IACR Transactions on Symmetric Cryptology
ISSN 2519-173X, Vol. 0, No. 0, pp. 0–0. DOI:10.46586/tosc.v0.i0.0-0

Computer-Aided Proof for Extended Generalized Feistel Networks

Yuchao Chen[1,4,5], Chun Guo[1,4,5(✉)], Muzhou Li[2,4,5,6(✉)], Shuo Peng[1,4,5], Hao Lei[1,4,5], Guang Zeng[3] and Meiqin Wang[1,4,5,6(✉)]

[1] School of Cyber Science and Technology, Shandong University, Qingdao, China
{chenyuchao,pengshuo,leihao}@mail.sdu.edu.cn
chun.guo.sc@gmail.com
mqwang@sdu.edu.cn
[2] School of Cryptologic Science and Engineering, Shandong University, Jinan, China
muzhouli@mail.sdu.edu.cn
[3] Huawei technologies, Beijing, China
zengguang13@huawei.com
[4] State Key Laboratory of Cryptography and Digital Economy Security, Shandong University, Qingdao, 266237, China
[5] Key Laboratory of Cryptologic Technology and Information Security, Ministry of Education, Shandong University, Jinan, China
[6] Quancheng Laboratory, Jinan, 250103, China

Abstract. (Multi-branch) Generalized Feistel Network (GFN) enables the construction of block ciphers from non-linear components with small domains, and has been adopted in various block ciphers. Berger et al. (SAC 2013) introduced the Extended Generalized Feistel Network (EGFN), which unified and extended existing Feistel-like structures by using a matrix representation.

Given an arbitrary matrix, it is typically difficult to determine how many EGFN rounds are sufficient for pseudorandom permutation (PRP) and strong PRP (SPRP) security. Remarkably, security proofs for structures with a larger number of branches have to analyze a huge amount of collision events, which is overly complicated and prone to errors.

To remedy this situation, we present AutoEGFN, a computer-aided proof tool that determines the number of rounds sufficient for PRP and SPRP security for various variants of EGFN. The tool operates by calculating three parameters: r_1, r_2, and r_3. The validity and soundness of AutoEGFN are formally established by a detailed security proof. To demonstrate the effectiveness of AutoEGFN, we have applied it to multiple structures such as Type-1/2 GFN (Zheng et al., CRYPTO 1989), YI11's Type-1 GFN (Yanagihara and Iwata, CANS 2011), DFLM19's GFN (Derbez et al., FSE 2019), DDGP22's GFN (Delaune et al., INDOCRYPT 2022), Type-1.x GFN (Yanagihara and Iwata, IEICE 2014), SH/TH GFN (Yanagihara and Iwata, CANS 2011), Nyberg's GFN (Nyberg, ASIACRYPT 1996), SM's GFN (Suzaki and Minematsu, FSE 2010), and BMT's EGFN (Berger et al., SAC 2013). As a result, we provide a systematic analysis of the (S)PRP security for Type-1 and Type-2 structures for different numbers of branches. Our tool efficiently determines the concrete number of rounds required to ensure PRP and SPRP security for EGFNs with different branch numbers. For comparison, previous work only proved the (S)PRP security for 8- and 16-branch BMT's EGFN. Our tool completes the proof within several minutes, even for variants with 32 branches. Meanwhile, for the other structures, we provide the first concrete (S)PRP security proofs without any restrictions on their permutation layers. Furthermore, AutoEGFN will significantly contribute to the enhancement of EGFN designs and implementations in various cryptographic applications.

Licensed under Creative Commons License CC-BY 4.0.

Keywords: Computer-Aided Proof · PRP and SPRP · Generalized Feistel Networks · Extended GFN

1 Introduction

1.1 Feistel-Like Structures

The classical Feistel network was introduced by Luby and Rackoff (LR for short) [LR85, LR88], providing a method for constructing a pseudorandom permutation (PRP) from a pseudorandom function (PRF). Each round of the Feistel network takes two n-bit blocks (P_1, P_2) as input and outputs $(P_2, P_1 \oplus F(P_2))$, where the key-dependent PRF F is referred to as the round function. A key advantage of the Feistel network is its inverse-free property: both encryption and decryption require only the forward evaluation of the underlying PRF. This leads to lower implementation costs compared to Substitution-Permutation Networks (SPNs). LR further demonstrated that a 3-round Feistel network is a secure PRP, while a 4-round one achieves Strong PRP (SPRP) security.

Since its proposal, there has been lots of research focusing on how to improve the Feistel network. For example, Naor and Reingold [NR99] showed that replacing the first and last rounds of the 4-round Feistel network with an Almost Universal (AU) function relaxes the security requirements of the original scheme. Additionally, they proposed a high-level framework for designing SPRPs. At FSE 1996, Schneier and Kelsey [SK96] proposed Unbalanced Feistel Networks (UFNs) with contracting and expanding functions, also known as Source-Heavy (SH) and Target-Heavy (TH) GFNs, which are another variant of the Feistel network. Lucks [Luc96] proposed the Altering Feistel structure, which alternates between using two different round functions: $F : \{0,1\}^n \to \{0,1\}^m$ in rounds with even numbers and $F' : \{0,1\}^m \to \{0,1\}^n$ in rounds with odd numbers. Thereafter, numerous studies [MV00, MP03, NPV17] analyzed the security of the above structures.

At CRYPTO 1989, Zheng et al. [ZMI89] proposed the Type-1, Type-2, and Type-3 structures, which can work for messages divided into more than 2 parts. All three types of structures take d sub-blocks as inputs, with d being an integer greater than or equal to 2. The most famous structure is Type-2, and its output for a single round is

$$(C_0, C_1, \cdots, C_{d-1}) = (P_1 \oplus F_0(P_0), P_2, \cdots, P_{d-1} \oplus F_{(d-2)/2}(P_{d-2}), P_0)$$

for input $(P_0, P_1, \cdots, P_{d-1})$, where each F_i denotes a round function. For a fixed message length, using a round function with a shorter width allows for a larger number of branches, making this structure suitable for small-scale implementations. Regarding their security, Zheng et al. [ZMI89] proved that $(2d-1)$ rounds of Type-1, $(d+1)$ rounds of Type-2, and $(d+1)$ rounds of Type-3 achieve PRP security. Their asymptotic SPRP security bounds are improved by Shen et al. [SGW20]. They proved that $((d^2+d-2)t+1)$ rounds of Type-1, $(2dt+1)$ rounds of Type-2, and $((d+2)t+1)$ rounds of Type-3 achieve SPRP security, respectively, where t is a parameter corresponding to different security bounds.

Later at ASIACRYPT 1996, Nyberg formally introduced the concept of Generalized Feistel Networks (GFNs) [Nyb96], which had a significant influence on the design and analysis of structures. Nowadays, GFNs have become a fundamental and widely used structure in block cipher design. Many well-known block ciphers are based on GFNs, including GMiMC[AGP+19] ((un)balanced Feistel Network), CAST-256 [AG99] (Type-1), RC6 [RRSY98], CLEFIA [SSA+07] (Type-2), LEA [HLK+13] (Type-3), TWINE [SMMK12]. Nyberg also proposed a specific instantiation in [Nyb96], which is a Type-2 variant that employs a special blockwise permutation in place of the cyclic shift. We refer to this variant as Nyberg's GFN. While the security of Nyberg's GFN against differential and linear cryptanalysis has been evaluated, few studies have established its concrete (S)PRP

security. At IEICE 2014, Yanagihara and Iwata [YI14] proposed the Type-1.x GFN, which further extends the classical Type-1 and Type-2 frameworks.

The primary drawback of Type-2 is its low diffusion property. To address this, Suzaki and Minematsu [SM10] (SM for short) improved GFN by using a more general blockwise permutation π per round to achieve better diffusion. Formally, for an input $(P_0, P_1, \cdots, P_{d-1})$, the output is given by:

$$(C_0, C_1, \cdots, C_{d-1}) = \pi(P_0, P_1 \oplus F_0(P_0), \cdots, P_{d-2}, P_{d-1} \oplus F_{d/2-1}(P_{d-2})).$$

Other works [YI11, YI13, DFLM19, DDGP22] aim to improve GFN by searching for an optimal π. This optimization aims to achieve various security goals, including the minimum number of rounds for full diffusion [SM10], resistance to generic attacks [Jut98, PNB06, PNB07], (impossible) differential or linear cryptanalysis [Shi10, KHS+03, NVP13], as well as improved security bounds [HR10a, HR10b, SGW20]. Notable structures include YI11's Type-1 GFN [YI11], DFLM19's GFN [DFLM19], and DDGP22's GFN [DDGP22]. However, all these novel structures lack rigorous security proofs, especially for their PRP and SPRP security.

At SAC 2013, Berger, Minier, and Thomas (BMT for short) [BMT13] took a novel view to analyze the structure of GFN by matrix representation. They divide each round of GFN into three layers: the round-function layer $\mathcal{F}$, the linear layer $\mathcal{L}$, and the permutation layer $\mathcal{P}$. The new class of constructions they proposed is called Extended Generalized Feistel Networks (EGFNs) with matrix representation $\mathcal{M} = \mathcal{P}\mathcal{L}\mathcal{F}$. This matrix-based framework provides a unified perspective on GFNs, enabling the study of numerous properties. BMT utilized this framework to analyze the diffusion properties of EGFNs and evaluate their security against classical attacks. Subsequent work [BFMT16] instantiated the EGFN framework into a lightweight block cipher called Lilliput, analyzing its security and hardware performance. The above-mentioned Type-1/2, YI11's Type-1 GFN, DFLM19's GFN, DDGP22's GFN, Type-1.x GFN, SH/TH GFN, Nyberg's GFN, and SM's GFN, can all be viewed as special instances of EGFNs.

1.2 Computer-aided Proof

Provable security plays an important role in the development of cryptography. This field originated from the pioneering work of Goldwasser and Micali [GM84], who first introduced the formal notions of polynomial and semantic security for public-key cryptosystems. Subsequently, Goldreich, Goldwasser, and Micali [GGM86] adapted the concept of "probabilistic encryption" defined above to the field of symmetric cryptography and formally defined the pseudorandom function for the first time. They also introduced the notion of "computational indistinguishability" to measure the randomness of functions, thereby establishing the mathematical foundation for the security of symmetric cryptographic primitives, including block ciphers and message authentication codes (MACs). LR [LR88] introduced the notion of the pseudorandom permutation and demonstrated how to construct a (S)PRP from a PRF, which significantly promoted research on the security of block ciphers. Bellare et al. [BDJR97] modeled chosen-plaintext attacks through four security notions: "real-or-random", "left-or-right", "find-then-guess", and "semantic security", and proved the security of the CTR and CBC modes within a concrete security framework.

Common proof methods in the field of symmetric cryptography include the Random Oracle Model (ROM) [BR93], the game-playing technique [BG90, Sho01, BR06], and the coefficient-H technique [Pat08]. The core of proofs is a reduction argument, which reduces the security of a scheme to that of its underlying primitives. However, the correctness and quality of manual proofs depend primarily on the expertise of the prover, making them prone to errors. For instance, Inoue et al. [IIMP19] demonstrated practical universal

forgeries and plaintext recovery attacks against OCB2, identifying flaws in its security proof. They showed that the security of the underlying XEX* mode (the combination of XE and XEX) does not always imply the security of OCB2; Li et al. [YL16] pointed out the errors in the proof of the CCTR mode and proposed a forgery attack that requires only a single query to the tag generation algorithm.

Consequently, computer-aided proof is a novel research hotspot, as it can significantly reduce the heavy workload of provers and assist in the design of provably secure encryption schemes. Several works have employed computer-aided proofs for cryptographic constructions such as authenticated encryption (AE) schemes [HKM15], modes of operation [Mea20, AFH24], and MACs [MPS+25]. Notably, Ambrona et al. [AFH24] presented an automated tool called AlgoROM, which focuses on symmetric primitives in the ROM setting, including Feistel networks, SPNs, key-alternating Feistel ciphers, and iterated Even-Mansour ciphers.

To the best of our knowledge, no existing computer-aided proof tool can handle various types of EGFN. Moreover, the security analysis provided in [BMT13] was conducted manually and only for specific EGFN instances with 8 and 16 branches. For variants with a larger number of branches, there is a lack of systematic security research regarding the number of rounds ensuring (S)PRP security. This motivates us to develop a computer-aided proof tool capable of analyzing various types of EGFN.

1.3 Our Contributions

We present AutoEGFN, the first computer-aided proof tool designed for the security analysis of Extended Generalized Feistel Networks (EGFNs). This tool provides a unified and automated framework for the design and analysis of different constructions.

1) We introduce the construction of AutoEGFN in Section 3, which automates security proofs by computing three key parameters: r_1, r_2, and r_3. Parameters r_1 and r_3 are computed using an SMT constraint solver (STP [GHS+14] or Z3 [Mic25]), while r_2 is derived via SageMath [SD20]. This automated approach minimizes errors in manual proofs and significantly simplifies the proof process.
2) We manually prove that any $(r_1 + r_2)$-round EGFN with branch numbers d is PRP-secure, and any $(r_1 + r_2 + r_3)$-round EGFN with branch numbers d is SPRP-secure in Section 4. Since [BMT13] proved the security only for specific EGFN instances, we adopt the proof framework of [NR99] and extend it using our method to establish the security of arbitrary EGFNs. For an adversary making q queries, our results indicate that the PRP security bound for $(r_1 + r_2)$-round EGFNs is $O(\frac{q^2 \cdot |\tau_1|^2 r_1}{2^n} + \frac{q^2}{2^{dn+1}})$, and the SPRP security bound for $(r_1+r_2+r_3)$-round EGFNs is $O(\frac{q^2 \cdot (|\tau_1|^2 r_1 + |\tau_2|^2 r_3)}{2^n} + \frac{q^2}{2^{dn}})$, respectively.
3) In Section 5.1 and Appendix A, we apply AutoEGFN to a wide range of structures, including Type-1/2, YI11's Type-1 GFN, DFLM19's GFN, DDGP22's GFN, Type-1.x GFN, SH/TH GFN, Nyberg's GFN, SM's GFN, and BMT's EGFN. Table 1 compares the results for Type-1/2 GFNs with prior works [ZMI89, MV00, SGW20]. These comparisons confirm that our results align with existing manual proofs [ZMI89, MV00] and, in particular, demonstrate an improvement in SPRP security for Type-1 GFNs over those in [SGW20].
   To the best of our knowledge, concrete security proofs for the other eight structures are largely absent. A comparison between previous works and ours is summarized in Table 2. For structures with existing security bounds, such as SM's GFN and BMT's EGFN, our results align with theirs. For the remaining structures, AutoEGFN provides the first systematic security proofs. Our tool completes these proofs within minutes, even for variants with 32 branches, demonstrating its capability to handle configurations whose complexity precludes manual verification. The detailed running

time for some of these proofs is provided in Section 5.2.

Table 1: Comparing the security round with previous results of Type-1/2. The "# of rounds" column indicates the number of rounds required to achieve (S)PRP security.

Structure	Branch numbers	Previous # of rounds		Our # of rounds	
		PRP [ZMI89]	SPRP	PRP	SPRP
Type-1	4	7	19 [SGW20]	7	**16**
	6	11	41 [SGW20]	11	**36**
	8	15	71 [SGW20]	15	**64**
	10	19	109 [SGW20]	19	**100**
	12	23	155 [SGW20]	23	**144**
	14	27	209 [SGW20]	27	**196**
	16	31	271 [SGW20]	31	**256**
	32	63	1055 [SGW20]	63	**1024**
Type-2	4	5	8 [MV00]	5	8
	6	7	12 [MV00]	7	12
	8	9	16 [MV00]	9	16
	10	11	20 [MV00]	11	20
	12	13	24 [MV00]	13	24
	14	15	28 [MV00]	15	28
	16	17	32 [MV00]	17	32
	32	33	64 [MV00]	33	64

Table 2: Computer-aided proof results among different structures.

Structure	Previous proof results	Our proof results
YI11's Type-1 GFN [YI11]	No security proof.	Table 5
DFLM19's GFN [DFLM19]		Table 6
DDGP22's GFN [DDGP22]		Table 7
Type-1.x GFN [YI14]		Table 9
SH/TH GFN [SK96]		Table 10
Nyberg's GFN [Nyb96]		Table 11
SM's GFN [SM10]	Only structures with even-odd permutation proven in [SM10].	Table 12†
BMT's EGFN [BMT13]	Only 8- and 16-branch structures have been proved [BFMT16].	Table 13‡

† AutoEGFN can prove all structures with arbitrary given permutations.
‡ AutoEGFN finishes proofs for any branch numbers.

To support the reproduction of our results, all source codes, along with detailed instructions, are available at https://github.com/taixujianyi/AutoEGFN.

Outline of this paper. The paper is structured as follows: After Section 2 reviews the preliminary concepts and definitions, Section 3 introduces the tool AutoEGFN and provides an illustrative example to facilitate understanding of its construction. Section 4 presents the detailed security proof for EGFNs behind the automated tool AutoEGFN. Section 5 discusses the computer-aided proof results for various EGFN structures, as well as the efficiency of our tool. Finally, Section 6 concludes this work.

2 Preliminaries

2.1 Notations

We define $\{0,1\}^*$ as the set of all bit strings of arbitrary finite length and $\{0,1\}^n$ as the set of all bit strings of length n. We adopt the following typographical conventions: lowercase letters x, y denote integers or bit strings; $\vec{x}$ denotes vectors; uppercase letters X, Y represent strings or functions; and curlicue uppercase letters $\mathcal{X}, \mathcal{Y}$ denote sets. Let $x[i]$ represent the i-th bit of x. Let $X\|Y$ denote the concatenation of strings X and Y, and let $X \oplus Y$ denote their bitwise exclusive-OR (XOR). We define $\mathbb{Z}[X]$ as the set of integers containing the element X. Let $\emptyset$ denote the empty set, $|X|$ as the length of X, and $x \xleftarrow{\$} \mathcal{X}$ indicate that x chosen uniformly at random from the set $\mathcal{X}$. Let $\mathsf{Perm}(\mathcal{S})$ denote the set of all permutations over the set $\mathcal{S}$, and $\mathsf{Func}(\mathcal{X}, \mathcal{Y})$ the set of all functions with domain $\mathcal{X}$ and codomain $\mathcal{Y}$. For any $F, G \in \mathsf{Func}(\mathcal{X}, \mathcal{Y})$, their composition is denoted by $F \circ G$ (i.e., $F \circ G(x) = F(G(x))$).

We define $\Pr[V]$ as the probability of event V. Consider an adversary $\mathbf{A}$ that runs in time at most t and makes at most q queries in total, interacting with given oracles in a black-box manner. The output of $\mathbf{A}$ after interacting with oracle $\mathcal{O}$ is denoted $\mathbf{A}^{\mathcal{O}} \Rightarrow b$, where $b \in \{0,1\}$. The advantage of $\mathbf{A}$ in distinguishing between oracles R (the real oracle) and I (the ideal oracle) for a security goal GOAL against primitive E is defined as:

$$\mathbf{Adv}_E^{\mathrm{GOAL}}(\mathbf{A}) \coloneqq \left|\Pr[\mathbf{A}^R \Rightarrow 1] - \Pr[\mathbf{A}^I \Rightarrow 1]\right|.$$

We then define the maximum advantage over all such adversaries as:

$$\mathbf{Adv}_E^{\mathrm{GOAL}}(q,t) \coloneqq \max_{\mathbf{A}} \left\{\mathbf{Adv}_E^{\mathrm{GOAL}}(\mathbf{A})\right\}.$$

Definition 1. (Almost Universal Hash Functions) Fix integers $d, n \geq 1$. Let $\mathcal{X}, \mathcal{Y} \subseteq \{0,1\}^{dn}$, and let $\mathcal{K}$ be a key space. Define $\vec{x} = (x_0, \cdots, x_{d-1}) \in \{0,1\}^{dn}$ with $\vec{x}_{[i]} = x_i$ $(0 \leq i < d)$ and τ as an index set. A keyed function $H : \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$ is called **ϵ-almost universal with respect to** τ (denoted ϵ-AU$_\tau$) if for any two distinct inputs $\vec{x}, \vec{x}' \in \mathcal{X}$ and for every $i \in \tau$, the following holds:

$$\Pr[K \xleftarrow{\$} \mathcal{K} : H_K(\vec{x})_{[i]} = H_K(\vec{x}')_{[i]}] \leq \epsilon.$$

Definition 2 (PRP Advantage)**.** Fix integers $k, d, n \geq 1$. Let $E : \{0,1\}^k \times \{0,1\}^{dn} \to \{0,1\}^{dn}$ be a keyed permutation and $\mathbf{A}$ be an adversary with access to the encryption oracle. Let $K \xleftarrow{\$} \{0,1\}^k$ and $\pi \xleftarrow{\$} \mathsf{Perm}(\{0,1\}^{dn})$. Then, the PRP advantage of $\mathbf{A}$ against E is defined as:

$$\mathbf{Adv}_E^{\mathrm{PRP}}(\mathbf{A}) \coloneqq \left|\Pr[\mathbf{A}^{E_K} \Rightarrow 1] - \Pr[\mathbf{A}^{\pi} \Rightarrow 1]\right|.$$

Definition 3 (Strong PRP Advantage)**.** Fix integers $k, d, n \geq 1$. Let $E : \{0,1\}^k \times \{0,1\}^{dn} \to \{0,1\}^{dn}$ be a keyed permutation and $\mathbf{A}$ be an adversary with access to both encryption and decryption oracles. Let $K \xleftarrow{\$} \{0,1\}^k$ and $\pi \xleftarrow{\$} \mathsf{Perm}(\{0,1\}^{dn})$. The Strong PRP advantage of $\mathbf{A}$ against E is defined as:

$$\mathbf{Adv}_E^{\mathrm{SPRP}}(\mathbf{A}) \coloneqq \left|\Pr[\mathbf{A}^{E_K, E_K^{-1}} \Rightarrow 1] - \Pr[\mathbf{A}^{\pi, \pi^{-1}} \Rightarrow 1]\right|.$$

2.2 The Matrix Representation of EGFNs

Let EGFN$_{r,dn}$ denote an r-round EGFN structure with d branches, where $r > 0$ and $d \geq 2$. The structure divides the input message into d blocks of n bits each. Denote the input blocks by $(x_0, \cdots, x_{d-1})$ and the corresponding output blocks by $(y_0, \cdots, y_{d-1})$. As shown

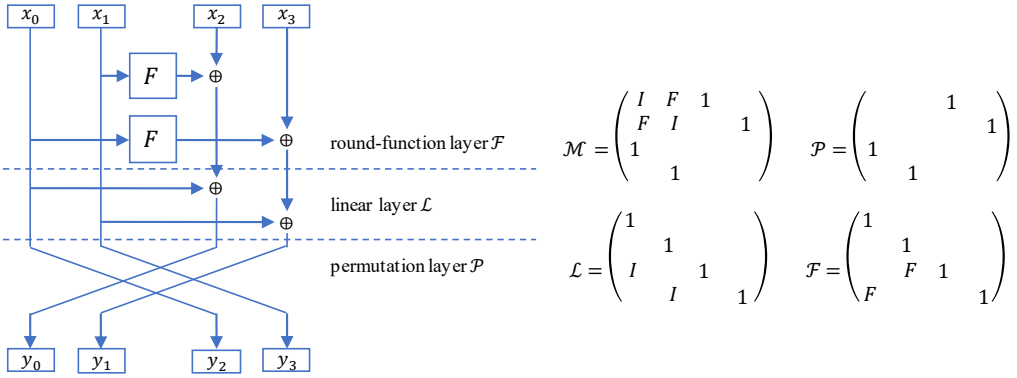


Figure 1: An example of the EGFN's three layers (left) and the corresponding matrix representation (right) for $d = 4$.

in [BMT13], an EGFN can be decomposed into three distinct layers: the round-function layer, the linear layer, and the permutation layer. Figure 1 illustrates this decomposition for a 4-branch EGFN.

We say that an input block x_i *affects* an output block y_j if x_i appears effectively in the expression of y_j, for $0 \leq i, j < d$. The EGFN structure comprises three layers:

- The *round-function layer* consists of non-linear round functions F that are related to the key. Each F takes certain blocks as input, and its output is XORed with blocks from other branches.
- The *linear layer* comprises a series of linear operations such as COPY and XOR.
- The *permutation layer* applies a blockwise permutation of d branches to achieve fast diffusion.

Different EGFN variants are characterized by the instantiation of the round function in the round-function layer. We focus on EGFNs that possess the quasi-involutiveness property (where encryption and decryption are almost identical processes), which satisfy Definition 4.

Definition 4 (EGFN matrix [BMT13])**.** A matrix $\mathcal{M}$ with coefficients in $\{0, 1, F, I\} \subset \mathbb{Z}[F, I]$ is an EGFN matrix if it can be expressed as $\mathcal{M} = \mathcal{P}\mathcal{N}$, where $\mathcal{P}$ is a permutation matrix and $\mathcal{N} = \mathcal{L}\mathcal{F}$ satisfies:

- The main diagonal entries are all 1;
- The off-diagonal coefficients are either 0, F, or I;
- For each index i, neither row i nor column i can both contain a non-zero coefficient, except for the diagonal;
- For each index i, if row i contains an I, then it must also contain an F.

This definition encompasses most known GFNs but excludes the well-known Type-3 structure, whose round functions must be evaluated sequentially. Therefore, Type-3 is not considered in this work.

The matrix $\mathcal{F}$, which corresponds to the round-function layer, is defined using the element 1 and the symbol F. All main diagonal elements of $\mathcal{F}$ are 1. An element F at position (i, j) indicates the existence of a round function that takes its input from the j-th branch and whose output affects the i-th branch, where $0 \leq i, j < d$. Here, F serves as an

abstract parameter denoting the presence of a round function, and the same F is used across different rounds of the structure. The matrix $\mathcal{L}$, representing the linear layer, is defined by elements 0 and I. $\mathcal{L}$ is structurally similar to $\mathcal{F}$, except that its off-diagonal non-zero elements are the coefficient I instead of F. Finally, the matrix $\mathcal{P}$ corresponds to the permutation layer and represents a blockwise permutation of the branches.

Thus, the EGFN can be represented by the matrix $\mathcal{M} = \mathcal{P} \times \mathcal{L} \times \mathcal{F}$, where $\mathcal{M}$ is a $d \times d$ matrix over $\mathbb{Z}[F, I]$. For indices $0 \leq i, j < d$:

- If the element at position (i, j) of $\mathcal{M}$ is 1, then the output y_i is directly determined by the input x_j without being affected by the round function F.
- If the element is F, then x_j is processed through the round function F to influence the output y_i.
- If the element is 0, then x_j has no direct effect on y_i.

The matrix $\mathcal{M}$ does not provide a complete specification of EGFNs, but it is sufficient for analyzing how each output block y_i is influenced by the input blocks $(x_0, \cdots, x_{d-1})$, either directly or through the round function F. Since EGFNs are invertible permutations built upon non-invertible round functions, the corresponding inverse matrix $\mathcal{M}^{-1}$ for decryption must not contain the inverse of F. This requirement implies that $\det(\mathcal{M})$ must be independent of F and I, and specifically, $\det(\mathcal{L} \cdot \mathcal{F}) = \pm 1$.

3 AutoEGFN: a Computer-aided Proof Tool

In this section, we present the computer-aided proof tool AutoEGFN. The tool is built upon three key parameters r_1, r_2, and r_3, whose definitions are provided in Section 3.1, alongside a simple example illustrating their calculations. Section 3.2 describes how these parameters are computed automatically for any given EGFN using an SMT solver and SageMath. Combined with the security proof in Section 4, this shows that $(r_1 + r_2)$ rounds ensure PRP security, while $(r_1 + r_2 + r_3)$ rounds are sufficient for SPRP security.

3.1 Basic Concepts used in AutoEGFN

Definition 5 (Parameters r_1 and r_3)**.** For an r-round EGFN, the parameter r_1 (resp., r_3) is the minimum number of rounds such that every input to the round functions at the $(r_1 + 1)$-th (resp., $(r_3 + 1)$-th) round is affected by every input block from the first round during encryption (resp., decryption), where $1 \leq r_1, r_3 < r$.

The parameter r_1 (resp., r_3) captures the diffusion property of EGFNs in the encryption (resp., decryption) direction. The notions r_1 and r_3 are similar to the notion "DR_i" defined in [SM10, Definition 1] or the "full diffusion delay" defined in [BMT13, Section 1.2]. In contrast, our definition specifically tracks the effect on the inputs to all round functions F, rather than focusing on the diffusion to all output sub-blocks after a given round. As established in Section 4, this specific form of diffusion property is enough for our security proof.

Definition 6 (Parameter r_2)**.** Let the output $\vec{c}$ of $\mathrm{EGFN}_{r_2,dn}$ be a linear function of all applied round functions, expressed as $\vec{c} = M \cdot \vec{F}$, where $\vec{F}$ is the vector containing all round function instances F_j^i applied over r_2 rounds, where the superscript i denotes the round number and the subscript j denotes the branch position. Then, r_2 is defined as the minimum number of rounds for which the transformation matrix M achieves full rank. Here, r_2 is a positive integer less than r.

The parameter r_2, as defined above, is the minimum number of rounds required for M to achieve full rank. This full-rank condition is a sufficient criterion for ensuring that the output ciphertext is both random, meaning that the ciphertext is effectively randomized by employing all available round functions, and independent, meaning the expressions for distinct ciphertext blocks are linearly independent as functions of the round function outputs. This prevents exploitable correlations between different blocks of the ciphertext. Thus, achieving r_2 rounds is fundamental to the security of the construction.

We illustrate the parameter evaluation process using the 4-branch Type-2 structure as a concrete example. Let $\text{Type2}_{r,4n}$ denote the r-round 4-branch Type-2 structure, where each branch is n bits wide, and the round functions in the i-th round are denoted as F_1^i and F_2^i.

Determination of r_1 and r_3. Figure 2 shows the diffusion path of $\text{Type2}_{3,4n}$ in the encryption direction, demonstrating that the inputs to both F_1^4 and F_2^4 are affected by every input block x_j $(0 \leq j \leq 3)$. Thus, according to Definition 5, $r_1 = 3$. Since the encryption and decryption processes of Type-2 are similar, we can derive $r_3 = 3$ in a similar way. Notice that this structure achieves full diffusion under the definition in [SM10, BMT13] only after 4 rounds.

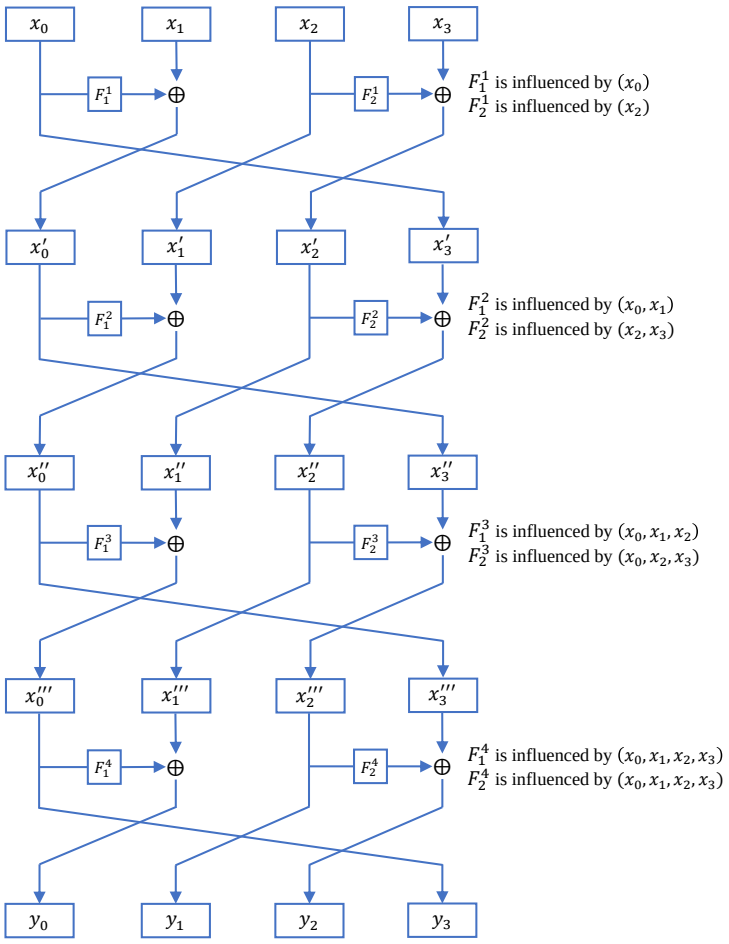


Figure 2: Encryption process of the 4-round Type-2 structure.

Determination of r_2. According to Definition 6, this requires verifying whether the matrix M formed by the output ciphertexts about the round functions is full rank. When

$r_2 = 2$, we can see that

$$x_0'' = x_1' \oplus F_1^2(x_0') = x_2 \oplus F_1^2(x_1 \oplus F_1^1(x_0)),\ x_1'' = x_2' = x_3 \oplus F_2^1(x_2),$$
$$x_2'' = x_3' \oplus F_2^2(x_2') = x_0 \oplus F_2^2(x_3 \oplus F_2^1(x_2)),\ x_3'' = x_0' = x_1 \oplus F_1^1(x_0).$$

We construct the matrix M by representing each output branch x_i'' as a row vector over the basis $(F_1^1, F_2^1, F_1^2, F_2^2)$, where a component is 1 if the output depends on the corresponding round function, and 0 otherwise. According to the above equations, the output x_0'' depends on F_1^1 and F_1^2, which corresponds to the first row vector $[1, 0, 1, 0]$ in Equation 1. The vectors for the subsequent outputs are $[0, 1, 0, 0]$, $[0, 1, 0, 1]$, and $[1, 0, 0, 0]$, respectively. Thus, the matrix M is:

$$M = \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{pmatrix}. \tag{1}$$

We can obtain the matrix M' after applying elementary row operations:

$$M' = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{pmatrix}.$$

Since M' has full rank and elementary row operations preserve rank, the original matrix M is also of full rank. This confirms that $r_2 = 2$ according to Definition 6.

3.2 Construction of AutoEGFN

AutoEGFN comprises two core algorithms: Algorithm 1 evaluates r_1 and r_3 via an SMT constraint solver using SMT-LIB2, and Algorithm 2 computes r_2 using SageMath. Let $\mathrm{EGFN}^u_{r,dn}$ denote a d-branch EGFN, where $u = \pm 1$ indicates encryption $(+1)$ or decryption (-1). The state after r rounds is represented by a matrix $\mathcal{M}^u_{r,dn}$. We define τ as the set of column indices corresponding to the round function symbol F in $\mathcal{M}^u_{r,dn}$, which specifically identifies the input positions of F in the r-th round.

Evaluation of r_1 and r_3. We frame the evaluation within the framework of differential cryptanalysis [BS90], which is amenable to efficient automation via constraint solvers like STP and Z3. Our goal, per Definition 5, is to verify that every input to the round functions in round r is affected by every input block from the first round. Equivalently, for any non-zero input difference in the first round, no input to a round function in round r can maintain a zero difference. Let $(x_0^0, x_1^0, \cdots, x_{d-1}^0)$ denote the input difference to the first round, and let $\{x_i^r \mid i \in \tau\}$ be the input differences to the round functions in the r-th round. Our goal is thus transformed: for any non-zero input difference $(x_0^0, x_1^0, \cdots, x_{d-1}^0)$, we must determine if there exists any x_i^r (for $i \in \tau$) that remains zero. If no such x_i^r exists, then all relevant inputs are affected, and $r_1 = r - 1$.

To describe the propagation of input differences, we employ a 2-bit encoding for each branch, where $\#b00$ represents no difference, $\#b01$ represents a known non-zero difference, and $\#b11$ represents a random difference. By treating each round function F as a basic component, any EGFN instance can be decomposed into three types of components: F-function, COPY, and XOR. The differential propagation rules for these components are specified in Properties 1 to 3. To formalize our verification goal, we introduce three additional constraints in Properties 4 to 6.

Property 1. (F-function) As each round function is instantiated with an independent URF, it can be modeled as an S-box with the following differential property: Let θ_{in} and θ_{out} denote the input and output differences, respectively. Then $\theta_{out} = \#b00$ if and only if $\theta_{in} = \#b00$; otherwise, $\theta_{out} = \#b11$.

Proof. Given an input difference $\theta_{in} = \#b00$, the inputs to the F-function are identical, making the output difference deterministic: $\theta_{out} = \#b00$. For any non-zero input difference, however, the outputs are indistinguishable from random, making their difference unpredictable. Hence, as claimed, $\theta_{out} = \#b11$. □

Property 2. (COPY) Let θ_{in} represent the input difference, and let the output differences be denoted as θ_{out1} and θ_{out2}. Then the relationship between them is $\theta_{in} = \theta_{out1} = \theta_{out2}$.

Proof. The COPY operation produces two outputs with the same value as the input; consequently, the difference for both outputs remains the same as the input difference. □

Property 3. (XOR) Let θ_{in1} and θ_{in2} be the input differences, and θ_{out} be the output difference. The XOR operation yields θ_{out} as follows:

- If one input difference is #b00, then θ_{out} equals the other.
- If both input differences are #b01, then θ_{out} is #b00 or #b01.
- In all other cases, $\theta_{out} = \#b11$.

Proof. Let (x_1, y_1) and (x_2, y_2) denote two sets of inputs to XOR, and their corresponding outputs are $z_1 = x_1 \oplus y_1$ and $z_2 = x_2 \oplus y_2$. Here, we take the second case as an example since it is a bit hard to understand. Since both input differences are #b01, we can deduce that both $\Delta_x = x_1 \oplus x_2$ and $\Delta_y = y_1 \oplus y_2$ are non-zero values which can be obtained by the adversary. If $\Delta_x = \Delta_y$, the output difference $z_1 \oplus z_2 = \Delta_1 \oplus \Delta_2 = 0$, thus, $\theta_{out} = \#b00$; if $\Delta_x \neq \Delta_y$, then $z_1 \oplus z_2 = \Delta_1 \oplus \Delta_2 \neq 0$ and is known to the adversary, thus $\theta_{out} = \#b01$. All the other cases can be deduced following a similar way, thus are omitted. □

Recall that our goal is to verify that no x_i^r (for $i \in \tau$) maintains a zero difference. This is equivalent to ensuring that the bitwise AND of all these x_i^r values is $\#b00$. We show this in Property 4. If this condition holds, it confirms that all relevant inputs are affected by the first-round input.

Property 4. For all inputs x_i^r to round functions in the r-th round (i.e. $\forall i \in \tau$), the bitwise AND over all such x_i^r must equal $\#b00$.

We impose two additional constraints on the input difference $(x_0^0, \cdots, x_{d-1}^0)$. First, the difference cannot be all-zero, as this corresponds to a plaintext pair with identical inputs, which provides no useful differential information. This restriction is formalized in Property 5. Second, since the adversary controls the input to the first round and consequently knows all input differences, no x_j^0 $(0 \leq j < d)$ can take the value $\#b11$ (denoting an unknown random difference). This constraint is specified in Property 6.

Property 5. The concatenated input difference $x_0^0 \| x_1^0 \| \cdots \| x_{d-1}^0$ must not equal $\#b00 \cdots 00$ (a string of $2d$ zeros).

Property 6. For each $0 \leq j < d$, the input difference x_j^0 must not equal $\#b11$.

Combining these properties, Algorithm 1 computes r_1 and r_3 by setting the direction flag $u = +1$ for encryption (yielding r_1) and $u = -1$ for decryption (yielding r_3).

Algorithm 1 Evaluating the parameters r_1 and r_3

Input: branch number d, querying direction u, index set τ
Output: r'

$r \leftarrow 0$, $flag \leftarrow 1$
while flag is 1 **do**
  $r \leftarrow r + 1$
  for $i = 0;\ i < d;\ i++$ **do**
    Declare the 2-bit input and output difference variables x_i^0 and x_i^r, restricting their values to $\{\#b00, \#b01, \#b11\}$.
  end for
  Model $(x_0^r, \cdots, x_{d-1}^r) = \mathrm{EGFN}_{r,dn}^u(x_0^0, \cdots, x_{d-1}^0)$ //Use Property 1, 2 and 3
  Declare 2-bit variable x
  $x \leftarrow \#b11$
  for each $i \in \tau$ **do**
    $x \leftarrow x$ & x_i^r //Bit-wise AND operation
  end for
  Assert $x = \#b00$ //Property 4
  Assert $\mathrm{concat}(x_0^0, \ldots, x_{d-1}^0) \neq \mathrm{bitvector}(\underbrace{00\cdots0}_{2d \text{ bits}})$ // Property 5
  for $i = 0;\ i < d;\ i++$ **do**
    Assert $x_i^0 \neq \#b11$ //Property 6
  end for
  Input all these equations into SMT solver and solve it.
  if SMT solver outputs "unsat" **then**
    $flag \leftarrow 0$
  end if
end while
$r' \leftarrow r - 1$
return r'

Evaluation of r_2. As established in Section 3.1, the evaluation of r_2 reduces to determining whether the matrix representing the output ciphertexts over the basis of round functions has full rank. This linear algebra computation is straightforward to implement in SageMath, which provides efficient matrix rank operations. The complete procedure is detailed in Algorithm 2.

4 Security Proof Underlying AutoEGFN

In this section, we prove the security of any EGFN in combination with AutoEGFN. This proof is an extension of the proof strategy described in [NR99].

Assume that the adversary **A** can make a finite number of encryption and decryption queries to the construction $\mathrm{EGFN}_{r,dn}$ in the real world R or the ideal world I. The adversary's goal is to distinguish whether the interaction occurs in the real world R or the ideal world I. We assume that **A** makes exactly q queries. For the i-th query **A** makes, we define the query-answer pair as (x_i, y_i), where $x_i, y_i \in \{0,1\}^{dn}$ for $1 \leq i \leq q$. The interaction process between **A** and the oracle is recorded as the transcript $\theta = \{(x_1, y_1), \cdots, (x_q, y_q)\}$.

Assume that **A** does not make repetitive queries. For the transcript θ, we define two sets to represent the types of queries:

$$Q_e = \{i \mid \text{the } i\text{-th query is an encryption query}\},$$

Algorithm 2 Evaluating the parameter r_2

Input: branch number d
Output: r

1: $r \leftarrow 0$; $rk \leftarrow 0$
2: **while** $rk \neq d$ **do**
3: $r \leftarrow r + 1$; $\mathcal{F} \leftarrow \emptyset$
4: $(x_0^0, \cdots, x_{d-1}^0) \leftarrow \mathrm{SR.var}('x_0^0 \cdots x_{d-1}^0{}')$
5: **for** $i = 0$; $i < r$; $i{+}{+}$ **do**
6: $(f_0^i, \cdots, f_{d-1}^i) \leftarrow \mathrm{SR.var}('f_0^i \cdots f_{d-1}^i{}')$
7: $\mathcal{F} \leftarrow \mathcal{F} \bigcup (f_0^i, \cdots, f_{d-1}^i)$
8: **end for**
9: $(x_0^r, \cdots, x_{d-1}^r) \leftarrow \mathrm{EGFN}_{r,dn}(x_0^0, \cdots, x_{d-1}^0)$
10: **for** $j = 0$; $j < d$; $j{+}{+}$ **do**
11: $\overrightarrow{m_j} \leftarrow$ vector representing the coefficients of x_j^r with respect to basis $\mathcal{F}$
12: **end for**
13: $M \leftarrow \begin{bmatrix} \overrightarrow{m_0} \\ \overrightarrow{m_1} \\ \vdots \\ \overrightarrow{m_{d-1}} \end{bmatrix}$
14: $rk \leftarrow rank(M)$
15: **end while**
16: **return** r

$$Q_d = \{i \mid \text{the } i\text{-th query is a decryption query}\}.$$

A transcript $\theta = \{(x_1, y_1), \ldots, (x_q, y_q)\}$ is called *possible* if there exists a permutation P on $\{0,1\}^{dn}$ such that $y_i = P(x_i)$ for every $i \in Q_e$ and $x_i = P^{-1}(y_i)$ for every $i \in Q_d$. θ is *inconsistent* if for some $1 \leq j < i \leq q$ the corresponding transcript satisfies $x_i = x_j$ but $y_i \neq y_j$, or $y_i = y_j$ but $x_i \neq x_j$. Otherwise, θ is *consistent*.

Similarly to [NR99], we consider yet another distribution of the answers to $\mathbf{A}$'s queries. In a random world $\widetilde{R}$, the answer to $\mathbf{A}$'s i-th query is determined as follows:

1. If $\mathbf{A}$'s query is in Q_e and there exists a $j < i$ such that $x_i = x_j$, then the answer is the corresponding previous output y_j.
2. If $\mathbf{A}$'s query is in Q_d and there exists a $j < i$ such that $y_i = y_j$, then the answer is the corresponding previous input x_j.
3. If neither 1 nor 2 holds, then $\widetilde{R}$'s answer is a uniformly chosen dn-bit random string.

We define θ_R, $\theta_{\widetilde{R}}$, and θ_I as the transcripts of $\mathbf{A}$'s interaction with world R, $\widetilde{R}$, and I, respectively.

Proposition 1. *Assume that the adversary* $\mathbf{A}$ *can make encryption and decryption queries. Let* $I \xleftarrow{\$} \mathsf{Perm}\left(\{0,1\}^{dn}\right)$ *be a random permutation. Then, for any* $\mathbf{A}$ *that makes at most* q *queries, we have:*

$$\left| \Pr[\mathbf{A}^{\widetilde{R}, \widetilde{R}^{-1}} \Rightarrow 1] - \Pr[\mathbf{A}^{I, I^{-1}} \Rightarrow 1] \right| \leq \frac{q^2}{2^{dn}}.$$

Proof. For any possible and consistent transcript $\sigma = \{(x_1, y_1), \cdots, (x_q, y_q)\}$, we have:

$$\Pr[\theta_I = \sigma] = \frac{1}{\frac{(2^{dn})!}{(2^{dn}-q)!}} = \Pr[\theta_{\widetilde{R}} = \sigma \mid \theta_{\widetilde{R}} \text{ is consistent}].$$

Therefore, under the conditions of $\theta_{\widetilde{R}}$ being consistent, the distribution of $\theta_{\widetilde{R}}$ and θ_I are identical. Furthermore, the probability that $\theta_{\widetilde{R}}$ is inconsistent is bounded by:

$$\Pr[\theta_{\widetilde{R}} \text{ is inconsistent}] \leq \binom{q}{2} \cdot 2 \cdot \frac{1}{2^{dn}} = \frac{q(q-1)}{2^{dn}} \leq \frac{q^2}{2^{dn}}.$$

The proposition follows:

$$\begin{aligned}
&\left| \Pr[\mathbf{A}^{\widetilde{R},\widetilde{R}^{-1}} \Rightarrow 1] - \Pr[\mathbf{A}^{I,I^{-1}} \Rightarrow 1] \right| \\
&\quad \leq \left| \Pr[\mathbf{A}^{\widetilde{R},\widetilde{R}^{-1}} \Rightarrow 1 \mid \theta_{\widetilde{R}} \text{ is consistent}] - \Pr[\mathbf{A}^{I,I^{-1}} \Rightarrow 1] \right| \cdot \Pr[\theta_{\widetilde{R}} \text{ is consistent}] \\
&\quad + \left| \Pr[\mathbf{A}^{\widetilde{R},\widetilde{R}^{-1}} \Rightarrow 1 \mid \theta_{\widetilde{R}} \text{ is inconsistent}] - \Pr[\mathbf{A}^{I,I^{-1}} \Rightarrow 1] \right| \cdot \Pr[\theta_{\widetilde{R}} \text{ is inconsistent}] \\
&\quad \leq \Pr[\theta_{\widetilde{R}} \text{ is inconsistent}] \cdot 1 + \Pr[\theta_{\widetilde{R}} \text{ is consistent}] \cdot 0 \\
&\quad \leq \frac{q^2}{2^{dn}}.
\end{aligned}$$

□

Next, we introduce some bad events that may occur during the interaction between the adversary and its oracles. We then prove that for any specific choice of keyed ϵ-AU$_\tau$ functions H_K over $\{0,1\}^{dn}$, the probability that these bad events occur is negligible, where $K \xleftarrow{\$} \{0,1\}^n$ is an independent key. In the following, we omit the key K for convenience. The specific definition of the key is implicitly contained in the instantiation of H.

Definition 7 (Bad events)**.** Let $H \in \mathsf{Func}\left(\{0,1\}^{dn}, \{0,1\}^{dn}\right)$ be a ϵ-AU$_\tau$ function for a fixed set of indices τ. We define $\mathsf{Bad}(H)$ as the set of all possible and consistent transcripts θ for which there exist distinct indices $\exists\ 1 \leq i < j \leq q$ such that $\exists k \in \tau, H(x_i)_{[k]} = H(x_j)_{[k]}$.

Proposition 2. *Let $H_1, H_2^{-1} \in \mathsf{Func}\left(\{0,1\}^{dn}, \{0,1\}^{dn}\right)$ be keyed ϵ_1-AU$_{\tau_1}$ and ϵ_2-AU$_{\tau_2}$ functions, respectively. Then for any possible and consistent transcript σ, we have:*

$$\Pr_{H_1, H_2^{-1}}\left[\sigma \in \mathsf{Bad}(H_1) \bigcup \mathsf{Bad}(H_2^{-1})\right] \leq \binom{q}{2} \cdot (|\tau_1|\epsilon_1 + |\tau_2|\epsilon_2) \leq \frac{q^2(|\tau_1|\epsilon_1 + |\tau_2|\epsilon_2)}{2}.$$

Proof. According to Definition 7, define $\sigma = \{(x_1, y_1), \cdots, (x_q, y_q)\}$, $\sigma \in \mathsf{Bad}(H_1) \bigcup \mathsf{Bad}(H_2^{-1})$ if and only if there exist $1 \leq i < j \leq q$ such that either:

- There exists $k_1 \in \tau_1$ with $H_1(x_i)_{[k_1]} = H_1(x_j)_{[k_1]}$, or
- There exists $k_2 \in \tau_2$ with $H_2^{-1}(y_i)_{[k_2]} = H_2^{-1}(y_j)_{[k_2]}$.

For any fixed pair (i, j) with $i < j$, by the union bound and the ϵ-AU property:

$$\Pr\left[\exists k_1 \in \tau_1 : H_1(x_i)_{[k_1]} = H_1(x_j)_{[k_1]}\right] \leq \sum_{k_1 \in \tau_1} \Pr\left[H_1(x_i)_{[k_1]} = H_1(x_j)_{[k_1]}\right] \leq |\tau_1| \cdot \epsilon_1.$$

Similarly,

$$\Pr\left[\exists k_2 \in \tau_2 : H_2^{-1}(y_i)_{[k_2]} = H_2^{-1}(y_j)_{[k_2]}\right] \leq \sum_{k_2 \in \tau_2} \Pr\left[H_2^{-1}(y_i)_{[k_2]} = H_2^{-1}(y_j)_{[k_2]}\right] \leq |\tau_2| \cdot \epsilon_2.$$

By the union bound, the probability that either event occurs for the pair (i, j) is at most $|\tau_1|\epsilon_1 + |\tau_2|\epsilon_2$. Applying the union bound again over all $\binom{q}{2}$ pairs (i, j) gives the desired result. □

Lemma 1. *Let* $\mathrm{EGFN}_{r,dn}$ *represent an* r*-round EGFN structure with* d *branches* ($d \geq 2$)*, where each branch is* n *bits and all round functions are instantiated with independent URFs. Consider the composite structure* $R = H_2 \circ \mathrm{EGFN}_{r_2,dn} \circ H_1$*, where* $\mathrm{EGFN}_{r_2,dn}$ *is an* r_2*-round EGFN structure with associated matrix* $\mathcal{M}$*,* H_1 *is an* ϵ_1*-*AU_{τ_1} *function,* H_2^{-1} *is an* ϵ_2*-*AU_{τ_2} *function, and* τ_1 *and* τ_2 *are the index sets of the columns containing the symbol* F *in* $\mathcal{M}$ *and* $\mathcal{M}^{-1}$*, respectively. The parameter* r_2 *is computed by Algorithm 2 when AutoEGFN is applied to* $\mathrm{EGFN}_{r,dn}$*.*

Let $\sigma = \{(x_1, y_1), \ldots, (x_q, y_q)\}$ *be any possible and consistent transcript obtained from* $\mathbf{A}$*'s queries. We consider the case where* $\sigma \notin \mathsf{Bad}(H_1) \cup \mathsf{Bad}(H_2^{-1})$*. Then we have:*

$$\Pr\left[\theta_R = \sigma \mid \sigma \notin \mathsf{Bad}(H_1) \bigcup \mathsf{Bad}(H_2^{-1})\right] = \Pr[\theta_{\widetilde{R}} = \sigma].$$

Proof. We first analyze the probability in the $\widetilde{R}$ world. Since σ is a possible transcript, $\theta_{\widetilde{R}} = \sigma$ if and only if for all $1 \leq i \leq q$, the i-th answer from $\widetilde{R}$ is y_i when $(x_i, y_i) \in Q_e$ (encryption query), and is x_i when $(x_i, y_i) \in Q_d$ (decryption query).

Since σ is consistent and $\mathbf{A}$ makes no repeated queries, for each query i, the input x_i (for encryption queries) or output y_i (for decryption queries) does not match any previous query. Therefore, by the definition of $\widetilde{R}$, each answer is an independent uniformly random dn-bit string. Thus, we have:

$$\Pr[\theta_{\widetilde{R}} = \sigma] = 2^{-dnq}. \tag{2}$$

Now consider the real structure $R = H_2 \circ \mathrm{EGFN}_{r_2,dn} \circ H_1$. Since σ is a possible transcript, $\theta_R = \sigma$ if and only if for all $1 \leq i \leq q$, we have $y_i = R(x_i)$ for encryption queries and $x_i = R^{-1}(y_i)$ for decryption queries.

Fix H_1 and H_2 such that $\sigma \notin \mathsf{Bad}(H_1) \cup \mathsf{Bad}(H_2^{-1})$. Let $h_i^0 = H_1(x_i)$ and $h_i^1 = H_2^{-1}(y_i)$. By the definition of R, for encryption queries we have:

$$y_i = R(x_i) = H_2(\mathrm{EGFN}_{r_2,dn}(H_1(x_i))) \Longleftrightarrow \mathrm{EGFN}_{r_2,dn}(h_i^0) = H_2^{-1}(y_i) = h_i^1,$$

and for decryption queries:

$$x_i = R^{-1}(y_i) = H_1^{-1}(\mathrm{EGFN}_{r_2,dn}^{-1}(H_2^{-1}(y_i))) \Longleftrightarrow \mathrm{EGFN}_{r_2,dn}^{-1}(h_i^1) = h_i^0.$$

Since $\sigma \notin \mathsf{Bad}(H_1) \cup \mathsf{Bad}(H_2^{-1})$, for any $i \neq j$:

- There exists some $k_1 \in \tau_1$ such that $H_1(x_i)_{[k_1]} \neq H_1(x_j)_{[k_1]}$,
- There exists some $k_2 \in \tau_2$ such that $H_2^{-1}(y_i)_{[k_2]} \neq H_2^{-1}(y_j)_{[k_2]}$.

This means the sequences $\{h_i^0\}_{i=1}^q$ and $\{h_i^1\}_{i=1}^q$ have sufficient distinction in the critical positions specified by τ_1 and τ_2. By Definition 6 and the properties of $\mathrm{EGFN}_{r_2,dn}$ (ensured by Algorithm 2), under such conditions both $\mathrm{EGFN}_{r_2,dn}$ and $\mathrm{EGFN}_{r_2,dn}^{-1}$ behave like random functions.

Therefore, for each i, the probability that $\mathrm{EGFN}_{r_2,dn}(h_i^0) = h_i^1$ (for encryption queries) or $\mathrm{EGFN}_{r_2,dn}^{-1}(h_i^1) = h_i^0$ (for decryption queries) is exactly 2^{-dn}. Since the q queries are independent, we have:

$$\Pr\left[\theta_R = \sigma \mid \sigma \notin \mathsf{Bad}(H_1) \bigcup \mathsf{Bad}(H_2^{-1})\right] = 2^{-dnq}. \tag{3}$$

Comparing equations (2) and (3) completes the proof. □

Next, in the following Theorem 1, we first prove that for any d-branch $\mathrm{EGFN}_{r_2,dn}$ combined with an AU function H_1 in the head, it is PRP secure, and for $\mathrm{EGFN}_{r_2,dn}$ combined with H_1 and H_2 in the head and tail, it is SPRP secure.

Theorem 1 (Composed Structure)**.** *Let* $\mathrm{EGFN}_{r,dn}$ *represent an* r*-round EGFN structure with its associated matrix* $\mathcal{M}$*, where each branch is* n *bits and all round functions are instantiated with independent URFs. Let* H_1 *and* H_2^{-1} *be keyed* ϵ_1*-*AU_{τ_1} *and* ϵ_2*-*AU_{τ_2} *functions in* $\mathsf{Func}\left(\{0,1\}^{dn},\{0,1\}^{dn}\right)$*, respectively, where* τ_1 *and* τ_2 *are the index sets of the columns of the symbol* F *in* $\mathcal{M}$ *and* $\mathcal{M}^{-1}$*. Then for any adversary* $\mathbf{A}$ *that runs in time at most* t *and makes at most* q *queries, we have:*

$$\mathbf{Adv}^{\mathrm{PRP}}_{\mathrm{EGFN}_{r_2,dn}\circ H_1}(q,t) \le q^2\cdot|\tau_1|\epsilon_1 + \frac{q^2}{2^{dn+1}}, \tag{4a}$$

$$\mathbf{Adv}^{\mathrm{SPRP}}_{H_2\circ\mathrm{EGFN}_{r_2,dn}\circ H_1}(q,t) \le q^2(|\tau_1|\epsilon_1 + |\tau_2|\epsilon_2) + \frac{q^2}{2^{dn}}, \tag{4b}$$

where r_2 *is the parameter calculated by Algorithm 2 when AutoEGFN is applied to* $\mathrm{EGFN}_{r,dn}$*.*

Proof. When $\mathbf{A}$ makes encryption queries, this theorem uses the fact that after processing the inputs by the function H_1, the inputs of $\mathrm{EGFN}_{r_2,dn}$ are sufficiently distinct. Therefore, the output of $\mathrm{EGFN}_{r_2,dn}$ can be regarded as a random string, and collisions are difficult to occur. The situation is similar as well when $\mathbf{A}$ makes decryption queries.

Let Γ be the set of all possible and consistent transcripts. Let $\mathsf{Bad} = \mathsf{Bad}(H_1)\cup\mathsf{Bad}(H_2^{-1})$, where $\mathsf{Bad}(H_1)$ and $\mathsf{Bad}(H_2^{-1})$ are defined as in Definition 7. By the H-coefficient technique, we have:

$$\begin{aligned}
&\left|\Pr[\mathbf{A}^{R,R^{-1}}\Rightarrow 1] - \Pr[\mathbf{A}^{\widetilde{R},\widetilde{R}^{-1}}\Rightarrow 1]\right| \\
&\quad\le \left|\sum_{\sigma\in\Gamma}\Big(\Pr[\theta_R=\sigma] - \Pr[\theta_{\widetilde{R}}=\sigma]\Big)\right| + \Pr_{\widetilde{R}}[\theta_{\widetilde{R}}\text{ is inconsistent}] \\
&\quad\le \sum_{\sigma\in\Gamma}\left|\Pr\Big[\theta_R=\sigma|\sigma\notin\mathsf{Bad}\Big] - \Pr[\theta_{\widetilde{R}}=\sigma]\right|\cdot\Pr[\sigma\notin\mathsf{Bad}] && (5a)\\
&\quad+ \sum_{\sigma\in\Gamma}\left|\Pr\Big[\theta_R=\sigma|\sigma\in\mathsf{Bad}\Big] - \Pr[\theta_{\widetilde{R}}=\sigma]\right|\cdot\Pr[\sigma\in\mathsf{Bad}] && (5b)\\
&\quad+ \Pr[\theta_{\widetilde{R}}\text{ is inconsistent}]. && (5c)
\end{aligned}$$

We proved that the value of (5c) is smaller than $q^2/2^{dn}$ in Proposition 1, and the value of (5a) is zero by Lemma 1. To get the upper bound of (5b), w.l.o.g., we use the triangle inequality and the fact that probabilities are at most 1:

$$\begin{aligned}
&\sum_{\sigma\in\Gamma}\left|\Pr[\theta_R=\sigma\mid\sigma\in\mathsf{Bad}] - \Pr[\theta_{\widetilde{R}}=\sigma]\right|\cdot\Pr[\sigma\in\mathsf{Bad}] \\
&\quad\le \sum_{\sigma\in\Gamma}\Pr[\sigma\in\mathsf{Bad}] + \sum_{\sigma\in\Gamma}\Pr[\theta_{\widetilde{R}}=\sigma]\cdot\Pr[\sigma\in\mathsf{Bad}] \\
&\quad\le \max_{\sigma\in\Gamma}\Pr[\sigma\in\mathsf{Bad}] + \max_{\sigma\in\Gamma}\Pr[\sigma\in\mathsf{Bad}] = 2\cdot\max_{\sigma\in\Gamma}\Pr[\sigma\in\mathsf{Bad}].
\end{aligned}$$

Then using Proposition 2, we can find (5b) is smaller than

$$2\cdot\max_{\sigma\in\Gamma}\Pr\Big[\sigma\in\mathsf{Bad}\Big] \le q^2(|\tau_1|\epsilon_1 + |\tau_2|\epsilon_2).$$

Combining all bounds, we have:

$$\left|\Pr[\mathbf{A}^{R,R^{-1}}\Rightarrow 1] - \Pr[\mathbf{A}^{\widetilde{R},\widetilde{R}^{-1}}\Rightarrow 1]\right| \le q^2(|\tau_1|\epsilon_1 + |\tau_2|\epsilon_2) + \frac{q^2}{2^{dn}}.$$

This completes the proof of Inequation (4b). The proof for Inequation (4a) is similar. □

Similar to [MI08, Theorem 1], we prove in Lemma 2 and Lemma 3 that both $\mathrm{EGFN}_{r_1,dn}$ and $\mathrm{EGFN}^{-1}_{r_3,dn}$ are AU functions. Thereafter, we can naturally conclude Theorem 2 by applying Theorem 1.

Lemma 2. *Consider the function* $\mathrm{EGFN}_{r_1,dn}$ *with its associated matrix* $\mathcal{M}$*, where* r_1 *is the parameter computed by Algorithm 1 when AutoEGFN is applied to* $\mathrm{EGFN}_{r,dn}$*. Let* τ *and* τ' *be the index sets of the columns and rows of symbol* F *in* $\mathcal{M}$*, respectively. Then* $\mathrm{EGFN}_{r_1,dn}$ *is a* $\frac{|\tau|\cdot r_1}{2^n}$*-*$AU_\tau$ *function.*

Proof. Assume that $(y_0^r, \cdots, y_{d-1}^r) = \mathrm{EGFN}_{r,dn}(x_0^0, \cdots, x_{d-1}^0)$. In the matrix $\mathcal{M}$, the column index set τ represents the index set of the input elements of the symbol F, that is, the input sub-block of F can be written as x_j^0, where $j \in \tau$; the row index set τ' represents the index set of the elements influenced by the output value of the symbol F.

To prove that $\mathrm{EGFN}_{r_1,dn}$ is an AU function, we need to show that for any two different messages $\vec{x} = (x_0^0, \cdots, x_{d-1}^0)$ and $\vec{x}' = (x_0'^0, \cdots, x_{d-1}'^0)$, the probability of collision in the output blocks $y_j^r, j \in \tau$ of $\mathrm{EGFN}_{r_1,dn}$ is small. W.l.o.g., we assume $x_i^0 \neq x_i'^0, i \in \tau'$, then we estimate the following probability:

$$\Pr[\mathrm{EGFN}_{r_1,dn}(\vec{x})_{[j]} = \mathrm{EGFN}_{r_1,dn}(\vec{x}')_{[j]}], \forall j \in \tau,$$

where the parameter r_1 is determined by Algorithm 1.

We study the diffusion property from a graph perspective by defining an associated directed graph with vertex set $\{0, \cdots, d-1\}$, where (i, j) is an edge if the output y_j^r depends on the input x_i^0 (directly or through a round function). In other words, the notion that block x_i^0 affects block $y_j^{r_1+1}$ means that there exists a path of exactly length $r_1 + 1$ from i to j.

By Definition 5 and the properties of Algorithm 1, for any input block position $p \in \{0, \cdots, d-1\}$ and any output block position $q \in \tau$, there exists an appropriate path of length r_1 in the graph of $\mathrm{EGFN}_{r_1,dn}$ from some starting vertex i to ending vertex q, where $q \in \tau$. This ensures that every input block can potentially affect every output block in τ.

For each $q \in \tau$, we fix an appropriate path from some input block to q. For $k = 1, \cdots, r_1$, we define two sequences of internal states $Z_k = \mathrm{EGFN}_{k,dn}(\vec{x})_{[s(k)]}$ and $Z'_k = \mathrm{EGFN}_{k,dn}(\vec{x}')_{[s(k)]}$ that follow the appropriate path, where $s(k)$ is a function that indicates the position on the appropriate diffusion path.

For $k = 1$, the condition $Z_1 = Z'_1$ is equivalent to an equation of the form: $F(x_i^0) \oplus x_j^0 = F(x_i'^0) \oplus x_j'^0$. When F is a URF, we have:

$$\Pr[Z_1 = Z'_1] = \Pr[F(x_i^0) \oplus x_j^0 = F(x_i'^0) \oplus x_j'^0] \leq \frac{1}{2^n}, j \in \tau', i \in \tau.$$

Furthermore, due to the independence among different round functions, we have $\Pr[Z_k = Z'_k \mid Z_{k-1} \neq Z'_{k-1}] \leq \frac{1}{2^n}, k = 2, \cdots, r_1$. Therefore, the probability $\Pr[Z_{r_1} = Z'_{r_1}]$ is bounded as follows:

$$\begin{aligned}
\Pr[Z_{r_1} = Z'_{r_1}] &\leq \Pr[Z_1 = Z'_1] + \Pr[Z_{r_1} = Z'_{r_1} \mid Z_1 \neq Z'_1] \\
&\leq \frac{1}{2^n} + \Pr[Z_{r_1} = Z'_{r_1} \mid Z_1 \neq Z'_1] \\
&\leq \frac{1}{2^n} + \frac{1}{2^n} + \Pr[Z_{r_1} = Z'_{r_1} \mid Z_2 \neq Z'_2 \wedge Z_1 \neq Z'_1] \\
&\;\;\vdots \\
&\leq \frac{r_1}{2^n}.
\end{aligned}$$

This shows that for any fixed $q \in \tau$:

$$\Pr[\mathrm{EGFN}_{r_1,dn}(\vec{x})_{[q]} = \mathrm{EGFN}_{r_1,dn}(\vec{x}')_{[q]}] \leq \frac{r_1}{2^n}.$$

Since this holds for each $q \in \tau$ independently, we have:

$$\Pr[\exists q \in \tau : \mathrm{EGFN}_{r_1,dn}(\vec{x})_{[q]} = \mathrm{EGFN}_{r_1,dn}(\vec{x}')_{[q]}] \leq \frac{|\tau| \cdot r_1}{2^n}.$$

Therefore, $\mathrm{EGFN}_{r_1,dn}$ is a $\frac{|\tau| \cdot r_1}{2^n}$-AU function over the output blocks indexed by τ. □

Lemma 3. *When making the query to* $\mathrm{EGFN}^{-1}_{r_3,dn}$ *with its associated matrix* $\mathcal{M}$*, where* r_3 *is the parameter calculated by Algorithm 1 when AutoEGFN is applied to* $\mathrm{EGFN}^{-1}_{r,dn}$*. Define* τ *and* τ' *as the index sets of the columns and rows of the symbol* F *in* $\mathcal{M}$*, respectively. Then,* $\mathrm{EGFN}^{-1}_{r_3,dn}$ *is a* $\frac{|\tau| \cdot r_3}{2^n}$*-*$AU_\tau$ *function.*

Proof. The proof of Lemma 3 is similar to that of Lemma 2, so it is omitted. □

In fact, Lemma 2 and Lemma 3 demonstrate that both $\mathrm{EGFN}_{r_1,dn}$ and $\mathrm{EGFN}^{-1}_{r_3,dn}$ construct a function with the AU property. When the inputs are different, the probability of collisions at the output is negligible. Finally, we prove that $\mathrm{EGFN}_{r_1+r_2,dn}$ and $\mathrm{EGFN}_{r_1+r_2+r_3,dn}$ are PRP/SPRP secure in the following Theorem 2, respectively.

Theorem 2. *Let* $\mathrm{EGFN}_{r,dn}$ *represent an* r*-round EGFN structure with its associated matrix* $\mathcal{M}$*, where each branch is* n *bits and all round functions are instantiated with independent URFs. Let* τ_1 *and* τ_2 *be the index sets of the columns of the symbol* F *in* $\mathcal{M}$ *and* $\mathcal{M}^{-1}$*, respectively. Then for any adversary* $\mathbf{A}$ *that runs in time at most* t *and makes at most* q *queries, we have:*

$$\mathbf{Adv}^{\mathrm{PRP}}_{\mathrm{EGFN}_{r_1+r_2,dn}}(q,t) \leq \frac{q^2 \cdot |\tau_1|^2 r_1}{2^n} + \frac{q^2}{2^{dn+1}},$$

$$\mathbf{Adv}^{\mathrm{SPRP}}_{\mathrm{EGFN}_{r_1+r_2+r_3,dn}}(q,t) \leq \frac{q^2 \cdot (|\tau_1|^2 r_1 + |\tau_2|^2 r_3)}{2^n} + \frac{q^2}{2^{dn}},$$

where r_1*,* r_2*, and* r_3 *are parameters calculated by Algorithm 1 and 2 when AutoEGFN is applied to* $\mathrm{EGFN}^{\pm 1}_{r,dn}$*.*

Proof. The proof of this theorem follows straightforwardly from Theorem 1, Lemma 2, and Lemma 3, and is omitted. □

5 Applications on Multiple EGFN Instances

This section presents the results of AutoEGFN applied to various EGFN structures. Due to space limitations, we present results of Type-1 and Type-2 GFNs in Section 5.1 and show the others in Appendix A. In Section 5.2, we compare the efficiency of AutoEGFN using different SMT solvers, demonstrating the tool's practicality for analyzing structures with large branch numbers, a task that is prohibitively complex for manual analysis.

5.1 Type-1 and Type-2

We begin by analyzing the PRP and SPRP security of Type-1 and Type-2 GFNs with branch numbers ranging from 4 to 32. Figure 3 illustrates the 4-branch variants of these two structures.

At CRYPTO 1989, Zheng et al. [ZMI89] proved that $(2d-1)$ rounds of Type-1 and $(d+1)$ rounds of Type-2 are PRP secure. Thereafter, Moriai and Vaudenay [MV00] proved that the $(d+1)$-round Type-2 is PRP secure, and the $(2d)$-round Type-2 is SPRP secure. Later, Hoang and Rogaway [HR10b] used a coupling-based method to prove that both $(2t(d^2+d))$-round Type-1 and $(t(2d+2))$-round Type-2 are SPRP secure. Shen et al. [SGW20] improved the coupling analyses of [HR10b] and proved the SPRP security

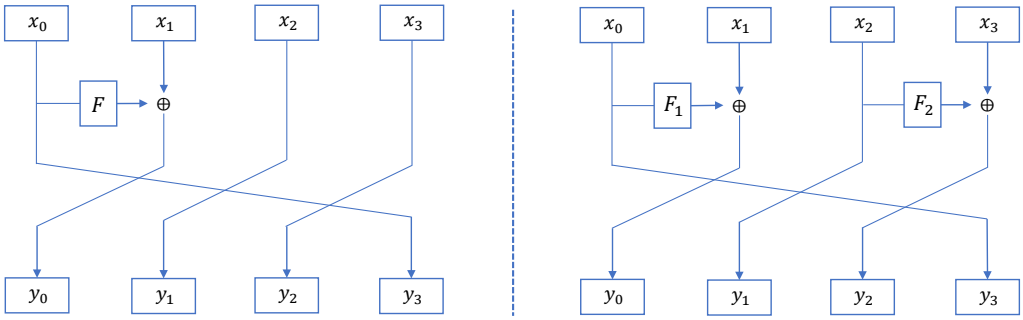


Figure 3: Encryption process of Type-1 (Left) and Type-2 (Right) structures with 4 branches.

for $((d^2+d-2)t+1)$-round Type-1 and $(2dt+1)$-round Type-2, where t is a parameter corresponding to different security bounds.

It is noteworthy that due to the complexity of these structures with a large number of branches, manual analysis is extremely challenging. To address this, we use AutoEGFN to evaluate Type-1 and Type-2 GFNs with different branch numbers; the results are summarized in Table 3.

Table 3: Computer-aided proof results for Type-1/2 structures with different branch numbers. Parameters $r1/r2/r3$ are defined in Section 3. The (S)PRP column shows the number of rounds required to achieve the corresponding security notion, calculated as r_1+r_2 for PRP and $r_1+r_2+r_3$ for SPRP.

Structure	**Branch number**	**Parameters**			**our # of rounds**	
		r_1	r_2	r_3	PRP	SPRP
Type-1	4	3	4	9	7	16
	6	5	6	25	11	36
	8	7	8	49	15	64
	10	9	10	81	19	100
	12	11	12	121	23	144
	14	13	14	169	27	196
	16	15	16	225	31	256
	32	31	32	961	63	1024
Type-2	4	3	2	3	5	8
	6	5	2	5	7	12
	8	7	2	7	9	16
	10	9	2	9	11	20
	12	11	2	11	13	24
	14	13	2	13	15	28
	16	15	2	15	17	32
	32	31	2	31	33	64

From Table 3, we can observe that the SPRP security rounds of Type-1 follow a very regular pattern. Therefore, we can naturally make the following conjectures.

Conjecture 1. *Let* $\mathrm{Type1}_{r,dn}$ *be the* r*-round,* dn*-bit Type-1 GFN with even branch number* d *and round functions being* n*-bit URFs. Then for any adversary making* q *queries, the*

SPRP advantage of $\mathrm{Type1}_{d^2,dn}$ *satisfies:*

$$\mathbf{Adv}^{\mathrm{SPRP}}_{\mathrm{Type1}_{d^2,dn}}(q,t) \leq \frac{q^2 \cdot d(d-1)}{2^n} + \frac{q^2}{2^{dn}}.$$

5.2 Efficiency of AutoEGFN

Since the efficiency bottleneck of AutoEGFN lies in Algorithm 1, we compare the running time of this algorithm using STP [GHS+14] and Z3 [Mic25] solvers in Table 4. The experiments were conducted on a WSL2 (Ubuntu 22.04.5 LTS) platform with an Intel Core i7-12700 CPU in single-thread mode. Table 4 presents representative results of the runtime comparison between the two solvers.

Table 4: Running time of AutoEGFN for calculating r_1 and r_3 with STP and Z3 solvers.

Structure	**Branch number**	**STP**		**Z3**	
		r_1	r_3	r_1	r_3
Type-1	16	0.13 s	11.62 s	0.19 s	13.07 s
	32	0.48 s	356.14 s	0.70 s	437.96 s
Type-2	16	0.34 s	0.33 s	0.35 s	0.35 s
	32	2.23 s	2.23 s	2.98 s	2.52 s
YI11's Type-1	15	2.93 s	2.77 s	3.61 s	3.52 s
	16	4.29 s	3.25 s	5.22 s	4.16 s
Type-1.x (8, 1)	8	0.35 s	0.18 s	0.48 s	0.27 s
Type-1.x (9, 2)	9	0.13 s	0.13 s	0.17 s	0.17 s
Nyberg's GFN	16	0.33 s	0.32 s	0.35 s	0.37 s
	32	2.13 s	2.25 s	1.63 s	1.40 s
SM's GFN	14	0.10 s	0.10 s	0.12 s	0.13 s
	16	0.12 s	0.11 s	0.13 s	0.13 s
BMT's EGFN	16	0.11 s	0.11 s	0.10 s	0.11 s
	32	0.21 s	0.21 s	0.20 s	0.23 s

The results demonstrate the high efficiency of our tool. The analysis completes within seconds for structures with a small number of branches or rapid diffusion, while even for more complex instances like computing r_3 for a 32-branch Type-1 structure, the runtime is under ten minutes. This represents a substantial improvement over the time-consuming process of manual proof.

6 Conclusions

In this paper, we introduce AutoEGFN, a computer-aided proof tool designed for EGFNs. By calculating three parameters, r_1, r_2, and r_3, we can determine the number of rounds to achieve the PRP and SPRP security for various instances of EGFNs. To compute r_1 and r_3 efficiently, we consider this process from the view of differential cryptanalysis. By modeling it as differential propagation and adding suitable restrictions, one can finally reach r_1 and r_3 with the help of an SMT constraint solver. The parameter r_2 is calculated using SageMath by computing the rank of the output with respect to the round function. Our tool is highly efficient, typically completing the proof process within minutes for most cases. The validity of AutoEGFN is guaranteed by a detailed security proof. Specifically, we adopted the three-layer framework proposed by Naor and Reingold to establish concrete

security bounds for the multi-branch EGFNs. We have applied our tool to a variety of EGFN structures and proved their security, which is much harder to be achieved manually. In summary, AutoEGFN substantially reduces the workload of the prover and facilitate intuitive comparisons between different structures, making the design and analysis of the structures easier.

Acknowledgments

The authors thank the anonymous ToSC reviewers for their many helpful comments. This work is supported by the National Cryptologic Science Fund of China (Grant No. 2025NCSF01013), the National Key R&D Program of China (Grant No. 2024YFA1013000), the National Natural Science Foundation of China (Grant No. U2336207, 62032014), Fundamental and Interdisciplinary Disciplines Breakthrough Plan of the Ministry of Education of China (JYB2025XDXM114). Chun Guo was supported by the National Natural Science Foundation of China (Grant No. 62372274). Muzhou Li was also supported by the National Natural Science Foundation of China (Grant No. 62502277), and the Natural Science Foundation of Shandong Province, China (Grant No. ZR2024QF218).

References

[AFH24] Miguel Ambrona, Pooya Farshim, and Patrick Harasser. Block ciphers in idealized models: Automated proofs and new security results. In Bo Luo, Xiaojing Liao, Jun Xu, Engin Kirda, and David Lie, editors, *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security, CCS 2024, Salt Lake City, UT, USA, October 14-18, 2024*, pages 2771–2785. ACM, 2024.

[AG99] Carlisle Adams and Jeff Gilchrist. The CAST-256 encryption algorithm. *RFC*, 2612:1–19, 1999.

[AGP+19] Martin R. Albrecht, Lorenzo Grassi, Léo Perrin, Sebastian Ramacher, Christian Rechberger, Dragos Rotaru, Arnab Roy, and Markus Schofnegger. Feistel structures for mpc, and more. In Kazue Sako, Steve A. Schneider, and Peter Y. A. Ryan, editors, *Computer Security - ESORICS 2019 - 24th European Symposium on Research in Computer Security, Luxembourg, September 23-27, 2019, Proceedings, Part II*, volume 11736 of *Lecture Notes in Computer Science*, pages 151–171. Springer, 2019.

[BDJR97] Mihir Bellare, Anand Desai, E. Jokipii, and Phillip Rogaway. A concrete security treatment of symmetric encryption. In *38th Annual Symposium on Foundations of Computer Science, FOCS 1997, Miami Beach, Florida, USA, October 19-22, 1997*, pages 394–403. IEEE Computer Society, 1997.

[BFMT16] Thierry P. Berger, Julien Francq, Marine Minier, and Gaël Thomas. Extended generalized feistel networks using matrix representation to propose a new lightweight block cipher: Lilliput. *IEEE Trans. Computers*, 65(7):2074–2089, 2016.

[BG90] Mihir Bellare and Shafi Goldwasser. New paradigms for digital signatures and message authentication based on non-interactive zero knowledge proofs. In Gilles Brassard, editor, *Advances in Cryptology — CRYPTO' 89 Proceedings*, pages 194–211, New York, NY, 1990. Springer New York.

[BMT13] Thierry P. Berger, Marine Minier, and Gaël Thomas. Extended generalized feistel networks using matrix representation. In Tanja Lange, Kristin E. Lauter, and Petr Lisonek, editors, *Selected Areas in Cryptography - SAC 2013 - 20th International Conference, Burnaby, BC, Canada, August 14-16, 2013, Revised Selected Papers*, volume 8282 of *Lecture Notes in Computer Science*, pages 289–305. Springer, 2013.

[BR93] Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In Dorothy E. Denning, Raymond Pyle, Ravi Ganesan, Ravi S. Sandhu, and Victoria Ashby, editors, *CCS '93, Proceedings of the 1st ACM Conference on Computer and Communications Security, Fairfax, Virginia, USA, November 3-5, 1993*, pages 62–73. ACM, 1993.

[BR06] Mihir Bellare and Phillip Rogaway. The security of triple encryption and a framework for code-based game-playing proofs. In Serge Vaudenay, editor, *Advances in Cryptology - EUROCRYPT 2006, 25th Annual International Conference on the Theory and Applications of Cryptographic Techniques, St. Petersburg, Russia, May 28 - June 1, 2006, Proceedings*, volume 4004 of *Lecture Notes in Computer Science*, pages 409–426. Springer, 2006.

[BS90] Eli Biham and Adi Shamir. Differential cryptanalysis of des-like cryptosystems. In Alfred Menezes and Scott A. Vanstone, editors, *Advances in Cryptology - CRYPTO '90, 10th Annual International Cryptology Conference, Santa Barbara, California, USA, August 11-15, 1990, Proceedings*, volume 537 of *Lecture Notes in Computer Science*, pages 2–21. Springer, 1990.

[DDGP22] Stéphanie Delaune, Patrick Derbez, Arthur Gontier, and Charles Prud'homme. New algorithm for exhausting optimal permutations for generalized feistel networks. In Takanori Isobe and Santanu Sarkar, editors, *Progress in Cryptology - INDOCRYPT 2022 - 23rd International Conference on Cryptology in India, Kolkata, India, December 11-14, 2022, Proceedings*, volume 13774 of *Lecture Notes in Computer Science*, pages 103–124. Springer, 2022.

[DFLM19] Patrick Derbez, Pierre-Alain Fouque, Baptiste Lambin, and Victor Mollimard. Efficient search for optimal diffusion layers of generalized feistel networks. *IACR Trans. Symmetric Cryptol.*, 2019(2):218–240, 2019.

[GGM86] Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions. *J. ACM*, 33(4):792–807, 1986.

[GHS+14] Vijay Ganesh, Trevor Hansen, Mate Soos, Dan Liew, and Ryan Govostes. Stp: Simple theorem prover. https://stp.github.io/, 2014. Version 2.3.3.

[GJNS16] Jian Guo, Jérémy Jean, Ivica Nikolic, and Yu Sasaki. Meet-in-the-middle attacks on classes of contracting and expanding feistel constructions. *IACR Trans. Symmetric Cryptol.*, 2016(2):307–337, 2016.

[GM84] Shafi Goldwasser and Silvio Micali. Probabilistic encryption. *J. Comput. Syst. Sci.*, 28(2):270–299, 1984.

[HKM15] Viet Tung Hoang, Jonathan Katz, and Alex J. Malozemoff. Automated analysis and synthesis of authenticated encryption schemes. In Indrajit Ray, Ninghui Li, and Christopher Kruegel, editors, *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, Denver, CO, USA, October 12-16, 2015*, pages 84–95. ACM, 2015.

[HLK+13] Deukjo Hong, Jung-Keun Lee, Dong-Chan Kim, Daesung Kwon, Kwon Ho Ryu, and Donggeon Lee. LEA: A 128-bit block cipher for fast encryption on common processors. In Yongdae Kim, Heejo Lee, and Adrian Perrig, editors, *Information Security Applications - 14th International Workshop, WISA 2013, Jeju Island, Korea, August 19-21, 2013, Revised Selected Papers*, volume 8267 of *Lecture Notes in Computer Science*, pages 3–27. Springer, 2013.

[HR10a] Viet Tung Hoang and Phillip Rogaway. On generalized feistel networks. In Tal Rabin, editor, *Advances in Cryptology - CRYPTO 2010, 30th Annual Cryptology Conference, Santa Barbara, CA, USA, August 15-19, 2010. Proceedings*, volume 6223 of *Lecture Notes in Computer Science*, pages 613–630. Springer, 2010.

[HR10b] Viet Tung Hoang and Phillip Rogaway. On generalized feistel networks. *IACR Cryptol. ePrint Arch.*, page 301, 2010.

[IIMP19] Akiko Inoue, Tetsu Iwata, Kazuhiko Minematsu, and Bertram Poettering. Cryptanalysis of OCB2: attacks on authenticity and confidentiality. In Alexandra Boldyreva and Daniele Micciancio, editors, *Advances in Cryptology - CRYPTO 2019 - 39th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2019, Proceedings, Part I*, volume 11692 of *Lecture Notes in Computer Science*, pages 3–31. Springer, 2019.

[Jut98] Charanjit S. Jutla. Generalized birthday attacks on unbalanced feistel networks. In Hugo Krawczyk, editor, *Advances in Cryptology — CRYPTO '98*, pages 186–199, Berlin, Heidelberg, 1998. Springer Berlin Heidelberg.

[KHS+03] Jongsung Kim, Seokhie Hong, Jaechul Sung, Changhoon Lee, and Sangjin Lee. Impossible differential cryptanalysis for block cipher structures. In Thomas Johansson and Subhamoy Maitra, editors, *Progress in Cryptology - INDOCRYPT 2003, 4th International Conference on Cryptology in India, New Delhi, India, December 8-10, 2003, Proceedings*, volume 2904 of *Lecture Notes in Computer Science*, pages 82–96. Springer, 2003.

[LR85] Michael Luby and Charles Rackoff. How to construct pseudo-random permutations from pseudo-random functions (abstract). In Hugh C. Williams, editor, *Advances in Cryptology - CRYPTO '85, Santa Barbara, California, USA, August 18-22, 1985, Proceedings*, volume 218 of *Lecture Notes in Computer Science*, page 447. Springer, 1985.

[LR88] Michael Luby and Charles Rackoff. How to construct pseudorandom permutations from pseudorandom functions. *SIAM J. Comput.*, 17(2):373–386, 1988.

[Luc96] Stefan Lucks. Faster luby-rackoff ciphers. In Dieter Gollmann, editor, *Fast Software Encryption, Third International Workshop, Cambridge, UK, February 21-23, 1996, Proceedings*, volume 1039 of *Lecture Notes in Computer Science*, pages 189–203. Springer, 1996.

[Mea20] Catherine Meadows. Symbolic and computational reasoning about cryptographic modes of operation. *IACR Cryptol. ePrint Arch.*, page 794, 2020.

[MI08] Atsushi Mitsuda and Tetsu Iwata. Tweakable pseudorandom permutation from generalized feistel structure. In Joonsang Baek, Feng Bao, Kefei Chen, and Xuejia Lai, editors, *Provable Security, Second International Conference, ProvSec 2008, Shanghai, China, October 30 - November 1, 2008. Proceedings*,

volume 5324 of *Lecture Notes in Computer Science*, pages 22–37. Springer, 2008.

[Mic25] Microsoft Research. Z3 theorem prover. https://github.com/Z3Prover/z3, 2025. Version 4.15.4.

[MP03] Ueli M. Maurer and Krzysztof Pietrzak. The security of many-round luby-rackoff pseudo-random permutations. In Eli Biham, editor, *Advances in Cryptology - EUROCRYPT 2003, International Conference on the Theory and Applications of Cryptographic Techniques, Warsaw, Poland, May 4-8, 2003, Proceedings*, volume 2656 of *Lecture Notes in Computer Science*, pages 544–561. Springer, 2003.

[MPS+25] Stefan Milius, Dominik Paulus, Dominique Schröder, Lutz Schröder, and Julian Thomas. Automated analysis and synthesis of message authentication codes. *IACR Cryptol. ePrint Arch.*, page 1165, 2025.

[MV00] Shiho Moriai and Serge Vaudenay. On the pseudorandomness of top-level schemes of block ciphers. In Tatsuaki Okamoto, editor, *Advances in Cryptology - ASIACRYPT 2000, 6th International Conference on the Theory and Application of Cryptology and Information Security, Kyoto, Japan, December 3-7, 2000, Proceedings*, volume 1976 of *Lecture Notes in Computer Science*, pages 289–302. Springer, 2000.

[NPV17] Valérie Nachef, Jacques Patarin, and Emmanuel Volte. *Feistel Ciphers - Security Proofs and Cryptanalysis*. Springer, 2017.

[NR99] Moni Naor and Omer Reingold. On the construction of pseudorandom permutations: Luby-rackoff revisited. *J. Cryptol.*, 12(1):29–66, 1999.

[NVP13] Valérie Nachef, Emmanuel Volte, and Jacques Patarin. Differential attacks on generalized feistel schemes. In Michel Abdalla, Cristina Nita-Rotaru, and Ricardo Dahab, editors, *Cryptology and Network Security - 12th International Conference, CANS 2013, Paraty, Brazil, November 20-22. 2013. Proceedings*, volume 8257 of *Lecture Notes in Computer Science*, pages 1–19. Springer, 2013.

[Nyb96] Kaisa Nyberg. Generalized feistel networks. In Kwangjo Kim and Tsutomu Matsumoto, editors, *Advances in Cryptology - ASIACRYPT '96, International Conference on the Theory and Applications of Cryptology and Information Security, Kyongju, Korea, November 3-7, 1996, Proceedings*, volume 1163 of *Lecture Notes in Computer Science*, pages 91–104. Springer, 1996.

[Pat08] Jacques Patarin. The "coefficients h" technique. In Roberto Maria Avanzi, Liam Keliher, and Francesco Sica, editors, *Selected Areas in Cryptography, 15th International Workshop, SAC 2008, Sackville, New Brunswick, Canada, August 14-15, Revised Selected Papers*, volume 5381 of *Lecture Notes in Computer Science*, pages 328–345. Springer, 2008.

[PNB06] Jacques Patarin, Valérie Nachef, and Côme Berbain. Generic attacks on unbalanced feistel schemes with contracting functions. In Xuejia Lai and Kefei Chen, editors, *Advances in Cryptology - ASIACRYPT 2006, 12th International Conference on the Theory and Application of Cryptology and Information Security, Shanghai, China, December 3-7, 2006, Proceedings*, volume 4284 of *Lecture Notes in Computer Science*, pages 396–411. Springer, 2006.

[PNB07] Jacques Patarin, Valérie Nachef, and Côme Berbain. Generic attacks on unbalanced feistel schemes with expanding functions. In Kaoru Kurosawa, editor, *Advances in Cryptology - ASIACRYPT 2007, 13th International Conference on the Theory and Application of Cryptology and Information Security, Kuching, Malaysia, December 2-6, 2007, Proceedings*, volume 4833 of *Lecture Notes in Computer Science*, pages 325–341. Springer, 2007.

[RRSY98] Ronald L. Rivest, M.J.B. Robshaw, R. Sidney, and Y.L. Yin. The rc6 block cipher. *In First advanced encryption standard (AES) conference*, august 1998.

[SD20] The Sage Developers. SageMath, the Sage Mathematics Software System. https://www.sagemath.org, 2020. Version 9.3.

[SGW20] Yaobin Shen, Chun Guo, and Lei Wang. Improved security bounds for generalized feistel networks. *IACR Trans. Symmetric Cryptol.*, 2020(1):425–457, 2020.

[Shi10] Kyoji Shibutani. On the diffusion of generalized feistel structures regarding differential and linear cryptanalysis. In Alex Biryukov, Guang Gong, and Douglas R. Stinson, editors, *Selected Areas in Cryptography - 17th International Workshop, SAC 2010, Waterloo, Ontario, Canada, August 12-13, 2010, Revised Selected Papers*, volume 6544 of *Lecture Notes in Computer Science*, pages 211–228. Springer, 2010.

[Sho01] Victor Shoup. OAEP reconsidered. In Joe Kilian, editor, *Advances in Cryptology - CRYPTO 2001, 21st Annual International Cryptology Conference, Santa Barbara, California, USA, August 19-23, 2001, Proceedings*, volume 2139 of *Lecture Notes in Computer Science*, pages 239–259. Springer, 2001.

[SK96] Bruce Schneier and John Kelsey. Unbalanced feistel networks and block cipher design. In Dieter Gollmann, editor, *Fast Software Encryption, Third International Workshop, Cambridge, UK, February 21-23, 1996, Proceedings*, volume 1039 of *Lecture Notes in Computer Science*, pages 121–144. Springer, 1996.

[SM10] Tomoyasu Suzaki and Kazuhiko Minematsu. Improving the generalized feistel. In Seokhie Hong and Tetsu Iwata, editors, *Fast Software Encryption, 17th International Workshop, FSE 2010, Seoul, Korea, February 7-10, 2010, Revised Selected Papers*, volume 6147 of *Lecture Notes in Computer Science*, pages 19–39. Springer, 2010.

[SMMK12] Tomoyasu Suzaki, Kazuhiko Minematsu, Sumio Morioka, and Eita Kobayashi. $\textnormal{\textsc{TWINE}}$: A lightweight block cipher for multiple platforms. In Lars R. Knudsen and Huapeng Wu, editors, *Selected Areas in Cryptography, 19th International Conference, SAC 2012, Windsor, ON, Canada, August 15-16, 2012, Revised Selected Papers*, volume 7707 of *Lecture Notes in Computer Science*, pages 339–354. Springer, 2012.

[SSA+07] Taizo Shirai, Kyoji Shibutani, Toru Akishita, Shiho Moriai, and Tetsu Iwata. The 128-bit blockcipher CLEFIA (extended abstract). In Alex Biryukov, editor, *Fast Software Encryption, 14th International Workshop, FSE 2007, Luxembourg, Luxembourg, March 26-28, 2007, Revised Selected Papers*, volume 4593 of *Lecture Notes in Computer Science*, pages 181–195. Springer, 2007.

[YI11] Shingo Yanagihara and Tetsu Iwata. On permutation layer of type 1, source-heavy, and target-heavy generalized feistel structures. In Dongdai Lin, Gene

Tsudik, and Xiaoyun Wang, editors, *Cryptology and Network Security - 10th International Conference, CANS 2011, Sanya, China, December 10-12, 2011. Proceedings*, volume 7092 of *Lecture Notes in Computer Science*, pages 98–117. Springer, 2011.

[YI13] Shingo Yanagihara and Tetsu Iwata. Improving the permutation layer of type 1, type 3, source-heavy, and target-heavy generalized feistel structures. *IEICE Trans. Fundam. Electron. Commun. Comput. Sci.*, 96-A(1):2–14, 2013.

[YI14] Shingo Yanagihara and Tetsu Iwata. Type 1.x generalized feistel structures. *IEICE Trans. Fundam. Electron. Commun. Comput. Sci.*, 97-A(4):952–963, 2014.

[YL16] Li Yu-Ling. Unprovable security of cctr authentication mode. *Journal of Cryptologic Research*, 3(4):374–381, 2016.

[ZMI89] Yuliang Zheng, Tsutomu Matsumoto, and Hideki Imai. On the construction of block ciphers provably secure and not relying on any unproved hypotheses. In Gilles Brassard, editor, *Advances in Cryptology - CRYPTO '89, 9th Annual International Cryptology Conference, Santa Barbara, California, USA, August 20-24, 1989, Proceedings*, volume 435 of *Lecture Notes in Computer Science*, pages 461–480. Springer, 1989.

Appendix A Computer-aided Proof Results

A.1 Variant Structures of Type-1 and Type-2 GFN

Numerous studies have aimed to improve the diffusion properties of Type-1 and Type-2 GFNs by searching for optimal permutations. For example, Yanagihara and Iwata [YI11, YI13] modified the block-wise cyclic shift in Type-1 GFNs and presented an explicit construction for the optimal permutation. For Type-2 GFNs, Derbez et al. [DFLM19] and Delaune et al. [DDGP22] proposed algorithms to identify the optimal even-odd and non-even-odd permutations, respectively.

With the help of AutoEGFN, we evaluate the (S)PRP security of these structures with different branch numbers. Detailed results are summarized in Tables 5, 6 and 7, with the corresponding permutations specified in Table 8.

Table 5: Computer-aided proof results for YI11's Type-1 with different branch numbers.

Structure	**Branch number**	**Permutation layer**	**Parameters**			**our # of rounds**	
			r_1	r_2	r_3	PRP	SPRP
YI11's Type-1	3	p_1	4	3	2	7	9
	4	p_2	9	4	3	13	16
	5	p_3	12	5	8	17	25
	6	p_4	25	6	5	31	36
	7	p_5	24	7	18	31	49
	8	p_6	35	8	21	43	64
	9	p_7	40	9	32	49	81
	10	p_8	63	10	27	73	100
	11	p_9	60	11	50	71	121
	12	p_{10}	77	12	55	89	144
	13	p_{11}	84	13	72	97	169
	14	p_{12}	117	14	65	131	196
	15	p_{13}	112	15	98	127	225
	16	p_{14}	135	16	105	151	256

Table 6: Computer-aided proof results for DFLM19's GFNs with different branch numbers.

Structure	**Branch number**	**Permutation layer**	**Parameters**			**our # of rounds**	
			r_1	r_2	r_3	PRP	SPRP
DFLM19's GFN	28	p_{15}	8	2	8	10	18
	30	p_{16}	8	2	8	10	18
	32	p_{17}	8	2	8	10	18
	36	p_{18}	8	2	8	10	18

Type-1.x GFN [YI14] is an extension of the classic Type-1/2 GFN. It constructs a family of GFNs that parameterized by different parameters (d, f), where d is the number of branches and f is the number of round functions per round. Figure 4 shows the structure for $(d, f) = (6, 2)$. Yanagihara et al. [YI14] studied the diffusion properties (DR_{max} [SM10]) of Type-1.x and its security against saturation, differential, impossible differential, and linear attacks.

However, few studies have analyzed the (S)PRP security of the Type-1.x GFN. We evaluate its PRP and SPRP security using AutoEGFN, with the optimal permutations listed in Table 9 taken from [YI14, Section 7].

Table 7: Computer-aided proof results for DDGP22's GFNs with different permutation layers.

Structure	**Branch number**	**Permutation layer**	**Parameters**			**our # of rounds**	
			r_1	r_2	r_3	PRP	SPRP
DDGP22's GFN	22	p_{19}	8	3	8	11	19
		p_{20}	8	3	8	11	19
		p_{21}	8	3	8	11	19
		p_{22}	8	3	8	11	19
		p_{23}	8	3	8	11	19
		p_{24}	8	3	8	11	19
		p_{25}	8	3	8	11	19
		p_{26}	8	3	8	11	19

Table 8: Permutation layers used in YI11's Type-1 GFN, DFLM19's GFN, and DDGP22's GFN.

	Permutation layer
p_1	$(1, 2, 0)$
p_2	$(1, 2, 3, 0)$
p_3	$(2, 3, 1, 4, 0)$
p_4	$(1, 2, 3, 4, 5, 0)$
p_5	$(2, 3, 4, 5, 1, 6, 0)$
p_6	$(2, 3, 4, 5, 1, 6, 7, 0)$
p_7	$(2, 3, 4, 5, 6, 7, 1, 8, 0)$
p_8	$(2, 3, 4, 5, 1, 6, 7, 8, 9, 0)$
p_9	$(2, 3, 4, 5, 6, 7, 8, 9, 1, 10, 0)$
p_{10}	$(2, 3, 4, 5, 6, 7, 8, 9, 1, 10, 11, 0)$
p_{11}	$(2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 1, 12, 0)$
p_{12}	$(2, 3, 4, 5, 6, 7, 8, 9, 1, 10, 11, 12, 13, 0)$
p_{13}	$(2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 1, 14, 0)$
p_{14}	$(2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 1, 14, 15, 0)$
p_{15}	$p = (1, 2, 3, 4, 5, 0, 7, 8, 9, 10, 11, 6, 12, 13)$ $q = (10, 7, 13, 11, 9, 8, 4, 1, 12, 5, 3, 2, 6, 0)$
p_{16}	$p = (1, 2, 3, 4, 5, 6, 7, 8, 9, 0, 11, 10, 13, 12, 14)$ $q = (12, 5, 10, 3, 11, 1, 13, 9, 14, 7, 4, 6, 2, 8, 0)$
p_{17}	$p = (1, 2, 3, 4, 5, 0, 7, 8, 9, 10, 11, 6, 13, 14, 12, 15)$ $q = (2, 6, 12, 10, 1, 13, 4, 15, 7, 9, 14, 5, 8, 3, 11, 0)$
p_{18}	$p = (1, 2, 3, 4, 5, 6, 7, 0, 9, 10, 11, 12, 13, 14, 15, 8, 16, 17)$ $q = (10, 9, 14, 12, 15, 11, 13, 17, 2, 1, 6, 4, 7, 3, 5, 16, 8, 0)$
p_{19}	$(3, 18, 5, 16, 7, 12, 9, 10, 1, 14, 13, 2, 15, 8, 11, 21, 17, 4, 19, 6, 0, 20)$
p_{20}	$(3, 6, 5, 12, 7, 10, 9, 18, 1, 2, 13, 4, 15, 16, 17, 8, 11, 21, 19, 14, 0, 20)$
p_{21}	$(3, 12, 5, 0, 7, 10, 9, 18, 1, 2, 13, 4, 15, 16, 17, 21, 11, 8, 19, 14, 6, 20)$
p_{22}	$(3, 8, 5, 16, 7, 21, 9, 14, 1, 2, 13, 18, 15, 0, 17, 6, 11, 12, 19, 4, 10, 20)$
p_{23}	$(3, 21, 5, 10, 7, 0, 9, 14, 1, 2, 13, 18, 15, 8, 17, 6, 11, 12, 19, 4, 16, 20)$
p_{24}	$(3, 8, 5, 6, 7, 4, 1, 12, 11, 2, 9, 21, 15, 19, 13, 17, 10, 16, 14, 20, 0, 18)$
p_{25}	$(3, 4, 5, 14, 7, 0, 9, 16, 11, 2, 1, 12, 15, 21, 13, 6, 19, 10, 17, 8, 18, 20)$
p_{26}	$(3, 6, 5, 10, 7, 16, 9, 18, 11, 14, 1, 2, 15, 4, 13, 0, 19, 8, 17, 21, 12, 20)$

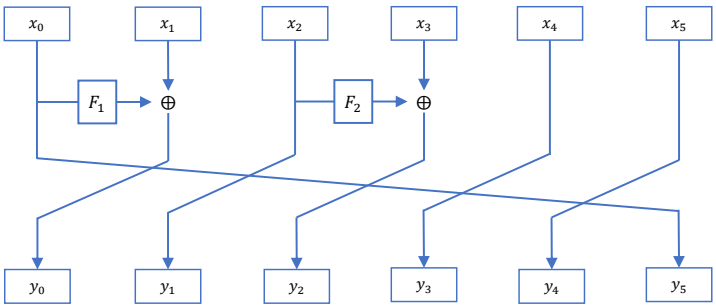


Figure 4: Encryption process of Type-1.x $(d, f) = (6, 2)$ with permutation layer $(5, 0, 1, 2, 3, 4)$.

Table 9: Computer-aided proof results for Type-1.x GFNs with different branch numbers.

Structure	(d, f)	**Permutation layer**	**Parameters**			**our # of rounds**	
			r_1	r_2	r_3	PRP	SPRP
Type-1.x GFN	$(3, 1)$	$(1, 2, 0)$	4	3	2	7	9
	$(4, 1)$	$(1, 2, 3, 0)$	9	4	3	13	16
	$(5, 1)$	$(2, 3, 1, 4, 0)$	12	5	8	17	25
	$(6, 1)$	$(1, 2, 3, 4, 5, 0)$	25	6	5	31	36
	$(7, 1)$	$(2, 3, 4, 5, 1, 6, 0)$	24	7	18	31	49
	$(8, 1)$	$(2, 3, 4, 5, 1, 6, 7, 0)$	35	8	21	43	64
	$(4, 2)$	$(1, 2, 3, 0)$	3	2	3	5	8
	$(5, 2)$	$(1, 2, 4, 0, 3)$	4	3	6	7	13
	$(6, 2)$	$(3, 4, 5, 0, 2, 1)$	6	4	6	10	16
	$(7, 2)$	$(1, 2, 4, 0, 5, 6, 3)$	6	5	11	11	22
	$(8, 2)$	$(1, 2, 4, 0, 5, 6, 7, 3)$	7	6	13	13	26
	$(9, 2)$	$(1, 3, 4, 2, 5, 6, 7, 8, 0)$	13	8	13	21	34
	$(6, 3)$	$(1, 2, 5, 0, 3, 4)$	4	2	4	6	10
	$(7, 3)$	$(1, 2, 4, 0, 5, 6, 3)$	5	3	6	8	14
	$(8, 3)$	$(3, 4, 1, 5, 6, 0, 7, 2)$	8	4	8	12	20

A.2 SH GFN and TH GFN

In this section, we study the PRP and SPRP security of SH and TH GFNs using AutoEGFN. Security properties of SH/TH GFN [SK96] have been extensively analyzed. Julta [Jut98] proposed generalized birthday attacks on TH GFNs, while Patarin et al. [PNB06, PNB07] developed distinguishing attacks against both structures. Yanagihara and Iwata [YI11] analyzed their diffusion properties, and Guo et al. [GJNS16] later demonstrated generic meet-in-the-middle attacks with different key lengths. The structures of 4-branch SH and TH GFNs are shown in Figure 5.

However, few studies have analyzed the (S)PRP security of the SH/TH GFNs. With AutoEGFN, we study the PRP and SPRP security of these structures with branch numbers from 3 to 8. To facilitate our analysis, we define the round functions as follows. For SH GFN, the round function $F : \{0,1\}^{(d-1)n} \to \{0,1\}^n$ is written as

$$F(x_0, x_1, \cdots, x_{d-2}) = F_0(x_0) \oplus F_1(x_1) \oplus \cdots \oplus F_{d-2}(x_{d-2}),$$

where each F_i is a permutation over $\{0,1\}^n$, $0 \le i \le d-2$. For TH GFN, the round

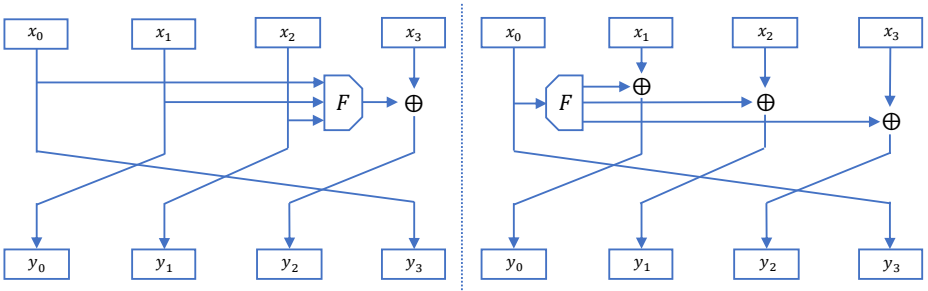


Figure 5: Encryption process of SH GFN (Left) and TH GFN (Right) with 4 branches.

function $F : \{0,1\}^n \to \{0,1\}^{(d-1)n}$ is written as

$$F(x_0) = (F_1(x_0), F_2(x_0), \cdots, F_{d-1}(x_0)),$$

where each F_i is a permutation over $\{0,1\}^n$, $1 \le i \le d-1$. The permutations listed in Table 10 are all derived from the optimal permutations provided in [YI11, Section 5].

Table 10: Computer-aided proof results for SH GFNs and TH GFNs with different branch numbers.

Structure	**Branch number**	**Permutation layer**	**Parameters**			**our # of rounds**	
			r_1	r_2	r_3	PRP	SPRP
SH GFN	3	$(1,2,0)$	2	3	2	5	7
	4	$(1,2,3,0)$	3	4	3	7	10
	5	$(1,2,3,4,0)$	4	5	4	9	13
	6	$(1,2,3,4,5,0)$	5	6	5	11	16
	7	$(1,2,3,4,5,6,0)$	6	7	6	13	19
	8	$(1,2,3,4,5,6,7,0)$	7	8	7	15	22
TH GFN	3	$(1,2,0)$	2	2	2	4	6
	4	$(1,2,3,0)$	3	2	3	5	8
	5	$(1,2,3,4,0)$	4	2	4	6	10
	6	$(1,2,3,4,5,0)$	5	2	5	7	12
	7	$(1,2,3,4,5,6,0)$	6	2	6	8	14
	8	$(1,2,3,4,5,6,7,0)$	7	2	7	9	16

A.3 Nyberg's GFN, SM's GFN and BMT's EGFN

In this section, we study the PRP and SPRP security of three notable EGFNs using AutoEGFN. We first analyze Nyberg's GFN [Nyb96] and SM's GFN [SM10], followed by BMT's EGFN [BMT13]. Diagrams of the first two structures are shown in Figure 6.

We apply AutoEGFN to Nyberg's and SM's GFNs under different branch numbers. The detailed results are presented in Table 11 and Table 12, respectively. Notably, our tool can efficiently analyze Nyberg's GFN with more than 32 branches.

Figure 1 shows the structure of BMT's EGFN. While the security of the 8- and 16-branch variants has been proven in [BMT13, Theorem 8] using techniques similar to [SM10, Theorem 1], we use AutoEGFN to evaluate the security across various branch numbers. The results, detailed in Table 13, show consistency with [BMT13] for 8- and 16-branch variants. Furthermore, our tool can handle structures with more than 32 branches.

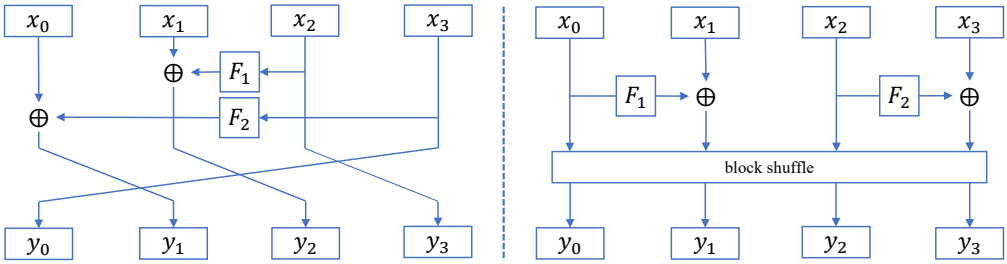


Figure 6: Encryption process of Nyberg's GFN (Left) and SM's GFN (Right) with 4 branches.

Table 11: Computer-aided proof results for Nyberg's GFNs with different branch numbers.

Structure	**Branch number**	**Parameters**			**our # of rounds**	
		r_1	r_2	r_3	PRP	SPRP
Nyberg's GFN	4	3	3	3	5	8
	6	5	4	5	9	14
	8	7	5	7	12	19
	10	9	6	9	15	24
	12	11	7	11	18	29
	14	13	8	13	21	34
	16	15	9	15	24	39
	32	31	17	31	48	79

Table 12: Computer-aided proof results for SM's GFNs with different branch numbers. SM's GFN employs the optimal even-odd shuffle as specified in [SM10, Appendix A].

Structure	**Branch**	**Parameters**			**our # of rounds**		**previous # of rounds**	
	number	r_1	r_2	r_3	PRP	SPRP	PRP [SM10]	SPRP [SM10]
SM's GFN	6	4	2	4	6	10	6	10
	8	5	2	5	7	12	7	12
	10	6	2	6	8	14	8	14
	12	7	2	7	9	16	9	16
	14	7	2	7	9	16	9	16
	16	7	2	7	9	16	9	16

Table 13: The computer-aided proof results for BMT's EGFNs with different branch numbers.

Structure	**Branch**	**Parameters**			**our # of rounds**		**previous # of rounds**	
	number	r_1	r_2	r_3	PRP	SPRP	PRP [BMT13]	SPRP [BMT13]
BMT's EGFN	4	2	2	2	4	6	4	6
	6	4	2	4	6	10	6	10
	8	4	2	4	6	10	6	10
	10	4	2	4	6	10	6	10
	12	4	2	4	6	10	6	10
	14	4	2	4	6	10	6	10
	16	4	2	4	6	10	6	10
	32	4	2	4	6	10	6	10