysis of block ciphers to enhance the efficiency of analyzing the security of block ciphers. The application of deep learning to find the neural distinguisher of a block cipher was first introduced by Gohr [23] in CRYPTO 2019, about the block cipher SPECK and then was used for differential cryptanalysis of some block ciphers such as [24–26].

In 2023, Zhang *et al.* [27] introduced a new lightweight block cipher named GFRX. GFRX uses Addition/AND, Rotation, and XOR (ARX or AND-RX) operations in a generalized Feistel structure. GFRX-b/k has different versions with the block length (b) of 64 bits to 128 bits and the key length (k) of 96 bits to 256 bits. The GFRX variants differ in the number of rounds they employ. For example GFRX-64/96 uses 26 rounds, while GFRX-64/128 uses 27 rounds.

So far, no independent cryptanalysis of this block cipher exists. Only the designers of GFRX analyzed the security of their block cipher against differential and linear cryptanalysis and asserted that the versions with more than 19 rounds and 13 rounds of the block cipher GFRX-64/128 are resistant to differential and linear cryptanalysis, respectively. Their cryptanalysis method was based on the premise that since one-half of the GFRX-64/128 block cipher is similar to the SIMON-32/64 block cipher, the results of the differential and linear cryptanalysis of the SIMON-32/64 block cipher can be considered as a bound for the differential and linear cryptanalysis of the GFRX-64/128 block cipher, respectively. This paper examines the security of GFRX against differential and linear attacks.

Contribution

In this paper, we cryptanalysis the block ciphers GFRX-64 against differential and linear attacks for the first time. In the beginning, we try to use deep learning for the differential cryptanalysis of the block cipher GFRX-64/96. Then, a SAT/SMT-based automated method is used to verify the security of the round-reduced versions of GFRX-64 against differential and linear cryptanalysis. The achievements of this paper are summarized as follows:

(1) A deep learning approach produces neural differential distinguishers for 1 to 7 rounds of GFRX-64/96. Using the 6-round neural distinguisher, an 8-round key ranking is achieved. These neural distinguishers are derived using a residual neural network [28], with their accuracies at various rounds shown in Table 1.

Table 1. Neural differential distinguishers for the GFRX-64/96 cipher

# Rounds	1	2	3	4	5	6	7	8
Accuracy	1.0000	1.0000	1.0000	1.0000	0.9997	0.8751	0.6264	0.5009

(2) The differential and linear behavior of the GFRX-64 is modeled as a SAT/SMT problem,

which is solved using the CryptoSMT tool. This results in an 11-round differential characteristic with a probability of 2^{-62} and a 15-round linear distinguisher with a correlation of 2^{-30}. While the differential cryptanalysis does not exceed the number of rounds covered by the designer's claims, the linear cryptanalysis represents an improvement of one round compared to the designers' claims. Additionally, we obtained a 17-round linear hull for the GFRX-64 block cipher with a correlation of $2^{-31.61}$. Table 2 compares these cryptanalysis results with prior GFRX-64 cryptanalysis.

Table 2. Comparison of differential and linear cryptanalysis results of the GFRX-64 block cipher

Cryptanalysis	#rounds	Prob./Corr.	Ref.
Diff.	11		[27]
Diff.	11	2^{-62}	This paper
Lin	13		[27]
Lin	15	2^{-30}	This paper
Lin. Hull.	17	$2^{-31.61}$	This paper

To facilitate further investigation, verification, and reproducibility, we have made our source code for both neural-based and SAT/SMT-based cryptanalysis publicly available at: https://github.com/CryptanalysisGFRX/GFRX64.git.

Organization

This paper is structured as follows: Section 2 covers the preliminaries. Section 3 discusses neural distinguishers for GFRX-64/96 and employs a 6-round distinguisher in a 8-round key recovery attack on the cipher. Automated differential and linear cryptanalysis of GFRX-64 are explained in Section 4 and Section 5, respectively. Finally, the conclusion of the paper is presented in Section 6.

2 Preliminaries

2.1 Specification of GFRX-64

The block cipher GFRX-64 are two variants of the GFRX block cipher family with the block length of 64 bits and key length of 96 and 128 bits. GFRX is a lightweight family of block cipher designed based on a generalized Feistel structure with four branches. Suppose that X is a $4n$-bit string. According to the proposal of GFRX, we consider $X = L_0 \parallel L_1 \parallel R_0 \parallel R_1$ where L_0, L_1, R_0, and R_1 are the words of n-bit length. We suppose that the subkey used in round r of GFRX is K^r and the output of this round is $X^r = L_0^r \parallel L_1^r \parallel R_0^r \parallel R_1^r$. In order to update the state X^r to the new state X^{r+1}, GFRX uses three round functions named F_{AN}, F_{ADL}, and F_{ADR} that are defined as:

$$F_{AN}(t) = (t <<< a) \& (t <<< b) \oplus (t <<< c),$$

$$F_{ADL}(s,t) = (s >>> d) \boxplus t,$$

$$F_{ADR}(s,t) = (s <<< e) \oplus t,$$

where $\boxplus$, $>>>$, $<<<$, $\&$, and $\oplus$ means addition module 2^n, rotation to right, rotation to left, bitwise AND, and bitwise XOR, respectively. Also a, b, c, d, e are named the rotation parameters and are $1, 8, 2, 8, 3$, respectively.

In the round i of GFRX, three round functions, F_{AN}, F_{ADL}, and F_{ADR}, are applied to the state $X^i = L_0^i \parallel L_1^i \parallel R_0^i \parallel R_1^i$ and the updated state $X^{i+1} = L_0^{i+1} \parallel L_1^{i+1} \parallel R_0^{i+1} \parallel R_1^{i+1}$ is produced as follows:

$$L_0^{i+1} = F_{ADL}(L_1^i, R_0^i) \oplus K_1^i, L_1^{i+1} = F_{AN}(R_0^i) \oplus R_1^i \oplus K_2^i,$$

$$R_0^{i+1} = F_{AN}(L_1^i) \oplus L_0^i \oplus K_0^i, R_1^{i+1} = F_{ADR}(L_0^{i+1}, R_0^i),$$

where K_0^i, K_1^i, and K_2^i are the subkeys. The structure of GFRX is depicted in Figure Figure 1.

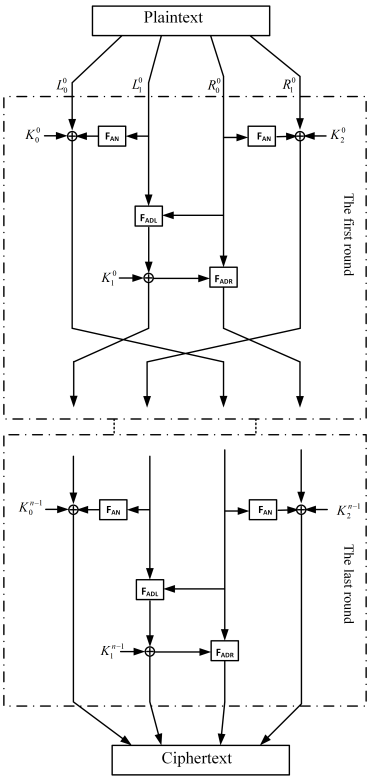

Figure 1. Structure of the GFRX [27]

2.2 Specification of (SAT/SMT-based) differential and linear cryptanalysis

Differential cryptanalysis [7] examines a specific plaintext difference and its propagation through the round transformations of a block cipher. It identifies the input and output differences with the highest occur-

rence probability for each transformation or round. From these, a differential characteristic for a given number of rounds is established. If this characteristic's probability exceeds that of the best differential characteristic of a random function, the differential attack is considered successful, allowing the characteristic to differentiate the block cipher from a random function.

In linear cryptanalysis [8], attackers focus on the linear relationships between plaintext bits, ciphertext bits, and key bits. To obtain these relationships, it is necessary to find a linear approximation of the behavior of the transformations or rounds of the block cipher. If the correlation of these linear approximations is higher than the correlation of them for a random function, then the linear attack on the block cipher is successful.

A Boolean formula is a formula that includes Boolean variables, logical operators AND, OR, NOT, and parentheses [22]. The Conjunctive Normal Form (CNF) of a Boolean formula is defined as $\bigwedge_{i=0}^{n} \bigvee_{j=0}^{m_i} C_{ij}$, where each C_{ij} $(0 \leq i \leq n, 0 \leq j \leq m_i$ [22]. The Boolean Satisfiability Problem, abbreviated as SAT, is a decision problem in computer science. The goal of this problem is to determine whether it is possible to assign values of TRUE and FALSE to the variables of a Boolean function such that the final value of this Boolean function equals TRUE. If such an assignment is found, then the Boolean function is said to be satisfiable. Otherwise, it is said to be unsatisfiable. The Satisfiability Modulo Theories (SMT) problem can be viewed as an extension of the SAT problem, where Boolean variables are replaced by more complex structures such as real numbers, arrays, and bit vectors within a formula. The challenge lies in determining whether this formula is satisfiable under specific logical theories or not. Computational complexity theory categorizes SAT problems as NP-complete, indicating that they are at least as difficult as any problem within the NP class. As a result, no efficient algorithm for solving SAT problems has been proposed thus far.

In the field of cryptanalysis of symmetric ciphers, especially block ciphers, SAT/SMT are used to automate the search for distinguishers (e.g. differential or linear distinguishers). For instance, Mouha and Preneel [29] employed a SAT/SMT-based approach in 2013 to automatically identify differential characteristics of ARX ciphers. Automated cryptanalysis of block ciphers using SAT/SMT consists of two main steps: modeling the block cipher's behavior against an attack as a SAT/SMT problem and solving the SAT/SMT problem. A solution to the SAT/SMT problem serves as a distinguisher for the cipher.

To model a block cipher's behavior against attacks such as differential or linear attacks, it is essential to first model the round function's behavior under these attacks. This also requires modeling the operators involved in the round function. Sun *et al.* [22] classified the common operators in block ciphers into linear and nonlinear categories and described how their differential and linear behaviors can be framed as SAT/SMT problems. This process involves converting the operations of these operators into logical formulas such as conjunctive normal form (CNF) which can then be solved with SAT/SMT solvers to evaluate their performance against differential or linear attacks. Branch, modular addition, bitwise AND, bitwise rotation, and XOR are the four main operators used in the design of the GFRX block cipher. The CNF representations of differential and linear behavior of these operators are provided in [22].

After the differential or linear behavior of the block cipher algorithm has been modeled as a SAT/SMT problem, SAT/SMT solvers such as STP [30] and Boolector [31] are used to solve this problem. The CryptoSMT [32] is an automated cryptanalysis tool for symmetric ciphers that solves the SAT/SMT models of a block cipher using one of the mentioned solvers.

2.3 Deep learning and neural networks

Deep learning, a subset of machine learning, employs multi-layered neural networks to extract high-level representations from data. These networks are particularly effective in capturing intricate patterns and structures, making them highly suitable for cryptanalysis tasks, such as distinguishing cipher outputs from random sequences [23, 33, 34]. Neural networks consist of multiple layers of interconnected neurons, where each neuron performs a weighted sum of its inputs followed by an activation function. The architecture typically includes an input layer that processes raw data, hidden layers that extract complex features, and an output layer that produces the final predictions, such as the probability of identifying a cipher output as non-random.

Training a neural network involves using labeled datasets, including a training set for optimizing model parameters and a validation set for monitoring generalization. The learning process is guided by a loss function, such as Mean Squared Error (MSE), which quantifies prediction errors, and an optimizer, such as stochastic gradient descent, which updates weights iteratively. A dynamic learning rate schedule helps balance convergence and exploration. After training, the model is evaluated on a test dataset using metrics such as accuracy and validation loss to measure its

performance and generalization ability.

2.4 Neural distinguishers

Neural distinguishers are an advanced application of deep learning in cryptanalysis, designed to differentiate between cipher outputs and random data. They rely on deep neural networks' ability to detect complex statistical biases that are often imperceptible through conventional analytical methods. A neural distinguisher is a supervised machine learning model trained to classify input data into two categories: samples derived from a cipher (target distribution) and purely random samples. Since encryption introduces subtle statistical dependencies, a well-structured neural network can learn to distinguish these patterns, providing a foundation for cryptanalytic attacks.

The construction of a neural distinguisher begins with data generation, where plaintext pairs with specific differences (e.g., Δ_{in}) are encrypted using a reduced-round version of the cipher to produce ciphertext pairs. These are labeled as positive samples ($Y = 1$), while randomly generated pairs serve as negative samples ($Y = 0$). The neural network is trained on these labeled samples, learning patterns unique to the cipher. Given a ciphertext pair $X = (C_0, C_1)$, where C_0 and C_1 originate from either the cipher with input difference Δ_{in} or from a random process, the network approximates the posterior probability $P(Y = 1 \mid X) = f(X)$. A classification threshold determines whether a given sample is more likely to belong to the cipher or the random distribution [35].

3 Differential-neural cryptanalysis of GFRX-64/96

In this section, we present a comprehensive neural cryptanalysis of the GFRX-64/96 block cipher. First, we construct neural distinguishers that exploit the differential characteristics of the cipher to distinguish ciphertext pairs with specific input differences. These distinguishers serve as a foundation for further cryptanalytic attacks. Next, we extend the neural distinguisher-based framework to perform a key recovery attack, demonstrating the practical utility of neural networks in breaking reduced-round versions of GFRX-64/96. The results highlight the effectiveness of deep learning in uncovering vulnerabilities and advancing our understanding of the cipher's security.

3.1 Neural distinguishers for GFRX-64/96

Neural distinguishers represent a novel and powerful approach in cryptanalysis, leveraging deep learning techniques to uncover statistical biases within cryptographic ciphers. This section presents the development and evaluation of neural distinguishers tailored for the GFRX-64/96 cipher. By utilizing a ResNet-based architecture and a systematically generated dataset, we construct distinguishers capable of identifying subtle patterns introduced by the cipher. The training, architecture, and testing of these distinguishers are discussed in detail, along with their performance across different rounds of the cipher.

Data generation

Generating high-quality data is a fundamental step in training effective neural distinguishers for the GFRX-64/96 block cipher. The data generation pipeline consists of creating labeled samples of ciphertext pairs, carefully designed to maximize the distinguishability between cipher outputs and random data. This section details the methodology for constructing the training and validation datasets.

The dataset comprises ciphertext pairs generated with specific input differences (Δ_{in}) to facilitate differential cryptanalysis. For positive samples ($Y = 1$), the ciphertext pairs are derived from plaintext pairs that differ by the chosen Δ_{in}. Negative samples ($Y = 0$) consist of randomly generated ciphertext pairs. The input difference is selected based on its effectiveness in revealing statistical biases within the cipher's structure. Previous studies have demonstrated that a low Hamming weight is the most influential factor in selecting the input difference for neural distinguishers [23, 36]. Therefore, through exploration and testing of various input differences with a Hamming weight of 1, the difference $(0x0, 0x0, 0x0, 0x0040)$ was found to yield the highest accuracy.

Plaintext pairs are generated as follows:

- Random plaintexts $P_0 = (P_{0L0}, P_{0L1}, P_{0R0}, P_{0R1})$ are created using a secure random number generator.
- Corresponding plaintexts P_1 are derived by XOR-ing P_0 with the chosen input difference Δ_{in} for positive samples. For negative samples, P_1 is replaced with randomly generated values.

Each plaintext pair is encrypted using the GFRX-64/96 cipher for a specified number of rounds (n_r). The encryption process produces ciphertext pairs $C_0 = (C_{0L0}, C_{0L1}, C_{0R0}, C_{0R1})$ and $C_1 = (C_{1L0}, C_{1L1}, C_{1R0}, C_{1R1})$, where each ciphertext is represented by four 16-bit words. Each ciphertext pair provides a total of 128 input bits, which are accompanied by a single binary label (Y) to form a complete dataset sample.

To prepare the data for neural network training, the ciphertext pairs are converted into a binary matrix format. The conversion process transforms the

16-bit words of the ciphertexts into their binary representation, ensuring compatibility with the input layer of the neural network.

The final dataset consists of:

- A feature matrix X, where each row represents the binary representation of a ciphertext pair, resulting in a total of 128 bits per sample.
- A label vector Y, indicating whether the sample is positive ($Y = 1$) or negative ($Y = 0$).

This structured data generation pipeline ensures a balanced and diverse dataset, critical for training robust neural distinguishers. By dividing each 64-bit ciphertext into four 16-bit words, the pipeline provides precise binary representations as input features for the neural network. The use of carefully selected input differences enhances the distinguishers' ability to identify statistical biases in the cipher's structure, paving the way for effective cryptanalysis.

Input difference selection

To identify the most effective input difference (Δ_{in}) for training neural distinguishers, multiple candidates were evaluated on a 6-round GFRX-64/96 cipher using a ResNet with a depth of 1 over 3 epochs. Table 3 summarizes some of the tested input differences and their corresponding validation accuracies. Based on the results, the difference $(0x0, 0x0, 0x0, 0x0040)$ achieved the highest accuracy of 0.8673, demonstrating its superior ability to reveal statistical biases in the cipher.

Table 3. Validation accuracies for different input differences (Δ_{in}) on 6-round GFRX-64/96 using a ResNet with depth 1 over 3 epochs.

Input Difference (Δ_{in})	Validation Accuracy
(0x0, 0x0, 0x0, 0x0040)	0.8673
(0x0, 0x0, 0x0, 0x0800)	0.8632
(0x0400, 0x0, 0x0, 0x0)	0.8200
(0x0, 0x0, 0x0, 0x0100)	0.8132
(0x0, 0x0, 0x0, 0x8000)	0.7723
(0x8000, 0x0, 0x0, 0x0)	0.7359
(0x0, 0x0, 0x0, 0x0001)	0.7347
(0x0, 0x0, 0x0, 0x0080)	0.7282
(0x0001, 0x0, 0x0, 0x0)	0.6201
(0x0, 0x0, 0x0001, 0x0)	0.5002
(0x0, 0x0001, 0x0, 0x0)	0.5004

From the results in Table 3, it is evident that the most effective input differences have their active bit in either the first word (P_{0L0}) or the last word (P_{0R1}). This insight highlights the importance of the placement of active bits in designing effective input differences for neural distinguishers.

Neural network architecture

The neural network architecture used for constructing distinguishers is based on a residual neural network (ResNet), following the principles outlined in [23]. ResNet incorporates shortcut connections, enabling the model to bypass certain layers and thereby improve training efficiency.The shortcut connection allows the network to learn modifications to the identity mapping, significantly improving gradient propagation and facilitating the training of deeper models [28]. The input representation consists of ciphertext pairs, where each ciphertext is divided into four 16-bit words, forming a 128-bit feature vector. Given a pair (C_0, C_1), the structured input is represented as $\mathbf{C} = \{C_{0L0}, C_{0L1}, C_{0R0}, C_{0R1}, C_{1L0}, C_{1L1}, C_{1R0}, C_{1R1}\}$. The preprocessing step reshapes this 128-dimensional input into a $(4 \times 4, 16)$ tensor, facilitating structured processing by convolutional layers. A permutation layer is then applied to reorder the tensor, aligning bits of the same position across different words to enhance bit-level feature extraction.

The network begins with an initial convolutional layer that extracts low-level features and generates 32 feature maps. This is followed by a sequence of 10 residual blocks, each containing two convolutional layers with kernel size 3, batch normalization, and ReLU activation. A shortcut connection within each block allows the model to retain identity mappings, improving gradient flow and learning efficiency. These residual blocks enable hierarchical feature learning, capturing intricate dependencies in ciphertext structures. The extracted feature maps are subsequently flattened and passed through two fully connected layers with 64 neurons each, refining the learned representations. The final output layer, equipped with a sigmoid activation function, produces a probability score indicating whether the input originates from the cipher ($Y = 1$) or is random ($Y = 0$).

Training and testing of neural distinguishers

The training dataset consists of 10^7 samples of ciphertext pairs, generated using the methodology described earlier. Each sample is labeled as either positive ($Y = 1$) or negative ($Y = 0$), depending on whether it corresponds to a specific input difference or a random pair. The dataset is balanced to prevent bias during training. To monitor and validate the performance of the network during training, a separate validation dataset of size 10^6 samples was generated. This dataset was created using the same process as the training dataset but was kept entirely separate to avoid data leakage.

The neural network was trained over 100 epochs with a batch size of 5000, using the Adam optimizer

due to its efficiency in handling large datasets and non-convex optimization problems. The loss function employed was Mean Squared Error (MSE), chosen for its capability to measure the difference between predicted and actual outputs effectively. To enhance convergence and prevent the model from getting stuck in local minima, a cyclic learning rate schedule was applied, oscillating between 0.002 and 0.0001 over 10 epochs.

The training process was conducted on Google Colab. After training, the neural distinguisher was evaluated on a fresh test dataset of 10^6 samples. Key metrics such as test accuracy and loss were recorded to assess the model's ability to distinguish between ciphertext pairs derived from the cipher and random data.

Results of neural distinguishers

Neural distinguishers were successfully trained for up to 7 rounds of the GFRX-64/96 cipher, achieving accuracies significantly higher than random guessing (50%). These results demonstrate the effectiveness of the proposed architecture in capturing statistical biases introduced by the cipher. For 8 rounds, the neural distinguisher achieved an accuracy close to 50%, indicating that the distinguishability diminishes as the number of rounds increases.

The detailed results and accuracies for each round are presented in Table 1. It is worth noting that these results serve as a baseline, and further improvements can be achieved through feature engineering on the ciphertexts and by employing more advanced neural network architectures.

3.2 Extension to key recovery attack

Inspired by the approach in [23] for SPECK32/64, we adapt our 6-round GFRX-64/96 neural distinguishers to perform an 8-round key-recovery procedure. The central idea is to reduce the encryption path to the 6-round region recognized by the distinguisher via partial decryption of the final round. Additionally, as the first subkey addition in GFRX-64/96 occurs after the first application of nonlinearity, it is possible to extend the distinguisher by one round at no additional cost. In a chosen-plaintext setting, the adversary can inject plaintext differences of their choosing such that they align perfectly with the output difference of the first round of GFRX-64/96.

Once this partial decryption is carried out using a guessed subkey word, the resulting 6-round internal state is fed into the trained network to produce a likelihood score. By iterating over all possible values of the guessed 16-bit subkey word, one obtains a key rank that indicates how likely each guess is to be correct.

Recall that in an n-round GFRX-64/96 encryption, each round uses three 16-bit subkeys (k_0^i, k_1^i, k_2^i). In our 8-round key recovery, we focus on one 16-bit word of the last-round subkey, for instance k_0^7. Denote the final ciphertext pair by

$$C_0 = (C_{0L0}, C_{0L1}, C_{0R0}, C_{0R1}),$$
$$C_1 = (C_{1L0}, C_{1L1}, C_{1R0}, C_{1R1}),$$

obtained after 8 rounds of GFRX-64/96. To test a candidate value κ for k_0^7, we partially decrypt C_0 and C_1 through the eighth round, leaving the first seven rounds intact. Formally,

$$(\tilde{L}_0, \tilde{L}_1, \tilde{R}_0, \tilde{R}_1) =$$
$$\text{DecOneRound}(C_{0L0}, C_{0L1}, C_{0R0}, C_{0R1};\, \kappa,\, k_1^7,\, k_2^7),$$

and similarly for C_1. Here, DecOneRound inverts exactly one GFRX-64/96 round, using the candidate κ in place of the true key word k_0^7, while k_1^7, k_2^7 (assumed known for this step) come from the expanded key. We thus obtain intermediate ciphertexts (the state after 7 rounds), which we feed into the 6-round neural distinguisher for scoring.

After that, we measure the likelihood of each partially decrypted pair via our 6-round neural distinguisher. Suppose the distinguisher outputs a score $Z_i^k \in [0, 1]$ for pair i, indicating the probability that the pair arises from an actual 6-round GFRX-64/96 encryption (versus random). To combine the scores of n pairs, we apply the following log-odds sum:

$$v_k = \sum_{i=1}^{n} \log_2\Big(\frac{Z_i^k}{1 - Z_i^k}\Big), \tag{1}$$

where k indexes the subkey guess κ. The final score v_k is used to rank all 2^{16} candidate κ. When v_k is largest for the correct subkey $\kappa^\star$, that subkey is ranked highest. Empirically, if the distinguisher is strong and n is sufficiently large (e.g., 64 or 128 ciphertext pairs), the true subkey $\kappa^\star$ will often occupy one of the top ranks.

In GFRX-64/96, the final round key (k_0^7, k_1^7, k_2^7) spans three 16-bit words. The approach above outlines how to recover one of these words (e.g., k_0^7). Repeating the partial-decryption attack with a guessed k_1^7 or k_2^7 allows retrieval of additional subkey words. To realize the attack in practice, one first chooses an input difference Δ_{in} known to yield strong results with the 6-round distinguisher (as established in Section 3). A set of n plaintext pairs (P_0^i, P_1^i) conforming to Δ_{in} is encrypted through 8 rounds to produce ciphertext pairs (C_0^i, C_1^i). Each C_0^i and C_1^i is then partially decrypted with every candidate subkey word κ, generating n intermediate states per guess. These

states are fed into the 6-round neural distinguisher to produce scores Z_i^k, which are aggregated according to (1) to obtain v_k. Sorting v_k in descending order yields the final key rank. Our experiments confirm that, for a moderate number of pairs (e.g., $n = 64$ or $n = 128$), the correct subkey emerges near the top of the ranking. Specifically, when $n = 128$, the median key rank of the correct subkey is observed to be 1, demonstrating the high effectiveness of neural-distinguishers-based key recovery on 8-round GFRX-64/96. Algorithm 1 outlines the procedure for the neural-based key recovery attack on 8-round GFRX-64/96.

Algorithm 1 .Neural-based Key Recovery Attack on 8-round GFRX-64/96

Require: A trained 6-round neural distinguisher ($\mathcal{N}$), Number of ciphertext pairs n, input difference Δ_{in}.
Ensure: Ranking of candidate keys.
Generate n plaintext pairs (P_0^i, P_1^i) with input difference Δ_{in}.
Encrypt plaintext pairs through 8 rounds of GFRX-64/96, obtaining ciphertext pairs (C_0^i, C_1^i).
for each candidate 16-bit last-round subkey word κ (total 2^{16} keys) **do**
  for each ciphertext pair (C_0^i, C_1^i), $i = 1, \ldots, n$ **do**
    Partially decrypt one round using key guess κ (and known subkeys k_1^7, k_2^7) and obtain $(\tilde{C}_0^i, \tilde{C1}^i)$.
    Feed the intermediate ciphertext pair into the 6-round neural distinguisher $\mathcal{N}$ to obtain probability score Z_i^κ.
  end for
  Compute aggregated key score using (1).
end for
Rank all candidate keys κ according to their aggregated scores $v\kappa$ in descending order.
return Key ranking.

4 Differential cryptanalysis of GFRX-64

Desisigners of the GFRX explained that the GFRX-64/128 block cipher can be divided into two parts: $GFRX_L - 32/64$ and $GFRX_R - 32/64$. In this way, the GFRX-64/128 block cipher can be analyzed and evaluated in terms of these two mentioned ciphers. Since the security of the two ciphers, $GFRX_L - 32/64$ and $GFRX_R - 32/64$, is higher than that of the SIMON-32/64 block cipher, it follows that the GFRX-64/128 cipher possesses adequate security against differential and linear attacks. The designers stated that since no differential trail longer than 13 rounds has been identified for the SIMON-32/64 block cipher, it is reasonable to conclude that a longer differential trail cannot be found for the GFRX-64/128 block cipher either. Also, they said that for SIMON-32/64 the best-extended attack based on the 13-round differential trail has 19 rounds. So differential cryptanalysis could still be performed on 19 rounds of the GFRX-64/128 block cipher and more rounds of this cipher are resistant against differential cryptanalysis.

In this section, we present differential cryptanalysis on 11 rounds of GFRX-64. For this, we used a SAT/SMT-based automated method to find the best differential characteristic of the reduced rounds of GFRX-64. After finding the SAT/SMT model of the differential behavior of GFRX-64, we use the tool CryptoSMT and the SAT solver STP to resolve the problem and find a solution. For rounds 0 to 11, the solutions along with their success probability are listed in Table 4.

Table 4. Differential charecteristics for reduced rounds of GFRX-64/96 and GFRX-64/128

Round i	L_0^i	L_1^i	R_0^i	R_1^i	$-log_2 Pr$
0	$0x2220$	$0x0088$	$0x0000$	$0x0200$	5
1	$0x8800$	$0x0200$	$0x2000$	$0x8800$	6
2	$0x2002$	$0x0800$	$0x8000$	$0x2003$	5
3	$0x8008$	$0x2000$	$0x0002$	$0x800C$	6
4	$0x0022$	$0x8000$	$0x0008$	$0x0032$	6
5	$0x0088$	$0x0002$	$0x0020$	$0x00C8$	6
6	$0x0220$	$0x0008$	$0x0080$	$0x0320$	6
7	$0x0880$	$0x0020$	$0x0200$	$0x0C80$	6
8	$0x2200$	$0x0080$	$0x0800$	$0x3200$	5
9	$0x8800$	$0x0200$	$0x2000$	$0xC800$	6
10	$0x2002$	$0x0800$	$0x8000$	$0x2003$	5
11	$0x8008$	$0x2001$	$0x0002$	$0x800C$	$-\Sigma log_2 Pr = 62$

With respect to Table 4, the 11-round differential characteristic of input differential

$$(0x2220, 0x0088, 0x0000, 0x0200),$$

and output differential

$$(0x8008, 0x2001, 0x0002, 0x800C),$$

has the probability 2^{-62} and is the best 11-round differential characteristic for the block cipher GFRX-64/96 and also the block cipher GFRX-64/128.

It should be noted that, according to the claims of the GFRX block cipher designers, a 13-round differential characteristic exists for half of the GFRX-64/128 block cipher, whereas in this paper, an 11-round differential characteristic was obtained for the entire GFRX-64/128 block cipher. Although in the differential cryptanalysis of the GFRX-64 block cipher we aimed to identify differential effects rather than differential characteristics, our results didn't lead to the discovery of a differential distinguisher for more than 11 rounds.

5 Linear cryptanalysis of GFRX-64

Similar to the explanations provided in the section on differential cryptanalysis of the GFRX-64 block

cipher, the designers referenced the best linear cryptanalysis of the SIMON-32/64 block cipher to assess its security against linear attacks. They claimed that since the best linear cryptanalysis (using linear characteristics or their effects) does not exceed 13 rounds, a linear sequence for the GFRX-64/128 block cipher with more than 11 rounds can not be obtained, and attacks based on linear characteristics or linear hulls on this cipher do not exceed 13 rounds. We employed an automated linear cryptanalysis method based on SAT/SMT, and after modeling the linear behavior of the GFRX-64 block cipher and obtaining the related SAT problem, we utilized the CryptoSMT tool and the STP solver to derive a solution for the SAT problem. From this, we identified the linear characteristics for various rounds of the GFRX-64 block cipher. These characteristics are listed in Table 5. Similar to the differential analysis, it can be stated that since the GFRX-64/96 and GFRX-64/128 block ciphers are similar and differ only in their key schedule, and given that the key schedule does not affect the process of obtaining a linear characteristic of a block cipher, the characteristics presented for the GFRX-64 block cipher hold true for the GFRX-64/96 and GFRX-64/128 block ciphers.

Table 5. Linear characteristics for reduced rounds of GFRX-64/96 and GFRX-64/128

Round i	L_0^i	L_1^i	R_0^i	R_1^i	$-log_2(Corr)$
0	$0x0004$	$0x0001$	$0x0001$	$0x0004$	2
1	$0x0010$	$0x0004$	$0x0004$	$0x0010$	2
2	$0x0040$	$0x0010$	$0x0010$	$0x0040$	2
3	$0x0100$	$0x0040$	$0x0040$	$0x0100$	2
4	$0x0400$	$0x0100$	$0x0100$	$0x0400$	2
5	$0x1000$	$0x0400$	$0x0400$	$0x1000$	2
6	$0x4000$	$0x1000$	$0x1000$	$0x4000$	2
7	$0x0001$	$0x4000$	$0x4000$	$0x0001$	2
8	$0x0004$	$0x0001$	$0x0001$	$0x0004$	2
9	$0x0010$	$0x0004$	$0x0004$	$0x0010$	2
10	$0x0040$	$0x0010$	$0x0010$	$0x0040$	2
11	$0x0100$	$0x0040$	$0x0040$	$0x0100$	2
12	$0x0400$	$0x0100$	$0x0100$	$0x0400$	2
13	$0x1000$	$0x0400$	$0x0400$	$0x1000$	2
14	$0x4000$	$0x1000$	$0x1000$	$0x4000$	2
15	$0x0001$	$0x4000$	$0x4000$	$0x0001$	$-\Sigma log_2 Cor = 30$

With respect to Table 5, the 15-round linear characteristic of input mask

$$(0x1001, 0x4C00, 0x0420, 0x1000),$$

and output mask

$$(0x0001, 0x4000, 0x4000, 0x0001),$$

has the correlation 2^{-30} and is the best 15-round linear characteristic for the block cipher GFRX-64/96 and also the block cipher GFRX-64/128. We note that the designers of the GFRX claimed that no linear characteristic with more than 13 rounds exists for half of the GFRX cipher. However, here, a 15-round linear characteristic has been identified for the entire GFRX cipher.

A 17-round linear hull for the block cipher GFRX-64

Following the identification of a 15-round linear characteristic for the GFRX-64 block cipher, we employed its linear behavior model to derive a linear hull. We began by identifying a 17-round linear characteristic

with input mask

$$(0x0080, 0x0060, 0x0020, 0x0080),$$

and output mask

$$(0x0004, 0x0080, 0x0080, 0x0004),$$

and correlation 2^{-39}, similar to the 15-round characteristic search. Then explored other linear characteristics sharing these masks. We obtained 20413 linear characteristics with a correlation of 2^{-39}, 24750 linear characteristics with a correlation of 2^{-40}, 22069 with a correlation of 2^{-41}, and 16054 with a correlation of 2^{-42}. According to [37], combining n_i characteristics with the correlations of c_i yields a linear hull with a correlation of $2^{-\sqrt{n_i c_i^2}}$. Thus, these characteristics collectively form 17-round linear hulls with correlations of $2^{-31.84}$, $2^{-32.70}$, $2^{-33.78}$, and $2^{-35.01}$, respectively. When aggregated, they produce a 17-round linear hull with a correlation of $2^{-31.6075}$. The 17-round linear characteristics along with their correlations are summarized in Table 6.

Table 6. A 17-round linear hull for GFRX-64. c_i and n_i are the correlation and number of 17-round linear characteristics (LC).

$-\log_2(c_i)$	# 17-round LC (n_i)	$-\log_2(\sqrt{n_i c_i^2})$	$-\log_2(\sqrt{\sum_{c_j \geq c_i} n_j c_j^2})$
39	20413	31.8413	31.8413
40	24750	32.7024	31.6504
41	22069	33.7851	31.6139
42	16054	35.0146	31.6075

6 Conclusion

In this paper, we studied and examined the security of two versions of the GFRX family of block ciphers, with a block length of 64 bits and key lengths of 96 and 128 bits, against differential and linear cryptanalysis methods. Our analytical methods can be generalized to other versions of this family of block ciphers.

Based on the cryptanalysis conducted, we identified differential and linear distinguishers, along with some observations regarding the GFRX-64 cipher. We first demonstrated how to utilize a deep learning-based method to obtain neural distinguishers for rounds 1 to 7 of the GFRX-64/96 block cipher. Additionally, we presented an 8-round key recovery attack using a 6-round neural distinguisher on this cipher. We then employed an automated cryptanalysis method based on SAT/SMT, and after modeling the differential and linear behavior of the GFRX-64 block cipher as a SAT/SMT problem and solving it, we obtained an 11-round differential distinguisher based on differential characteristics, as well as 15 and 17-round distinguishers based on linear characteristics and linear hulls, respectively. The distinguishers of GFRX-64 were obtained for the first time through direct cryptanalysis of this block cipher, demonstrating that the claims made by the designers of the GFRX regarding the security of this cipher in relation to the security of the SIMON block cipher are not valid.

Acknowledgment

We would like to express our gratitude and appreciation to Mr. Iman Mirza Ali for his efforts and guidance in completing the section on cryptanalysis of GFRX-64/96 using deep learning.

References

[1] Ray Beaulieu, Douglas Shors, Jason Smith, Stefan Treatman-Clark, Bryan Weeks, and Louis Wingers. The simon and speck lightweight block ciphers. In *Proceedings of the 52nd annual design automation conference*, pages 1–6, 2015.

[2] Wen Chen, Lang Li, Ying Guo, and Ying Huang. Sand-2: An optimized implementation of lightweight block cipher. *Integration*, 91:23–34, 2023.

[3] Patricia Greene, Mark Motley, and Bryan Weeks. Aradi and llama: Low-latency cryptography for memory encryption. *Cryptology ePrint Archive*, 2024.

[4] Xing Zhang, Chenyang Shao, Tianning Li, Ye Yuan, and Changda Wang. Gfspx: an efficient lightweight block cipher for resource-constrained iot nodes. *The Journal of Supercomputing*, 80(17):25256–25282, 2024.

[5] Xingqi Yue, Lang Li, Qiuping Li, Jiahao Xiang, and Zhiwen Hu. Qlw: a lightweight block cipher with high diffusion. *The Journal of Supercomputing*, 81(1):224, 2025.

[6] Rashmi Sahay, Lingutla Lakshmi, and Zion Dodhiawala. Liteencrypt: A lightweight block cipher for secure communication in iot enabled sensor. *Internet Technology Letters*, page e613, 2024.

[7] Eli Biham and Adi Shamir. Differential cryptanalysis of des-like cryptosystems. *Journal of CRYPTOLOGY*, 4:3–72, 1991.

[8] Mitsuru Matsui. Linear cryptanalysis method for des cipher. In *Workshop on the Theory and Application of of Cryptographic Techniques*, pages 386–397. Springer, 1993.

[9] Eli Biham, Alex Biryukov, and Adi Shamir. Cryptanalysis of skipjack reduced to 31 rounds using impossible differentials. In *Advances in Cryptology—EUROCRYPT'99: International Conference on the Theory and Application of Cryptographic Techniques Prague, Czech Republic, May 2–6, 1999 Proceedings 18*, pages 12–23. Springer, 1999.

[10] Andrey Bogdanov and Meiqin Wang. Zero correlation linear cryptanalysis with reduced data complexity. In *Fast Software Encryption: 19th International Workshop, FSE 2012, Washington, DC, USA, March 19-21, 2012. Revised Selected Papers*, pages 29–48. Springer, 2012.

[11] Lars Knudsen and David Wagner. Integral cryptanalysis. In *Fast Software Encryption: 9th International Workshop, FSE 2002 Leuven, Belgium, February 4–6, 2002 Revised Papers 9*, pages 112–127. Springer, 2002.

[12] David Wagner. The boomerang attack. In *International Workshop on Fast Software Encryption*, pages 156–170. Springer, 1999.

[13] Susan K Langford and Martin E Hellman. Differential-linear cryptanalysis. In *Advances in Cryptology—CRYPTO'94: 14th Annual International Cryptology Conference Santa Barbara, California, USA August 21–25, 1994 Proceedings 14*, pages 17–25. Springer, 1994.

[14] Atiyeh Mirzaie, Siavash Ahmadi, and Mohammad Reza Aref. Boomerang attacks on reduced-round midori64. *ISeCure*, 15(2), 2024.

[15] Mehmet Emin Gönen, Muhammed Said Gündoğan, and Kamil Otal. Integral Cryptanalysis of Reduced-Round SAND-64 Based on Bit-Based Division Property. *ISeCure*, 16(2), 2023.

[16] Navid Vafaei, Maryam Porkar, Hamed Ramzanipour, and Nasour Bagheri. Practical differential fault analysis on skinny. *ISeCure*, 14(3), 2022.

[17] Hamed Ramzanipour, Navid Vafaei, and Nasour Bagheri. Practical differential fault analysis on craft, a lightweight block cipher. *ISeCure*, 14(3), 2022.

[18] Zhuolong Zhang, Shiyao Chen, Wei Wang, and Meiqin Wang. Full round distinguishing and key-recovery attacks on sand-2 (full version). *Cryptology ePrint Archive*, 2023.

[19] Javad Alizadeh, Hoda A Alkhzaimi, Moham-

mad Reza Aref, Nasour Bagheri, Praveen Gauravaram, Abhishek Kumar, Martin M Lauridsen, and Somitra Kumar Sanadhya. Cryptanalysis of simon variants with connections. In *International Workshop on Radio Frequency Identification: Security and Privacy Issues*, pages 90–107. Springer, 2014.

[20] Nicky Mouha, Qingju Wang, Dawu Gu, and Bart Preneel. Differential and linear cryptanalysis using mixed-integer linear programming. In *Information Security and Cryptology: 7th International Conference, Inscrypt 2011, Beijing, China, November 30–December 3, 2011. Revised Selected Papers 7*, pages 57–76. Springer, 2012.

[21] Emanuele Bellini, Alessandro De Piccoli, Mattia Formenti, David Gerault, Paul Huynh, Simone Pelizzola, Sergio Polese, and Andrea Visconti. Differential cryptanalysis with sat, smt, milp, and cp: a detailed comparison for bit-oriented primitives. In *International Conference on Cryptology and Network Security*, pages 268–292. Springer, 2023.

[22] Ling Sun, Wei Wang, and Meiqin Wang. Accelerating the search of differential and linear characteristics with the sat method. *IACR Transactions on Symmetric Cryptology*, pages 269–315, 2021.

[23] Aron Gohr. Improving attacks on round-reduced speck32/64 using deep learning. In *Advances in Cryptology–CRYPTO 2019: 39th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18–22, 2019, Proceedings, Part II 39*, pages 150–179. Springer, 2019.

[24] Emanuele Bellini, Mattia Formenti, David Gérault, Juan Grados, Anna Hambitzer, Yun Ju Huang, Paul Huynh, Mohamed Rachidi, Raghvendra Rohit, and Sharwan K Tiwari. Claasping aradi: Automated analysis of the aradi block cipher. In *International Conference on Cryptology in India*, pages 90–113. Springer, 2024.

[25] Dongsu Shen, Yijian Song, Yuan Lu, Saiqin Long, and Shujuan Tian. Neural differential distinguishers for gift-128 and ascon. *Journal of Information Security and Applications*, 82:103758, 2024.

[26] Wanqing Wu and Mingyu Guo. Improved integral neural distinguisher model for lightweight cipher present. *Cybersecurity*, 7(1):1–14, 2024.

[27] Xing Zhang, Shaoyu Tang, Tianning Li, Xiaowei Li, and Changda Wang. Gfrx: A new lightweight block cipher for resource-constrained iot nodes. *Electronics*, 12(2):405, 2023.

[28] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[29] Nicky Mouha and Bart Preneel. Towards finding optimal differential characteristics for arx: Application to salsa20. *Cryptology ePrint Archive*, 2013.

[30] Trevor Hansen, Norbert Manthey, and Mate Soos. Stp in the smtcomp 2019.

[31] Aina Niemetz, Mathias Preiner, and Armin Biere. Boolector 2.0. *Journal on Satisfiability, Boolean Modeling and Computation*, 9(1):53–58, 2014.

[32] Stefan Kölbl. Cryptosmt: An easy to use tool for cryptanalysis of symmetric primitives.

[33] Anubhab Baksi and Anubhab Baksi. Machine learning-assisted differential distinguishers for lightweight ciphers. *Classical and Physical Security of Symmetric Key Cryptographic Algorithms*, pages 141–162, 2022.

[34] Emanuele Bellini, David Gerault, Anna Hambitzer, and Matteo Rossi. A cipher-agnostic neural training pipeline with automated finding of good input differences. *IACR Transactions on Symmetric Cryptology*, 2023(3):184–212, 2023.

[35] Liu Zhang, Zilong Wang, et al. Improving differential-neural cryptanalysis. *Cryptology ePrint Archive*, 2022.

[36] Iman Mirzaali Mazandarani, Nasour Bagheri, and Sadegh Sadeghi. A comprehensive exploration of deep learning approaches in differential cryptanalysis of lightweight block ciphers. *Biannual Journal Monadi for Cyberspace Security (AFTA)*, 12(1), 2023.

[37] Antonio Flórez-Gutiérrez and María Naya-Plasencia. Improving key-recovery in linear attacks: Application to 28-round present. In *Advances in Cryptology–EUROCRYPT 2020: 39th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, May 10–14, 2020, Proceedings, Part I 39*, pages 221–249. Springer, 2020.

Javad Alizadeh earned his Ph.D. in Cryptography in 2016, supervised by Dr. M. Aref and Dr. N. Bagheri. He is currently an Assistant Professor at the Faculty and Research Center of Computer at Imam Hossein university. His research focuses on the cryptanalysis of symmetric ciphers, particularly block ciphers.

Bahman Madadi earned his master's degree in Telecommunications in 2011 and is now pursuing a Ph.D. in the same field at Imam Hossein university. His research focuses on the cryptanalysis of block ciphers.