K. D. Tsaregorodtsev, Шифрование, сохраняющее формат: обзор, *Матем. вопр. криптогр.*, 2022, том 13, выпуск 2, 133–153

DOI: 10.4213/mvk412

МАТЕМАТИЧЕСКИЕ ВОПРОСЫ КРИПТОГРАФИИ
2022 Т. 13 № 2 С. 133–153

УДК 519.719.2 DOI https://doi.org/10.4213/mvk412

Format-preserving encryption: a survey

K. D. Tsaregorodtsev

JSC «NPK Kryptonite», *Moscow*

Получено 10.XI.2021

Abstract. This article gives a survey on the format-preserving encryption, algorithms proposed for standardization, and attacks on them. Additionally, we propose a new format-preserving encryption scheme based on quasigroup operations.

Keywords: format-preserving encryption, quasigroup, provable security

Шифрование, сохраняющее формат: обзор

К. Д. Царегородцев

АО «НПК Криптонит», *Москва*

Аннотация. Статья содержит обзор методов шифрования, сохраняющего формат, алгоритмов, предложенных для стандартизации, а также атак на них. Описана новая схема сохраняющего формат шифрования, основанная на квазигрупповых операциях.

Ключевые слова: сохраняющее формат шифрование, квазигруппа, доказуемая стойкость

Citation: Matematicheskie Voprosy Kriptografii, 2022, v. 13, № 2, pp. 133–153 (Russian)

Introduction

Format-preserving encryption (FPE) is an encryption algorithm with the following property: the resulting ciphertext format must be the same as the format of plaintext. For instance, if we encrypt 9-digit individual insurance account number (SNILS), the result must be 9-digit ciphertext. In this case, the cipher must look like a random permutation on the given (usually small) domain. The small size of the domain makes it hard to provide both strong security properties in the presence of an adversary (which can usually obtain all ciphertexts of all points in the domain due to its size) and keep the resulting algorithm as efficient as possible. Many attempts were made to build FPE scheme using standardized solutions (such as AES) and well-studied principles (Feistel networks). The current state of the art is unsatisfactory: most of the solutions proposed for standardization are broken in some sense.

In this paper we give a survey of algorithms and concrete attacks for FPE schemes. Also, we propose a new approach for FPE based on quasigroups operation.

The structure of the paper is the following: in Section 1 we give a formal definition of format-preserving encryption and tweakable block cipher. Section 2 is devoted to algorithms proposed for FPE, in Section 3 we consider cryptanalysis of these solutions. In Section 4 a new approach to the FPE algorithms based on quasigroup operations is presented.

The author is thankful to Stanislav Smyshlyaev for useful comments on the cycle walking technique attack and anonymous reviewers for their careful reading of the manuscript and many insightful comments and suggestions regarding both the content and language of this paper.

1. Preliminaries

1.1. Problem statement

The very first question is: why do we need format-preserving encryption (FPE) at all, and why is it not enough to use existing primitives such as block ciphers?

The request to preserve the format may be appropriate in the following situations.

1) Database structure may be incompatible with encrypted messages. If we want to keep the data encrypted, we either need to restructure the database or use FPE.

2) Some applications may require the data to be in a pre-defined format. In this case, we can rewrite an application from scratch or use FPE.

Why do usual block ciphers not solve the problem? A block cipher acts as a permutation on the fixed length binary strings (for instance, $\{0,1\}^{128}$ for Kuznyechik [1]). Even if the domain $\texttt{Dom}$ of data is embedded in $\{0,1\}^n$, the result of encrypting $m \in \texttt{Dom}$ is very unlikely to fall into the same subset: $E_k(m) \notin \texttt{Dom}$ due to its relatively small size in the real-world situations and applications, where $E_k(m)$ is an encryption of m under key k.

As an example, we can consider the case of credit card number (CCN).

Example 1. CCN consist of the following numbers: 6 digits — bank number, 9 digits — account number, 1 digits — checksum, and all digits, except for account number (i.e., 7 out of 16), are publicly available. In this case the domain $\texttt{Dom} = \{0,\dots 9\}^9$, i.e. $|\texttt{Dom}| \approx 2^{30}$.

Small domain size is dangerous. Due to a large number of possible input blocks for the usual block cipher (2^{64} or 2^{128}), it seems appropriate to ignore the capability of the adversary to collect the whole codebook (i.e. all pairs of the type $[m, E_k(m)]$) under some key k. In example 1 we have $|\texttt{Dom}| \approx 2^{30}$, which is perfectly feasible value to mount the dictionary attack.

Example 2 (Dictionary attack)**.** CCNs from different banks may have the same account number. Using the bijective property of the cipher, we can be sure that the matching ciphertext blocks correspond to matching plaintext (see Table 1).

Table 1. Bank account database

Bank number	Account number	Checksum
012345	$E_k(000001111)$	c_1
	$\updownarrow$ **same**	
987654	$E_k(000001111)$	c_2

1.2. Notation

We will use the following standard notation:

$x \xleftarrow{\$} X$ — choose x equiprobably at random from a set X,

$E_k(x)$ — encryption of a block x using block cipher E under a key k,

$x.keys$ — the set of all keys (indices) of an associative array (dictionary) x,

S_X — the set of all bijections (permutations) on a set X.

1.3. Tweakable block ciphers and FPE

As it was shown earlier (Example 2), the small size of the domain is a severe threat. In order to prevent the class of attacks with a dictionary a tweakable block cipher primitive may be used [2].

Definition 1. Tweakable block cipher (TBC) is a pair of algorithms:

$$E,\ D : \texttt{Keys} \times \texttt{Twk} \times \texttt{Dom} \to \texttt{Dom},$$

such that $D_k^t(E_k^t(m)) = m$, where $t \in \texttt{Twk}$ is a tweak (a block cipher parameter), $k \in \texttt{Keys}$ is a key, $m \in \texttt{Dom}$ is a message.

Usually, the sets of keys, tweaks, and messages for TBC are of the standard form $\{0,1\}^n$ for some n. The main idea of the construction is that $E_k^t(\cdot)$ are «weakly dependent» permutations for different $t \in \texttt{Twk}$.

Some properties of a tweak may be pointed out:

– tweak acts like IV/nonce in the usual block cipher modes of operation,

– the main goal of the tweak is to expand the set of possible permutations,

– tweak may not be secret.

The property of «weak dependence» may be formalized in provable security framework (see [3,4] for more details on provable security paradigm) as follows. Consider the following experiments:

Algorithm 1 Experiment Left

function INIT
  for $t \in \texttt{Twk}$ **do**
    $\pi^t \leftarrow^{\$} S_{\texttt{Dom}}$
function $\mathcal{O}(t, m)$
  return $\pi^t(m)$

Algorithm 2 Experiment Right

function INIT
  $k \leftarrow^{\$} \texttt{Keys}$
function $\mathcal{O}(t, m)$
  return $E_k^t(m)$

Definition 2. Let $Adv_E^{TPRP}(\mathcal{A})$ be the advantage of the adversary $\mathcal{A}$ in the distinguishing attack, i.e.:

$$Adv_E^{TPRP}(\mathcal{A}) = \mathbb{P}[Right(\mathcal{A}) \to 1] - \mathbb{P}[Left(\mathcal{A}) \to 1],$$

where Experiments $Right$ and $Left$ are defined above. Hereinafter we implicitly assume that the Experiment returns the bit outputted by the adversary.

By $InSec_E^{TPRP}(q_t, q_e, T)$ we denote the maximal advantage Adv_E^{TPRP} over all adversaries $\mathcal{A}$, whose running time does not exceed T, q_t is the number of different tweaks used by $\mathcal{A}$, q_e is the (maximal) number of encryption queries per tweak.

The probability $\mathbb{P}[\cdot]$:

1) is taken over random choice of permutations $\pi^t \leftarrow^{\$} S_{\mathtt{Dom}}$ for different tweaks t and random coins of $\mathcal{A}$ (if any) — in case of Left experiment,

2) is taken over random choice of key $k \leftarrow^{\$} \mathtt{Keys}$ and random coins of $\mathcal{A}$ (if any) — in case of Right experiment.

Note that the adversary's queries are of the form (t, m), i.e., t is not secret and is under adversary's control.

Example 3 (TBC)**.** If F is the usual PRP-strong block cipher, then the following construction is TBC:

$$E_k^t(m) := F_k(t \oplus F_k(m)).$$

The paper [2] gives the following estimation:

$$InSec_E^{TPRP}(q_t, q_e, T) \leqslant InSec_F^{PRP}(2q_tq_e, q_tq_e + T) + \mathcal{O}\left(\frac{(q_tq_e)^2}{2^n}\right),$$

where n is the length of the block (i.e. the domain is of the form $\mathtt{Dom} = \{0,1\}^n$), q_t, q_e and T are introduced in Definition 2.

1.4. Format-preserving encryption

Definition 3. Format-preserving encryption is a tweakable block cipher with an arbitrary set $\mathtt{Dom}$.

The main difference between TBC and FPE is that for TBC the domain $\mathtt{Dom}$ is usually of the standard form (i.e. $\mathtt{Dom} = \{0,1\}^{128}$), but in case of FPE we are interested in «atypical» domains, such as $\mathtt{Dom} = \{0, \ldots, 9\}^9$ for CCN. Also it is possible for FPE scheme to have an empty tweak space, i.e. $\mathtt{Twk} = \varnothing$. Let us note that disk encryption (see [5] for example) with the block size of 512 bits may be viewed as FPE scheme as well with $\mathtt{Dom} = \{0,1\}^{512}$. More formal treatment of FPE is given in [6].

The quest to develop a good FPE scheme for all sizes of domains seems hard at the moment. There are subtle issues involved in the case of small-size domains: for instance, the adversary can run an exhaustive search on

the small space. As a consequence, the cryptographic strength of the scheme strongly depends on the size of the domain.

First attempts to develop an encryption algorithm for arbitrary domain are described in [7,8]. Three papers [9–11] proposed standardized solutions for FPE algorithms. In 2016 NIST recommendations were issued based on these proposals. After the publication, a series of papers with significant advances in cryptanalytic techniques were published ([12–16]), which lead to theoretical threats and even practical attacks for these algorithms. The current state of the art is rather unsatisfactory: for domains with the size between $\sim 2^{20}$ to $\sim 2^{64}$ there is no good provably secure and efficient algorithm. For «tiny» domains, as well as for the «huge» one, there exists provably secure schemes (see [5, 17]).

2. Proposed FPE algorithms

This section gives an overview of algorithms proposed for NIST standardization (**FF1**-**FF3**), one of the ISO standardization candidates **FEA-2** and general techniques (cycle walking, prefix encryption) helpful in designing FPE algorithms. We do not cover wide-block/disk encryption schemes with the size of the domain larger than the block size of modern ciphers. Schemes based on principles other than Feistel networks are also not touched upon [18–21].

2.1. FF1, FF3

The structure of both algorithms is the same: the semi-balanced Feistel network over the group $\mathtt{Dom} = \mathbb{Z}_M \times \mathbb{Z}_N$, where $M \approx N$ (see [22] for more details).

The algorithm takes the key $k \in \mathtt{Keys}$, the element to be encrypted $(A, B) \in \mathtt{Dom}$, and the tweak $t \in \mathtt{Twk}$ (usually tweak space $\mathtt{Twk}$ is of the form $\{0,1\}^{tlen}$). One round of the encryption process transforms the pair via the following rule:

$$(A, B) \to (B, A \boxplus Q),$$

where $Q \in \mathbb{Z}_M$ is derived by the following rule:

$$Q = PRF_k(B, t, i, params),$$

where i is the round number, $params$ is some (non-secret) information, PRF is a pseudorandom function (see [3, Ch. 3.5.1, pp. 76–79] for the definition

of PRF). We omit some technical details here; see [22] for a full description of algorithms.

It was announced that 10 rounds of Feistel network are enough for **FF1** security and 8 rounds for **FF3**. The original paper ([9]) did not provide full security proof of the scheme. The arguments in favor of the proposed schemes include Patarin papers on the (classical) Feistel networks (see [23–25]) and paper on the Feistel networks over groups $\mathbb{Z}_M \times \mathbb{Z}_N$ [6].

Also, a wrong choice of tweak mixing in the PRF function was made in the **FF3** scheme, which leads to some specific attacks on the scheme. Slight modifications were proposed to mitigate these attacks.

2.2. FF2

Along with the two algorithms mentioned above (**FF1**, **FF3**), a third one (**FF2**) was initially proposed for standardization at NIST. Unfortunately, a design flaw leads to a theoretical attack on **FF2**. In this subsection, we briefly describe the main idea of the algorithm **FF2** (VAES3, [11]) and the corresponding attack ([26]).

The algorihm **FF2** consists of two steps. The input to the algorithm is three parameters: $k \in \texttt{Keys}, t \in \texttt{Twk}, m \in \texttt{Dom}$.

1. Derive the secret key for the given tweak:

$$sk = E_k(t) \in \{0,1\}^{128}.$$

2. Encrypt the message with obtained key:

$$c = Feistel_{sk}(m).$$

The main problem of the algorithm is that the key length $|sk| = 128$ is too short to guarantee the strong security bound.

If we have n ciphertexts of the form

$$c_j = Feistel_{E_k(t_j)}(m),$$

then we can try different keys $sk_j \in \{0,1\}^{128}$ and obtain

$$c'_j = Feistel_{sk_j}(m).$$

If some c'_j is the same as c_i for some i, then with high probability we will have $sk_j = E_k(t_j)$, i.e., we can recover the derived key for the given tweak t_j without knowing the value of master key k.

If the number of ciphertexts $n = 2^u$, then the collision is expected to occur after 2^{128-u} steps. Even though the attack is somewhat hypothetical, it was decided to finalize standard NIST SP 800-38G without **FF2** (but see also more recent work on **FF2** [27]).

2.3. FEA-2 algorithm

In [28], a new family of tweakable block ciphers based on Feistel networks was presented. One round of the proposed scheme is depicted below (taken from the original paper [28]).

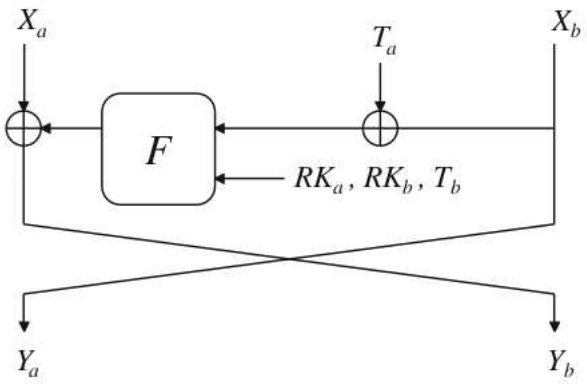

Fig. 1. One round of FEA-2

The following notation is used:

– $X_a \,||\, X_b$ — left and right blocks of the message;

– $T_a \,||\, T_b$ — left and right blocks of the tweak;

– $RK_a \,||\, RK_b$ — left and right blocks of the round key;

It is assumed that $|T_a| = |X_b|, |T_a| + |T_b| = 128 = |RK_a| = |RK_b|$).
Some remarkable features of the algorithm are the following:

1. In contrast with the previous algorithms (**FF1**–**FF3**), this proposition embeds tweak at the primitive layer, i.e., tweak is used in each encryption round.

2. Unlike simple block ciphers, **FEA-2** provides the ability to encrypt messages of various lenghts, $\mathtt{Dom} = \{0,1\}^n$, where $n \in \{8, 9, \ldots 128\}$.

3. The number of rounds in Feistel network depends on the block size and starts with 18.

4. The key lenght is not fixed: $klen \in \{128, 192, 256\}$.

5. Encryption on the domain $\texttt{Dom} = \{1, \ldots, N\}$ is done via embedding $\texttt{Dom} \subseteq \{0,1\}^n$ combined with the cycle walking technique (see Subsection 2.5 for more details).

The article [28] is devoted to the consideration of applications of various cryptanalytic techniques to the **FEA-2** algorithm. Linear and differential analysis, as well as related-key attacks, were investigated. Additionally, attacks specific to Feistel networks over small domains were analyzed. The authors claimed that for the domains of size greater than 2^8 the proposed attacks require at least 2^{64} encryptions on different parameters $t \in \texttt{Twk}$.

2.4. Prefix encryption

If the domain $\texttt{Dom}$ is small enough, then we can use the following idea:

1. Compute the number list:

$$S = (E_k(0), \ldots, E_k(N-1));$$

2. To encrypt the message $m \in \mathbb{Z}_N$, map m to the position of $E_k(m)$ in the sorted list.

This method is provably secure but requires $O(N)$ encryption operations at the initial step and $O(N)$ memory to store the table.

2.5. Cycle walking

If the domain $\texttt{Dom}$ has the property that $\frac{|\texttt{Dom}|}{2^n} \leqslant 1$ is close to 1 for some «standard» block size n (for instance, $n = 64$ or $n = 128$), then the following approach works:

1. For $m \in \texttt{Dom}$ compute $c \leftarrow E_k(m)$.

2. If $c \in \texttt{Dom}$, then m maps to c.

3. If $c \notin \texttt{Dom}$, then $c \leftarrow E_k(c)$ and go to step 2.

The expected number of encryption operations (before one obtains $c \in \texttt{Dom}$) is determined by the quantity $\frac{2^n}{|\texttt{Dom}|}$.

It may be shown that the algorithm is provably secure ([8,29]), and side-channel attacks (due to the time leakage for various length of encryption operation) does not give additional information to the adversary in the following sense.

Let us assume that there are some *fixed* domain $\texttt{Dom}$ and some $TPRP$-secure block cipher $E(\cdot)$. Consider the two following experiments. The first one provides to the adversary an oracle for the result of applying a real cycle-walking cryptoscheme (constructed for the domain $\texttt{Dom}$ using cipher $E(\cdot)$) and a real cycle length l for $x \in \texttt{Dom}$. The second Experiment provides an oracle which gives the result of applying a random permutation (parametrized by $t \in \texttt{Twk}$) and *simulates* the cycle length l. If E is $TPRP$-secure, then it is hard to distinguish between these two experiments, i.e. additional information about cycle length does not give rise to the timing attacks.

However, the result mentioned above is relative to the model used. In the most recent paper [30] it was shown that if the adversary is able to manipulate the size of the domain $\texttt{Dom}$ without affecting the mapping used for cycle walking, then the practical attacks on the scheme are possible. This possibility is not reflected in the original security model for cycle walking, hence the result does not formally contradict previous papers.

2.6. FNR and DTP algorithms

Cisco company proposed **FNR** algorithm (see [31]), based on the Feistel cipher with two additional permutations. Protegrity company proposed **DTP** algorithm ([32]). In [14] it was shown that:

- **FNR** is slightly worse than **FF1** and **FF3**,
- **DTP** has serious weaknesses, which lead to a total break.

2.7. Techniques for tiny-size domains

The following methods for tiny-space domains with the size $|\texttt{Dom}| = N$ are mentioned in [33]:

1. Exhaustive permutation numbering: each key k is mapped to a number between 1 and $N!$, the point $x \in \texttt{Dom}$ is mapped to $\pi_k(x)$.
2. Knuth–Fisher–Yates shuffle: to shuffle the array of numbers one has to repeatedly choose an element from a decreasing prefix and moving it to the end. The element $x \in \texttt{Dom} = \{1, \ldots, N\}$ is mapped into the position of x in the shuffled array.

3. Prefix encryption method mentioned earlier in Subsection 2.4.

Each of the methods is provably secure, but encryption time (or set-up time) is proportional to the size of the domain, hence the limitation on the size.

3. Summary of cryptanalysis of proposed algorithms

This section summarizes cryptanalysis on **FF1**, **FF3** and generic algorithms based on Feistel networks.

Currently, there are two types of attacks on FPE algorithms:

– The first type of cryptanalytic attacks exploits the wrong design of tweak mixing (specific for **FF3**). In these attacks the adversary adaptively chooses plaintexts to be encrypted on two selected $t_1, t_2 \in \mathtt{Twk}$.

– The second type uses the intrinsic feature of Feistel network over small domains. The general idea is that the proposed number of rounds is not enough to hide the plaintext statistics: there is a slight bias after one round of Feistel network, which can be boosted using different tweaks $t \in \mathtt{Twk}$ for the same message. This asymmetry can be exploited to mount the distinguishing attack (or even key recovery attacks).

By attack we do not necessarily mean *practical* attack: as usual, by attack we understand a method to violate some claimed security property better than generic methods (i.e., message recovery for almost exhaustive search on a given tweak or distinguisher with a brute force key search).

The specificity of attacks on FPE algorithms is that the number of required texts formally exceeds the domain size, but in fact only a minimal (even constant in most of the attacks proposed in the literature) number of plaintexts are required for each $t \in \mathtt{Twk}$. This observation was not reflected in the original security model [6]. All proofs were obtained in a weaker model, in which the adversary cannot make the number of requests to the oracle that exceeds the domain size.

The following notation in Tables 2, 3, 4 is used:

n — bitsize of one half of the message $m \in \mathtt{Dom}$, i.e. $\mathtt{Dom} = \{0,1\}^{2n}$,

$N = 2^n$ — number of different parts of the message,

r — number of rounds in Feistel network,

q_t — number of different tweaks used in the attack,

q_e — number of plaintexts per tweak used in the attack,

T — time complexity of the attack.

Table 2 covers existing attacks on generic Feistel Networks with tweaks. Table 3 summarizes attacks specific for **FF3** algorithm (and one attack on **FF2**). One of the recent attacks ([16]) has a significant impact on FEA-type ciphers. The results from [16] are presented in Table 4 (the value q is the total number of tweak-plaintext queries, i.e. $q = q_t \cdot q_e$ assuming that equal number of plaintexts q_e is used for each tweak).

4. Quasigroup based FPE

In this section we describe one possible approach to FPE. Due to the cycle-walking technique, we can limit our consideration to domains of the particular form $\mathtt{Dom} = \{0,1\}^n$ for some «small» n.

4.1. Quasigroups

Definition 4 ([34])**.** Quasigroup is a set Q with a binary operation on it $\circ : Q \times Q \to Q$, which obeys the following property: for each $a, b \in Q$ there exist unique $x, y \in Q$ such that:

$$a \circ x = b, \qquad y \circ a = b.$$

In other words, operations of left and right multiplication

$$L_a : Q \to Q,\ L_a(x) = a \circ x,$$

$$R_a : Q \to Q,\ R_a(y) = y \circ a$$

are bijections on Q.

Some applications of quasigroup theory to cryptography may be found in [35, 36].

Table 2. Generic Feistel Networks attacks on FPE algorithms

Paper	Resources	Threat	Comments
2004, [24]	$q_t = N^{r-2}$ $q_e = 2$ $T \approx q_t \cdot q_e$	Distinguisher	Attack distinguishes Feistel network output from a random string
2016, [12]	$q_t = \mathcal{O}(n \cdot N^{r-2})$ $q_e = 3$ $T = \mathcal{O}(q_e \cdot q_t)$	Message recovery	1. The adversary knows ciphertexts of three different messages (x, x', x^*) under tweaks $t_1, \ldots, t_q$, and recovers the message x. 2. The message x' is fully known to the adversary but unrelated to x. 3. x^* and x share a common right side; only the left side of x^* is known to the adversary. 4. Nonadaptive attack: only the knowledge of plaintexts is required.
2018, [14]	$q_t = \mathcal{O}(N^{r-4}(n \cdot N + p))$ $q_e = \mathcal{O}(n \cdot N)$ $T = \mathcal{O}(n \cdot N^{r-2}(n+p))$	Recovery of multiple messages $m_1, \ldots, m_p$	1. Nonadaptive attack: only the knowledge of plaintexts is required. 2. It is assumed that the adversary knows ciphertexts for τ known plaintexts $x_1, \ldots x_\tau$ and for p messages (plaintexts) under attack $m_1, \ldots m_p$ under q different tweaks. 3. It is assumed that right halves of $x_1, \ldots, x_\tau$ comprise all possible right halves of messages. 4. Correlation between $x_1, \ldots, x_\tau$ and $m_1, \ldots m_p$ is not required.
2020, [16]	$q_t = 4 \cdot N^{r-6}$ $q_e = N^2$ (i.e. the entire codebook for each tweak) $T = \mathcal{O}(N^{r-3})$	Distinguisher	Attack distinguishes Feistel network output from a random string

Table 3. Specific attacks on **FF2-FF3** algorithms

Paper	Resources	Threat	Comments
2015, [26]	$q_t = q$ $q_e = 1$ $T \approx \frac{2^{128}}{q}$	Subkey recovery for a given tweak, **FF2**	The attack is not adaptive, the knowledge of m is required, same $m \in \texttt{Dom}$ is used for all tweaks
2017, [13]	$q_t = 2$ $q_e = \mathcal{O}(N^{\frac{11}{6}})$ $T = \mathcal{O}(N^5)$	Entire codebook recovery for a given tweaks t_1, t_2 for **FF3**	1. The adaptive choice of messages is required. 2. The adversary is assumed to control the choice of $t \in \texttt{Twk}$. The attack does not work if the adversary does not have complete control over t. Partial truncation of the tweak can be applied (as shown in the [13]) to prevent this attack.
2019, [15]	$q_t = 2$ $q_e = \mathcal{O}(N^{\frac{11}{6}})$ $T = \mathcal{O}(N^{\frac{17}{6}})$	Entire codebook recovery for t_1, t_2 for **FF3**	1. The attack is the strengthened version of [13]. 2. The adaptive choice of messages is required. 3. The attack does not work if the adversary cannot obtain full control over t.

Table 4. Recent attack on FPE algorithms [16]

Algorithm	Resourses	Threat
FF1, $klen = 128, r = 10$	$q = 2^{60}, T = 2^{70}$	Distinguishing attack
FF3-1, $klen = 128, r = 8$	$q = 2^{80}, T = 2^{100}$	Distinguishing attack
FEA-2, $klen = 128, r = 18$	$q = 2^{80}, T = 2^{84}$	Distinguishing attack
FEA-2, $klen = 256, r = 24$	$q = 2^{80}, T = 2^{84}$	Distinguishing attack
FEA-1, $klen = 192, r = 14$	$q = 2^{36}, T = 2^{136}$	Key recovery
FEA-1, $klen = 256, r = 16$	$q = 2^{48}, T = 2^{136}$	Key recovery

We use the following measure of quasigroup operation complexity. Given some quasigroup Q, we want to measure how the composition of quasigroup operations (for instance, left multiplications) is close to the random permutation on Q. To formalize this notion, we introduce the following Experiments (the λ parameter is analogous to the security parameter in classical cryptography).

Algorithm 3 Experiment Left

1: **function** INIT(λ)
2: $\quad \pi \leftarrow^{\$} S_Q$
3: **function** $\mathcal{O}(m)$
4: $\quad$ **return** $\pi(m)$

Algorithm 4 Experiment Right

1: **function** INIT(λ)
2: $\quad k_1, \ldots, k_\lambda \leftarrow^{\$} Q$
3: **function** $\mathcal{O}(m)$
4: $\quad$ **return** $k_1 \circ (k_2 \circ (\ldots (k_\lambda \circ m) \ldots)$

Again, as in Subsection 1.3, we introduce an adversary $\mathcal{A}$, who tries to distinguish between random and «structured» permutations.

Definition 5. Let $Adv_Q^{PRP}(\mathcal{A})$ be the advantage of the adversary $\mathcal{A}$ in the quasigroup operation distinguishing attack, i.e.:

$$Adv_Q^{PRP}(\mathcal{A}) = \mathbb{P}[Right(\mathcal{A}) \to 1] - \mathbb{P}[Left(\mathcal{A}) \to 1],$$

where experiments $Left$ and $Right$ are defined above.

Let $InSec(t, q)$ be the maximal advantage Adv_Q^{PRP} over all adversaries $\mathcal{A}$, whose running time does not exceed t and who uses no more than q $\mathcal{O}$-oracle queries.

We want this quantity to be as small as possible for the given t and q. The quantity **directly depends** on the structure of the quasigroup Q. The inappropriate choice of quasigroup (i.e., $Q = \mathbb{Z}_N$) can make the problem trivial to solve.

Why this problem can be hard at all? The reason for that is that there exists a class of groups (called polynomially complete quasigroups), for which the problem of deciding whether or not an equation over such a quasigroup has a solution is NP-complete [37].

Definition 6. Quasigroup Q of size k is called functionally (polynomially) complete if the system of functions consisting of binary operation $\circ$ and all constants $x \in Q$ (considered as functions of arity 0) with the operation of superposition generate all possible functions over Q, i.e.

$$[\{\circ\} \cup \{x \in Q\}] = P_k.$$

Polynomially complete qusigroups are actively studied [38–40]. There exists an algorithm (polynomial in the size of the quasigroup) that checks whether the quasigroup is polynomially complete [40]. The NP-completeness of decision problem ([37]) is an evidence in favor of hardness of problem (as it is stated in Experiment from Subsection 4.1)

in the worst case. However, the problem is not (yet) studied in the average-case scenario, and it is not clear what the value of parameter λ should be, as well as whether the problem is hard in the average case. As it is pointed out in the conclusion (see Section 5), this is one of the main directions of future research.

Let Q be the quasigroup over the set `Dom`. We will use the following method: given the key $k \in$ `Keys` and the tweak $t \in$ `Twk`, we will first use some keyed pseudorandom generator PRG (see [3, Ch. 3.3.1, pp. 60–64] or [41, Ch. 3]) to produce a sequence of «random-looking» and «independent» elements $z_i \in Q, i = 1, \ldots, \lambda$, where λ is the parameter of the scheme and is chosen based on the quasigroup structure (it is selected in such a way that the distinguishing problem (as it is stated in Experiment from Subsection 4.1) is hard to solve).

Then we will encrypt our message $m \in$ `Dom` using the quasigroup operation. Some possible variants might be:

$$m \to L_{z_\lambda}(\ldots L_{z_1}(m)\ldots) = z_\lambda \circ (\ldots \circ z_2 \circ (z_1 \circ m)\ldots), \tag{1}$$

$$m \to R_{z_\lambda}(\ldots R_{z_1}(m)\ldots) = (\ldots (m \circ z_1) \circ z_2 \ldots) \circ z_\lambda, \tag{2}$$

$$m \to D_{z_\lambda}(\ldots D_{z_1}(m)\ldots). \tag{3}$$

In description of operation (3) we are using the following agreement: the operation D_{z_i} equals L_{z_i} if i-th bit of output of some random generator (for instance, based on values k and t) is equal to 0, and R_{z_i} otherwise. In this case, operations (1), (2) are special cases of operation (3). We describe several possible variants of the scheme because a scheme with only left (or only right) multiplication might be vulnerable.

Now we will describe the variant of scheme (1) in more detail. We will present a series of Experiments, where Experiment 0 is the original cryptosystem, and the final Experiment is (semantically) Left Experiment from Subsection 1.3. Our goal is to show that (under assumptions that PRG used is good and the quantity $InSec$ from Definition 5 is small) the scheme is secure in the TPRP model (as it is stated in the Experiment from Subsection 1.3.

Theorem 1. *Let q_t be the maximal number of different tweaks, q_e be the maximal number of encryption queries per tweak, T is the number of operations (running time). Then the following estimate holds*:

$$InSec^{TPRP}(q_t, q_e, T) \leqslant InSec^{PRG}(q_t, T + \lambda q_t q_e) + q_t InSec_Q^{PRP}(q_e, T + (1+\lambda^2) q_e q_t).$$

Proof. Denote by $L_{z_\lambda \dots z_1}(m)$ the following operation:

$$L_{z_\lambda \dots z_1}(m) = z_\lambda \circ (\dots \circ z_2 \circ (z_1 \circ m) \dots).$$

The first experiment Exp^0 is the original scheme. The transition from Exp^0 to Exp^1 is done via replacement of PRG by the random choice of $z_i \in Q$. The last transition from Exp^1 to Exp^2 is done by replacing the operation $L_{z_\lambda \dots z_1}(m)$ with the operation $\pi^t(m)$.

Algorithm 5 Exp^0

function INIT
  $k \leftarrow^{\$}$ `Keys`
  $zs = \{\}$
function $\mathcal{O}(t, m)$
  if $t \notin zs.keys$ **then**
    $z_1, \dots, z_\lambda \leftarrow^{\$} PRG_k(t)$
    $zs[t] = (z_1, \dots, z_\lambda)$
  $z_1, \dots, z_\lambda \leftarrow zs[t]$
  $res \leftarrow L_{z_\lambda \dots z_1}(m)$
  return res

Algorithm 6 Exp^1

function INIT
  $k \leftarrow^{\$}$ `Keys`
  $zs = \{\}$
function $\mathcal{O}(t, m)$
  if $t \notin zs.keys$ **then**
    $z_1, \dots, z_\lambda \leftarrow^{\$} Q$
    $zs[t] = (z_1, \dots, z_\lambda)$
  $z_1, \dots, z_\lambda \leftarrow zs[t]$
  $res \leftarrow L_{z_\lambda \dots z_1}(m)$
  return res

Algorithm 7 Exp^2

function INIT
  $zs = \{\}$
function $\mathcal{O}(t, m)$
  if $t \notin zs.keys$ **then**
    $\pi^t \leftarrow^{\$} S_Q$
    $zs[t] = \pi^t$
  $\pi^t \leftarrow zs[t]$
  $res \leftarrow \pi^t(m)$
  return res

For any adversary $\mathcal{A}$, we can write the following equality:

$$\begin{aligned} Adv^{TPRP}(\mathcal{A}) &= \mathbb{P}[Exp^0(\mathcal{A}) \to 1] - \mathbb{P}[Exp^2(\mathcal{A}) \to 1] \\ &= \Big(\mathbb{P}[Exp^0(\mathcal{A}) \to 1] - \mathbb{P}[Exp^1(\mathcal{A}) \to 1]\Big) \\ &\qquad + \Big(\mathbb{P}[Exp^1(\mathcal{A}) \to 1] - \mathbb{P}[Exp^2(\mathcal{A}) \to 1]\Big). \end{aligned}$$

Then we can upperbound each of the summands.

The first difference is small due to the fact that PRG is a good pseudorandom generator. Namely, if $\mathcal{A}$ can distinguish between Exp^0 and Exp^1 with high probability, then it can be used to attack the pseudorandomness property of PRG. We can construct an adversary $\mathcal{B}$, who runs $\mathcal{A}$ as a subroutine. When $\mathcal{A}$ asks for encryption of m under fresh tweak t, $\mathcal{B}$ calls his oracle on the input t and gets values $z_1, \ldots, z_\lambda$. It saves the values in the memory, computes $res \leftarrow L_{z_\lambda \ldots z_1}(m)$ and gives it to $\mathcal{A}$. At the end of experiment $\mathcal{B}$ outputs the resulting bit of $\mathcal{A}$. The running time of $\mathcal{B}$ is the running time of $\mathcal{A}$ plus the time needed for simulation: $t_{\mathcal{B}} \leqslant t_{\mathcal{A}} + \lambda q_e q_t$ (we assume here for simplicity that operation $a \circ b$ may be done in one step). The number of oracle queries $q_{\mathcal{B}} = q_t$. Thus, we obtain the following estimate:

$$\Big(\mathbb{P}[Exp^0(\mathcal{A}) \to 1] - \mathbb{P}[Exp^1(\mathcal{A}) \to 1]\Big) \leqslant InSec^{PRG}(q_t, T + \lambda q_t q_e).$$

The second transition may be done via standard hybrid argument technique (see [3] for example). We have at most q_t queries for the fresh $(z_1, \ldots, z_\lambda)$, and for each tuple $(z_1, \ldots, z_\lambda)$ we ask no more than q_e encryption queries. We will replace all queries of encrypting m on i-th tweak t_i (i.e. all $L_{z_\lambda \ldots z_1}(m)$, z_i are chosen uniformly from Q) by $\pi(m)$ (where π is chosen uniformly from S_Q).

If there is an adversary $\mathcal{A}$, who can distinguish this replacement, then we can use $\mathcal{A}$ to construct $\mathcal{B}$, who will attack PRP-property of $L_{z_\lambda \ldots z_1}(\cdot)$:

$(m_1^{t_1}, t_1)\ (m_1^{t_2}, t_2)\ \ldots (m_1^{t_{i-1}}, t_{i-1})$	$(m_1^{t_i}, t_i)$	$(m_1^{t_{i+1}}, t_{i+1})\ \ldots (m_1^{t_{q_t}}, t_{q_t})$
$\vdots$	$\vdots$	$\vdots$
Simulated via choosing $(z_1, \ldots, z_\lambda) \leftarrow^{\$} Q$	Oracle queries	Simulated via choosing $\pi^t \leftarrow^{\$} S_Q$
Running time $\leqslant \lambda^2 q_e q_t$	$q_{\mathcal{B}} \leqslant q_e$	Running time $\leqslant q_e q_t$

The running time of $\mathcal{B}$ is the time of $\mathcal{A}$ plus the time to simulate all other environment (except for the queries on tweak t_i):

$$T_{\mathcal{B}} \leqslant T_{\mathcal{A}} + \lambda^2 q_e q_t + q_e q_t$$

(we assume here for simplicity that operation $a \circ b$ and a random choice of the element $q \leftarrow^{\$} Q$ may be done in one step). The number of oracle

queries $q_{\mathcal{B}} \leqslant q_e$. Thus, we obtain the following estimate:

$$\mathbb{P}[Exp^1(\mathcal{A}) \to 1] - \mathbb{P}[Exp^2(\mathcal{A}) \to 1] \leqslant q_t \cdot InSec_Q^{PRP}(q_e, T + (1+\lambda^2)q_e q_t).$$

Combining both estimates we get the statement of the theorem. □

We stress out that currently there are no concrete estimates of the hardness of the problem from Definition 5, hence no concrete bounds on $InSec^{TPRP}(q_t, q_e, T)$.

5. Conclusion

This paper contains a survey of the FPE, algorithms proposed for standardization (**FF1**-**FF3**, **FEA-2**), and of attacks on them. We propose a new cryptoalgorithm based on quasigroup operations.

Further areas of research may be useful.

1. The study of specific classes of quasigroups as a basis for proposed cryptosystem (with an emphasis on the polynomially complete quasigroups).

2. Estimating the hardness of problem stated in definition 5 based on existing results on NP-completeness of problem of deciding whether the equation has the solution over polynomially complete quasigroup [37].

3. Implementing the cryptosystem over specific quasigroups and estimating statistical properties of resulting algorithms.

References

[1] "Information technology. Cryptographic data security. Block ciphers. GOST R 34.12-2015", 2016 (in Russian).

[2] Liskov M., Rivest R., Wagner D., "Tweakable Block Ciphers", *J. Cryptology*, **24**:3 (2011), 588–613.

[3] Katz J., Lindell Y., *Introduction to Modern Cryptography*, CRC press, Boca Raton, Florida, 2020, 626 pp.

[4] Guo F., Susilo W., Mu Y., *Introduction to Security Reduction*, Springer, Cham, Switzerland, 2018, 253 pp.

[5] Alekseev E. K., Akhmetzyanova L. R., Babueva A. A., Smyshlyaev S. V., "Data storage security and full disk encryption", *Prikl. Diskr. Mat.*, 2020, № 49, 78–97.

[6] Bellare M., Ristenpart T., Rogaway P., Stegers T., "Format-preserving encryption", SAC 2009, Lect. Notes Comput. Sci., **5867**, 2009, 295–312.

[7] Brightwell M., Smith H., "Using datatype-preserving encryption to enhance data warehouse security", 20th Nat. Inf. Syst. Security Conf. Proc. (NISSC), 141–149.

[8] Black J., Rogaway P., "Ciphers with arbitrary finite domains", CT-RSA 2002, Lect. Notes Comput. Sci., **2271**, 2002, 114–130.

[9] Bellare M., Rogaway P., Spies T., "The FFX mode of operation for format-preserving encryption", *NIST submission*, 2010.

[10] Brier E., Peyrin T., Stern J., "BPS: A format-preserving encryption proposal", *NIST submission*, 2010.

[11] Vance J., "VAES3 scheme for FFX: An addendum to the FFX mode of operation for format preserving encryption", *NIST submission*, 2011.

[12] Bellare M., Hoang V. T., Tessaro S., "Message-recovery attacks on Feistel-based format preserving encryption", *Proc.* 2016 *ACM SIGSAC Conf. Comput. and Commun. Security*, 2016, 444–455.

[13] Durak F. B., Vaudenay S., "Breaking the FF3 format-preserving encryption standard over small domains", Lect. Notes Comput. Sci., **10402**, 2017, 679–707.

[14] Hoang V. T., Tessaro S., Trieu N., "The curse of small domains: new attacks on format-preserving encryption", CRYPTO 2018, Lect. Notes Comput. Sci., **10991**, 2018, 221–251.

[15] Hoang V. T., Miller D., Trieu N., "Attacks Only Get Better: How to Break FF3 on Large Domains", EUROCRYPT 2019, Lect. Notes Comput. Sci., **11477**, 2019, 85–116.

[16] Dunkelman O., Kumar A., Lambooij E., Sanadhya S. K., "Cryptanalysis of Feistel-based format-preserving encryption", *Cryptology ePrint Archive*, *Report* 2020/1311, 2020, `https://eprint.iacr.org/2020/1311`.

[17] Morris B., Rogaway P., Stegers T., "How to encipher messages on a small domain", CRYPTO 2009, Lect. Notes Comput. Sci., **5677**, 2009, 286–302.

[18] Thorp E. O., "Nonrandom shuffling with applications to the game of Faro", *J. Amer. Statist. Assoc.*, **68** (1973), 842–847.

[19] Granboulan L., Pornin T., "Perfect block ciphers with small blocks", Lect. Notes Comput. Sci., **4593**, 2007, 452–465.

[20] Chang D., Ghosh M., Gupta K. C., Jati A., Kumar A., Moon D., Ray I. G., Sanadhya S. K., "SPF: a new family of efficient format-preserving encryption algorithms", Inscrypt 2016, Lect. Notes Comput. Sci., **10143**, 2016, 64–83.

[21] Morris B., Rogaway P., Stegers T., "Deterministic encryption with the Thorp shuffle", *J. Cryptology*, **31**:2 (2018), 521–536.

[22] Dworkin M., "Recommendation for block cipher modes of operation: methods for format-preserving encryption", *NIST Special Publication* 800-38*G*, 2016.

[23] Patarin J., "Luby-Rackoff: 7 Rounds Are Enough for $2^{n(1-\epsilon)}$ Security", CRYPTO 2003, Lect. Notes Comput. Sci., **2729**, 2003, 513–529.

[24] Patarin J., "Security of Random Feistel Schemes with 5 or More Rounds", Lect. Notes Comput. Sci., **3152**, 2004, 106–122.

[25] Nachef V., Patarin J., Volte E., *Feistel Ciphers — Security Proofs and Cryptanalysis*, Springer, Cham, Switzerland, 2017, 309 pp.

[26] Dworkin M., Perlner R., "Analysis of VAES3 (FF2)", *Cryptology ePrint Archive*, *Report* 2015/306, 2015, `https://eprint.iacr.org/2015/306`.

[27] Vance J., Bellare M., "An extension of the FF2 FPE scheme", *NIST submission*, 2014.

[28] Lee J.-K., Koo B., Roh D., Kim W.-H., Kwon D., "Format-preserving encryption algorithms using families of tweakable blockciphers", ICISC 2014, Lect. Notes Comput. Sci., **8949**, 2014, 132–159.

[29] Li J., Jia C., Liu Z., Dong Z., "Cycle-walking revisited: consistency, security, and efficiency", *Security and Communic. Networks*, **6**:8 (2013), 985–992.

[30] Amon O., Dunkelman O., Keller N., Ronen E., Shamir A., "Three third generation attacks on the format preserving encryption scheme FF3", EUROCRYPT 2021, Lect. Notes Comput. Sci., **12697**, 2021, 127–154.

[31] Sashank D., Fluhrer S., "FNR: arbitrary length small domain block cipher proposal", SPACE 2014, Lect. Notes Comput. Sci., **8894**, 2014, 146–154.

[32] Mattsson U., "Format-controlling encryption using datatype-preserving encryption", *Cryptology ePrint Archive*, *Report* 2009/257, 2009, `https://eprint.iacr.org/2009/257`.

[33] Rogaway P., "A Synopsis of Format-Preserving Encryption", unpublished manuscript, 2010.

[34] Keedwell A., Denes J., *Latin Squares and Their Applications*, 2nd ed., Burlington, North Holland, 2015, 438 pp.

[35] Shcherbacov V., *Elements of quasigroup theory and applications*, CRC Press, Boca Raton, Florida, 2017, 598 pp.

[36] Glukhov M. M., "Some applications of quasigroups in cryptography", *Prikl. Diskr. Matem.*, **2**:2 (2008), 28–32 (in Russian).

[37] Horváth G., Nehaniv C., Szabó C., "An assertion concerning functionally complete algebras and NP-completeness", *Theor. comput. sci.*, **407**:1-3 (2008), 591–595.

[38] Artamonov V. A., Chakrabarti S., Pal S. K., "Characterization of polynomially complete quasigroups based on Latin squares for cryptographic transformations", *Discrete Applied Mathematics*, **200** (2016), 5–17.

[39] Artamonov V. A., "Quasigroups and their applications", *Chebyshevskii Sbornik*, **19**:2 (2018), 111–122 (in Russian).

[40] Galatenko A., Pankratiev A., Rodin S., "Polynomial completeness of finite quasigroups", *Intell. Syst.*, **23**:1 (2019), 81–87 (in Russian).

[41] Goldreich O., *Foundations of Cryptography*: *Vol.* 1, *Basic Tools*, Cambridge Univ. Press, Cambridge, UK, 2001.