BREAKING THE FF3 FORMAT PRESERVING ENCRYPTION*

F. Betül Durak[1] and Serge Vaudenay[2]

[1] Rutgers University
Department of Computer Science
fbdurak@cs.rutgers.edu
[2] Ecole Polytechnique Fédérale de Lausanne (EPFL)
LASEC - Security and Cryptography Laboratory
Lausanne, Switzerland
serge.vaudenay@epfl.ch

Abstract. The NIST standard FF3 scheme (also known as BPS scheme) is a tweakable block cipher based on a 8-round Feistel Network. We break it with a practical attack. Our attack exploits the bad domain separation in FF3 design. The attack works with chosen plaintexts and tweaks when the message domain is small. Our FF3 attack requires $\mathsf{O}(\mathsf{N}^{\frac{11}{6}})$ chosen plaintexts with time complexity N^5, where N^2 is domain size to the Feistel Network.
Due to the bad domain separation in 8-round FF3, we reduced the FF3 attack to an attack on 4-round Feistel Networks. In our generic attack, we reconstruct the entire codebook of 4-round Feistel Network with $\mathsf{N}^{\frac{3}{2}}\left(\frac{\mathsf{N}}{2}\right)^{\frac{1}{6}}$ known plaintexts and time complexity N^4.

1 Introduction

Encryption schemes provide a handy tool to data owners to protect their data privacy when stored on untrusted servers or transmitted over insecure channels. The design of encryption schemes often shaped by the desired properties of data that keeps it functional, secure, or computable on. The standard block cipher designs such as AES aim to build schemes from 128-bit strings to 128-bit stings. For an arbitrary length ℓ, modes of operation is a way to encrypt ℓ-bit strings into ℓ-bit string for arbitrary values of ℓ. When the goal of encryption scheme is to preserve the format of the data in ciphertexts (i.e. the data not necessarily binary and large but instead decimal and small), we need a cipher design that encrypts the data into the same format as itself. This is called Format Preserving Encryption [7, 10] and evolved as a most useful tool in applied cryptography.

Format Preserving Encryption (FPE) schemes encrypt a message into a ciphertext by providing its format such as a valid credit card number (CCN) into a valid CCN or a valid social security number (SSN) into a valid SSN. Thus,

[*] Short Version

FPE provides a way to store the data without changing the database scheme or application software that runs on a specific data format.

Ideally, FPE constructions should not be format specific for different types of formats. In particular, the task of FPE scheme can be reduced to design an FPE for an integral domain as it allows to apply the exact encryption to all other formats. Therefore, the cheap way to build an FPE scheme is to construct an easy to compute 1-to-1 mapping from message domain to itself which is a power of a basis (such as 2^ℓ or N^2). More precisely, we desire to encrypt on $\mathbb{Z}_{\mathsf{N}_\mathsf{l}} \times \mathbb{Z}_{\mathsf{N}_\mathsf{r}}$ with $\mathsf{N}_\mathsf{l} \approx \mathsf{N}_\mathsf{r}$.[3]

The National Institute of Standards and Technology (NIST) published an FPE standard [1] (finalized in March 2016) that includes two-approved Feistel-based FPE: FF1 [4] and FF3 [6]. Both FF1 and FF3 standards are tweakable Feistel schemes with modular addition. FF1 is standardized with round numbers 10 whereas FF3 is standardized with 8 rounds.

In this work, we are particularly interested in the attacks for breaking the FN-based standard FF3 [1] and attacks against Feistel Network. The former attack utilizes the latter that is designed as a generic round function recovery attack. In FF3, a tweak XORed with a round counter is used as an input to the block cipher. A tweak is a public information and can be controlled by the adversary. The XOR operation guarantees that round functions are pairwise different. This is usually called "domain separation". The security of FF3 asserts that it achieves several cryptographic goals including chosen-plaintext security or even PRP-security against an adaptive chosen-ciphertext attack under the assumption that the underlying round function is a good pseudorandom function (PRF). Our work shows that its security goal has not met even when the round functions are replaced by secure PRFs and gives a round function recovery attack on FF3.

Overview Of Previous Work: A security for message recovery in FPE constructions along with many other notions for FPE was first defined by Bellare et. al. [3]. A recent work by Bellare et. al. [2] gives a practical message recovery attack on NIST standard Feistel-based FPE (both FF1 and FF3) on small domain sizes. In this work, however, the security definition they consider is under the new message recovery security that they define in the same work. The attack by Bellare et. al. in [2] works using $\mathsf{O}(\mathsf{N}^5 \log \mathsf{N})$ data and time complexity with many tweaks on 8 rounds. This is quite interesting when the amount of data is limited for each tweak. It is a decryption attack. Our attack herein is more traditional. It uses only two tweaks, but $\mathsf{O}(\mathsf{N}^{\frac{11}{6}})$ chosen plaintexts with $\mathsf{O}(\mathsf{N}^5)$ time complexity. We recover the entire codebook (for both tweaks).

Since its invention, Feistel Networks have created active research areas for cryptographers (both in theory and in practice). The security of Feistel schemes aims to either distinguish a Feistel scheme from a random permutation or to recover the round functions. In their famous work [8], Luby and Rackoff proved

[3] When the ranking is to a domain the size of which is not exactly a product $\mathsf{N}_\mathsf{l} \times \mathsf{N}_\mathsf{r}$, we can use additional techniques such as shuffling (random walk in a graph) in a slightly larger domain.

the indistinguishability of Feistel Networks for 3-rounds against chosen plaintext attack and 4-rounds against chosen plaintext and ciphertext attacks for the number of queries $q \ll \sqrt{2^n}$, where $2n$ is the input length. The directions derived from this result tried to improve the security bounds until $q \ll 2^n$ (that is called the "birthday bound"). A work by Patarin [9], using the mirror theory, showed improved proofs and stronger security bounds for 4, 5, and 6 rounds Feistel Networks. Namely, for $q \ll 2^n$, 4 rounds are secure against known plaintext attacks, 5 rounds are secure against chosen plaintext attacks, and 6 rounds are secure against chosen plaintext and ciphertext attacks.

From an information theory viewpoint, we could recover all functions in time $2^{O(n2^n)}$ by exhaustive search. Our attack uses $q \sim 2^{\frac{3}{2}n}$ and is polynomial in 2^n with known plaintexts up to 4 rounds.

2 Known Plaintext Round Function Recovery Attack on Feistel Scheme

In a recent work by Biryukov et. al. [5], the new cryptanalysis results against Feistel Networks with modular addition for 4 and 5 rounds are presented. For 4 rounds, they achieve the full recovery of round functions with data complexity $O(2^{\frac{3n}{2}})$ with a guess and determine technique, where $2n$ is defined as message length. They use chosen plaintexts and ciphertexts. We summarize their results and ours on Table 1.

# rounds	mode	time	data	ref
3	known plaintext	2^n	2^n	Section 2.1
4	chosen plaintext and ciphertext	$2^{3n/2}$	$2^{3n/2}$	[5]
4	known plaintext	2^{3n}	$2^{3n/2}$	Section 2.2
5	chosen plaintext and ciphertext	$2^{n2^{3n/4}}$	2^{2n}	[5]
5	chosen plaintext	$2^{O(n2^{n/2})}$	$2^{3n/2}$	Section 2.2
$r \geqslant 6$	chosen plaintext	$2^{(r-5)n2^n}$	$2^{3n/2}$	Section 2.2

Table 1. Round function recovery attacks against balanced Feistel schemes with two n-bit branches and any addition rule (we omitted polynomial terms in n)

2.1 Round Function Recovery on 3-Round Feistel Scheme

Consider a 3-round Feistel Scheme with three round functions F_0, F_1, F_2 and modular addition. Given x and y in $\mathcal{X}$, we define:

$$\begin{aligned} c &= x + F_0(y) \\ t &= y + F_1(c) \\ z &= c + F_2(t) \end{aligned} \tag{1}$$

A 3-round Feistel scheme with (F_0, F_1, F_2) for an input (xy) will output the same pair (zt) to a 3-round Feistel scheme with round functions $(F_0-\delta, F_1', F_2+\delta)$ for the input (xy) with $F_1'(c) = F_1(c+\delta)$. This allows us to fix F_0 on one point arbitrarily (i.e. we can reconstruct F_0 up to a constant δ). The idea of our attack is to concentrate on data for which we know how to evaluate F_0 so that we can deduce the output for the round function F_2. Then, we concentrate on data for which we know how to evaluate F_2 and we deduce more points in F_0. We continue by alternating the deduction between F_0 and F_2 until we recover them all. When we continue iterating as described, we can fully recover the tables for all three round functions (F_0, F_1, F_2). Our attack presented in Algorithm 1 in more detail.

```
Algorithm 1: $(F_0, F_1, F_2)$ Recovery Attack
  Input  : a set $S$ of tuples $(xyzt)$ of size $\theta N$.
1 Take a subset $S_1 \subseteq S$ of size $\theta$ such that $y$ is constant in $S_1$.
2 Fix $F_0(y) = 0$ arbitrarily and make a 2-round attack to deduce $\theta$ tuples
    $(c, y, z, t)$. Notice that when $F_0(y) = 0$, $c = x$. Since $c = x$, we collect $\theta$
    equations of the form $F_2(t) = z - c$.
3 Take the subset $S_2 \subseteq S$ of all $(xyzt) \in S$ such that $\exists(x'y'z't') \in S_1$ with
    $t = t'$. The expected size of $S_2$ is $\theta^2$.
4 Using the $\theta$ equations $F_2(t) = z - c$, we deduce $\theta^2$ tuples $(xyct)$. From
    these tuples, we obtain $\theta^2$ equations of the form $F_0(y) = c - x$.
5 Take the subset $S_3 \subseteq S$ of all $(xyzt) \in S$ such that $\exists(x'y'z't') \in S_2$ with
    $y = y'$. The expected size of $S_3$ is $\theta^3$.
6 Using the $\theta^2$ equations $F_0(y) = c - x$ we obtained from $S_2$, we collect $\theta^3$
    equations $F_2(t)$ in $S_3$.
7 Play yo-yo until nothing new is recovered.
  Output: (partial) tables for $F_0F_1F_2$
```

We model our set S as a bipartite graph with two parties of N vertices (one for the y's and the other for the t's) and edges for each (y, t) pair in tuples from S. What our algorithm does is just looking for a connected component of a random starting point y with complexity $O(\theta N)$. Following the theory of random graphs, we have θN random edges so that the graph is likely to be fully connected when $\theta \approx \ln(N)$. For a constant $\theta \geqslant 1$, it is likely to have a giant connected component. This component corresponds to a constant fraction of the tables of F_0 and F_2. Therefore, after $\log_\theta N$ iterations, we can reconstruct F_0 and F_2 which allow us to reconstruct F_1. For any y, we can see that it does not appear in S with probability $\left(1-\frac{1}{N}\right)^{\theta N} \approx 1 - e^{-\theta}$. Thus, we can only hope to recover a fraction $1 - e^{-\theta}$ of the table of F_0. The same holds for F_1 and F_2. When we set $\theta = 2$, we need $2N$ tuples to apply this attack on 3-round Feistel Scheme.

2.2 Round Function Recovery on 4-Round Feistel Scheme

Consider a 4-round Feistel scheme with round functions F_0, F_1, F_2, F_3. Given x and y in $\mathcal{X}$, we define the following equations (See Fig. 1 (a)):

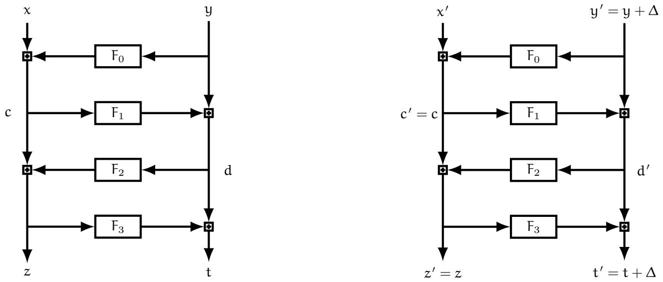
$$
\begin{aligned}
c &= x + F_0(y)\\
d &= y + F_1(c)\\
z &= c + F_2(d)\\
t &= d + F_3(z)
\end{aligned}
$$

Assume that we collected M random pairwise different plaintext messages (x, y). We collect the pairs:

$$V = \{(xyzt, x'y'z't') \mid z' = z, t' - y' = t - y, xy \neq x'y'\}$$

$$V_{good} = \{(xyzt, x'y'z't') \mid z' = z, c' = c, xy \neq x'y'\}$$

where c, d (respectively c', d') are defined from (x, y, z, t) (respectively form (x', y', z', t')) as above. We define $Label(xyzt, x'y'z't') = x - x'$.



(a) (b)

Fig. 1. 4-round Feistel Scheme Attack

We define a directed graph $G = (V, E)$ with the vertex set V as defined above. We take $(x_1y_1z_1t_1x_1'y_1'z_1't_1', x_2y_2z_2t_2x_2'y_2'z_2't_2') \in E$ if $y_1' = y_2$ (i.e. a vertex v_1 is connected to a vertex v_2 if the y_1' in the second message of v_1 is same as the first message in v_2). Furthermore, we let $E_{good} = V_{good}^2 \cap E$ and define the sub-graph $G_{good} = (V_{good}, E_{good})$.

We state an important observations in the following Lemma with four properties:

Lemma 1. *Given a graph* G *with a vertex set* V *defined as above:*

1. $V_{good} \subseteq V$.
2. *If* $(xy, x'y') \in V_{good}$, *then* $F_0(y') - F_0(y) = Label(xy, x'y')$.
3. *For all cycles* $v_1v_2...v_Lv_1$ *of* G_{good}, $\sum_{i=1}^{L} Label(v_i) = 0$.

The principle of our attack is as follows: if we get vertices in $\mathsf{V_{good}}$, the property 2 from Lemma 1 gives equations to characterize F_0. One problem is that we can identify vertices in V, but we cannot tell apart good and bad ones. One way to recognize good vertices is to use property 3 in Lemma 1: to find cycles with zero sum of labels. For this, we will prove in Lemma 3 that this is a characteristic property of good cycles, meaning that all the vertices in these cycles are good vertices.

Lemma 2. *For* $\mathsf{F}_0, \mathsf{F}_1, \mathsf{F}_2, \mathsf{F}_3$ *random,* $\mathbb{E}\left(\frac{\#\mathsf{V_{good}}}{\#\mathsf{V}}\right) = \frac{1-\frac{1}{N}}{2-\frac{1}{N}} \approx \frac{1}{2}$.

We have the property that for each cycle $\nu_1\nu_2...\nu_L\nu_1 \in \mathsf{G}$, if $\nu_1, ..., \nu_L$ are all in $\mathsf{V_{good}}$, then the sum of $\mathsf{Label}(\nu_i)$ is zero due to Lemma 1, property 3. If one vertex is not good, the sum may be random. This suggests a way to find good vertices in V that is to look for long cycles in G with a zero sum of labels.

Lemma 3. *(*$\mathsf{L} = 2$ *case) If* $\nu_1 = (x_1y_1z_1t_1, x_1'y_1'z_1't_1')$ *we say that* ν_1 *and* ν_2 *are permuting if* $\nu_2 = (x_1'y_1'z_1't_1', x_1y_1z_1t_1)$. *If* $\nu_1\nu_2\nu_1$ *is a cycle in* G *with zero sum of labels, and* ν_1, ν_2 *are not permuting, then* ν_1 *and* ν_2 *are likely to be good. More precisely, for* ν_1 *and* ν_2 *random, we have*
$\Pr[\nu_1, \nu_2 \in \mathsf{V_{good}} \mid \nu_1\nu_2\nu_1 \text{ is a cycle}, \nu_1, \nu_2 \text{ not permuting}, \sum_{i=1}^{2} \mathsf{Label}(\nu_i) = 0]$
$\geqslant \frac{1}{1+\frac{10}{N-5}}$.

We believe that Lemma 3 remains true for valid cycles of small length except in trivial cases. We extend to $\mathsf{L} > 2$ for cycles satisfying some special non-repeating condition [$\neg$repeat] on the c and d values to rule out many trivial cases. However, this condition [$\neg$repeat] cannot be checked by the adversary. Instead, we could just avoid repetitions of any message throughout the cycle (as repeating messages induce repeating c's or d's). We use the following conjecture (which is supported by experiment for $\mathsf{L} = 3$),

Conjecture 1 *If* $\nu_1\nu_2...\nu_L\nu_1$ *is a random cycle of length* L *in* G *with zero sum of labels and the vertices use no messages in common, then* $\nu_1...\nu_L$ *are all good with probability close to 1.*

Now, we give the full algorithm of our attack to 4-round Feistel scheme.

```
Algorithm 2: ($F_0, F_1, F_2, F_3$) Recovery Attack (Strategy $S_2$)
Input  : $M$ known plaintexts and ciphertext pairs ($xyzt$)
Create $G = (V, E)$.
Collect non-trivial cycles of length L with zero label sum.
Deduce $\frac{M^{2L}}{N^{3L}}$ relations $Label(\nu_i) = F_0(y') - F_0(y)$
Create $G'$ from $\{y, y'\}$ from the collected vertices.
Find the largest connected component C in $G'$ (works for $M \geqslant N^{\frac{3}{2}} + \frac{1}{2L}$).
Assign one $F_0(y)$ value arbitrarily and deduce $F_0$ on C.
We have $\frac{M}{N} \times \#C$ tuples with known $F_0(y)$
Apply 3-round attack on all known $F_0(y)$ (works since $\frac{M}{N} \times \#C > N$ )
Play a yo-yo game on 4-round FN with the results from 3-round attack.
Output: (partial) tables for $F_0F_1F_2F_3$
```

Our attack algorithm has two phase transitions. The first phase transition occurs with enough data to be able to make the graph G and find cycles in it. The second phase transition occurs with the bad edges in the collected cycles. If this happens, we must enrich to be able to collect desired vertices. Since there is a sufficient window in between these two phase transitions, our attack breaks the scheme with good probability of success without overcoming the difficulty with the bad edges in the second phase.

In Table 2, we show the experimental results of success probability of the entire attack for a strategy called S_2. Let S_2 be an event. In S_2, we look at the largest connected component and fail unless it has no bad edges in G'. What we report in Table 2 includes the success probability $\Pr_{\mathsf{succ}}$ of S_2 and we recover the entire tables for each round function.

N	M	#trials	$\Pr[\mathsf{succ}, S_2]$	$(\Pr[S_2])$
2^2	9	3864	3.60%	(88.69%)
2^3	29	5791	29.11%	(78.62%)
2^4	91	6585	49.83%	(73.27%)
2^5	288	6814	62.91%	(71.79%)
2^6	913	6981	73.80%	(77.14%)
2^7	2897	6609	83.10%	(83.83%)
2^8	9196	3154	89.22%	(89.38%)
2^9	29193	212	92.45%	(92.45%)

Table 2. Experiments with $\Pr[S_2]$ and success probability over many trials for $\mathsf{L} = 3$ and $\mathsf{M} \sim \mathsf{N}^{\frac{3}{2}} \left(\frac{\mathsf{N}}{2}\right)^{\frac{1}{2\mathsf{L}}}$

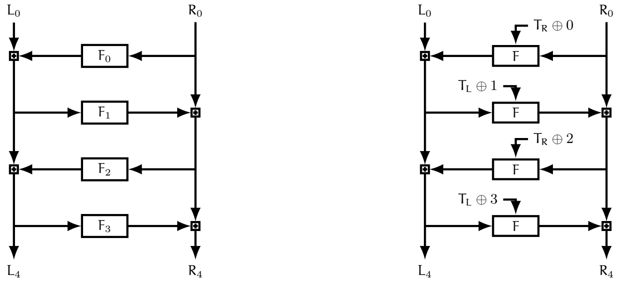
The data complexity of our attack in Algorithm 2 is $\mathsf{M} = \mathsf{O}(\mathsf{N}^{\frac{3}{2}+\frac{1}{2\mathsf{L}}})$. We compute the time complexity for the algorithm based on the step 1, 2, 3, and 4 since the other steps are much shorter than these steps. The complexity is weighted by Step 2, we have the time complexity of our attack as the complexity of finding cycles in G. The cycles of length L in our graph can be found with multiplication on adjacency matrix (which is sparse). Matrix multiplication can be done in $\mathsf{O}(|\mathsf{V}|^2\mathsf{d})$ where $\mathsf{d} = 2\frac{|\mathsf{E}|}{|\mathsf{V}|}$ is the average degree of a vertex. Therefore the complexity is $\mathsf{O}(|\mathsf{V}||\mathsf{E}|)$. With the Floyd-Warshall algorithm, we need $(\mathsf{L}-1)$ multiplications by the adjacency matrix in the max-plus algebra that leads us to a complexity $\mathsf{O}(\mathsf{L}|\mathsf{V}||\mathsf{E}|)$. With $\mathsf{E} \sim \frac{|\mathsf{V}|^2}{\mathsf{N}}$, where $|\mathsf{V}| = 2\frac{\mathsf{M}^2}{\mathsf{N}^2} = 2^{3-\frac{1}{\mathsf{L}}}\mathsf{N}^{1+\frac{1}{\mathsf{L}}}$ and L constant, we have $\mathsf{O}(\frac{|\mathsf{V}|^3}{\mathsf{N}})$ which is equal to $\mathsf{O}(\mathsf{N}^{2+\frac{3}{\mathsf{L}}})$.

The 4-round attacks extend to more rounds by doing exhaustive search on the additional round functions. Interestingly, by changing our attack to a chosen plaintext attack, we do not need to run a full exhaustive search on the first round function.

3 Slide Attack on FF3

3.1 The FF3 Scheme

A Tweakable Format Preserving Encryption (TFPE) is a block cipher that preserves the format of domain in the output. A TFPE function $\mathsf{E} : \mathcal{K} \times \mathcal{T} \times \mathcal{X} \mapsto \mathcal{X}$ is defined from a key space $\mathcal{K}$, a tweak space $\mathcal{T}$, and a domain $\mathcal{X}$ to the same domain $\mathcal{X}$. We are particularly interested in a TFPE scheme by Brier, Peyrin, and Stern [6] whose design is based on Feistel Network depicted in Fig. 2 (b). It is named as FF3 in the NIST standards.



(a) Feistel Network

(b) FF3 Encryption

Fig. 2. 4-round Feistel Network and FF3 Encryption

The FF3 encryption algorithm splits the input X into two substrings L_0 and R_0. For the right half (respectively left half), the algorithm first takes the tweak T_R (respectively T_L) XORed with the encoded round index i and R_i (respectively L_i) to input tweakable PRF F_K. Second, it applies modular addition of the output of F_K to L_i (respectively R_i). The mod operation is necessary to make sure the ciphertext stays in the domain to preserve the format. In concrete proposal round functions are AES encryption with 128 bits. [1].

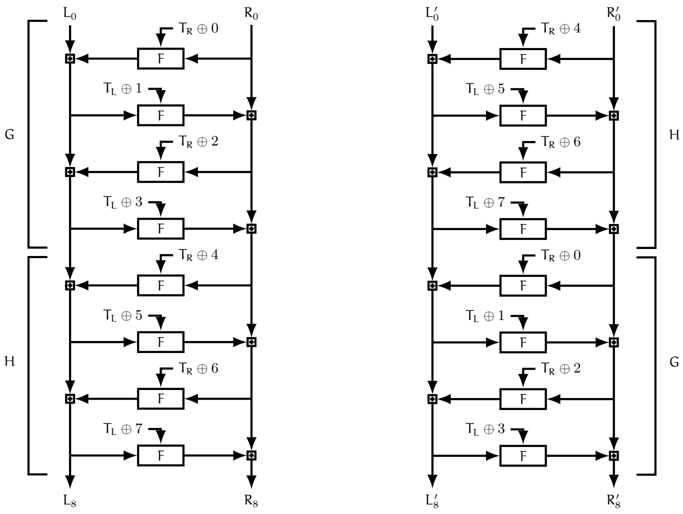
For simplicity and by abuse of notations, we say that FF3 encrypts the plaintext $(\mathsf{L}_0, \mathsf{R}_0)$ into the ciphertext $(\mathsf{L}_w, \mathsf{R}_w)$ with tweak $(\mathsf{T}_\mathsf{L}, \mathsf{T}_\mathsf{R})$. We illustrate the 4-round FF3 scheme in Fig. 2 (b).

3.2 FF3 Attack

We develop an attack on FF3 that aims to reconstruct the entire codebook for a challenge tweak for a number of queries which is lower than the size of the brute force codebook attack. The main idea of the designed FF3 attack takes advantage of the flexibility to change the tweak to permute the round functions.

Consider two functions G and H, where G is a 4-round Feistel scheme using tweakable block cipher F with tweaks $(\mathsf{T}_\mathsf{R} \oplus 0, \mathsf{T}_\mathsf{L} \oplus 1, \mathsf{T}_\mathsf{R} \oplus 2, \mathsf{T}_\mathsf{L} \oplus 3)$ and H is a 4-round Feistel scheme using tweakable block cipher F with tweaks $(\mathsf{T}_\mathsf{R} \oplus 4, \mathsf{T}_\mathsf{L} \oplus$

$5, T_R \oplus 6, T_L \oplus 7$). In Fig. 3, we show two runs of FF3 encryption with tweak $T = T_L || T_R$ in (a) and tweak $T' = T_L \oplus 4 || T_R \oplus 4$ on two distinct plaintext. We observe that $\mathsf{FF3}.\mathsf{Enc}(K, T, \cdot) = H \circ G$ and $\mathsf{FF3}.\mathsf{Enc}(K, T', \cdot) = G \circ H$. With this observation, we desire to form a "cyclic" behavior of plaintext/ciphertext pairs under two FF3 encryption with sliding G and H.



(a) FF3 Encryption

(b) Slided Encryption

Fig. 3. FF3 Encryption with Sliding Round Functions

More precisely, we take two parameters α and β to be optimized and consider two sets of messages of size N^{α} picked at random $X = \{x_{10}, , \ldots, x_{i0}, \ldots x_{N^{\alpha}0}\}$ and $X' = \{x'_{10}, \ldots x'_{i0}, \ldots x'_{N^{\alpha}0}\}$. For each message x_{i0} in X, set $x_{i(j+1)} = \mathsf{Enc}(K, T, x_{ij})$ with a fixed tweak $T \in \mathcal{T}$ and a fixed key $K \in \mathcal{K}$. We repeat the chain encryption of outputs N_{β} times for each message in X. Let XC be the set of chain encryption of elements of X. It is a segment of length N^{β} of a cycle of $H \circ G$. Similarly, for each message x'_{i0} in X', set $x'_{i(j+1)} = \mathsf{Enc}(K, T', x'_{ij})$ with the fixed tweak $T' \in \mathcal{T}$ under the same key K. Let $X'C$ be the set of chain encryption of elements of X'. Apparently, we have $|XC| = N^{\alpha}N^{\beta}$ and $|X'C| = N^{\alpha}N^{\beta}$. Given these 2 sets XC and $X'C$, we attempt to find a collision between XC and $X'C$ such that $G(x_{ij}) = x'_{i'0}$ or $G(x_{i0}) = x'_{i'j'}$ for $1 \leqslant i, i' \leqslant N^{\alpha}$ and $1 \leqslant j, j' \leqslant N^{\beta}$. Upon having a table with inputs to G and H, we can apply the known plaintext recovery attack on 4-round Feistel Networks. The concrete algorithm to collect plaintext/ciphertext pairs is given in Algorithm 3.

Algorithm 3: FF3 Attack

```
Input : a tweak bit string $T$ such that $|T| = 64$, a key $K$
$T' \leftarrow T_L \oplus 4 \| T_R \oplus 4$
foreach $i = 1 \cdots N^\alpha$ do
    pick $x_{i0}$ and set $x_{ij} = \mathsf{FPE.Enc}^T(x_{i(j-1)})$ for $j = 1, \dots N^\beta$
    pick $x'_{i0}$ and set $x'_{ij} = \mathsf{FPE.Enc}^{T'}(x_{i(j-1)})$ for $j = 1, \dots N^\beta$
end
foreach $i, i' = 1 \cdots N^\alpha$ do
    foreach $j = 0 \cdots N^\beta - M - 1$ do
        assume that $G(x_{ij}) = x'_{i'0}$
        run 4-round attack on $G$ with $G(x_{i(j+k)}) = x'_{i'k}$ for $k = 0 \cdots N^\beta - j$
        if succeeded, run attack on $H$ with samples $H(G(x_{ik})) = x_{i(k+1)}$
          for $k = 0 \cdots N^\beta - 1$
    end
    foreach $j = 0 \cdots N^\beta - M - 1$ do
        assume that $G(x_{i0}) = x'_{i'j'}$
        run 4-round attack on $G$ with samples $G(x_{ik}) = x'_{i'(j+k)}$ for
          $k = 0 \cdots N^\beta - j$
        if succeeded, run attack on $H$ with samples $H(G(x_{ik})) = x_{i(k+1)}$
          for $k = 0 \cdots N^\beta - 1$
    end
end
```

Our attack has $2N^{\alpha+\beta}$ data complexity. The time complexity is $N^{2\alpha+\beta}$ times the complexity of 4-round recovery attack on Feistel Network. To minimize the data complexity $2N^{\alpha+\beta}$, we want $\alpha + \beta$ to be minimal such that $2\alpha + \beta = 2$ and $N^\beta \geqslant M$. Now, α can be parameterized by β and we minimize $\frac{\beta+2}{2}$. When we set $N^\beta = 2M$, then $N^\alpha = \frac{N}{\sqrt{2M}}$. Therefore, we have data complexity of FF3 attack as $2N\sqrt{M}$ and time complexity as N^2 times the complexity of 4-round recovery attack on Feistel Network and $p_{success} \approx 1 - e^{-p^{Feistel}_{success}}$.

We fully implemented the attack but to test its success probability we could skip some parts of the running time we knew the attack would fail. We show on Table 3 the experimental probability of success of the whole attack following the strategies S_2. The probability was computed for 10,000 executions.

We conclude that the full attack succeeds with high probability using our parameters.

4 Conclusion

We took the NIST standard FF3 and investigated its security on small domain sizes. We exploit that the round functions can be permuted due to a bad domain separation in the tweak scheme which uses a XOR with the round index. This bad design choice lead us to develop a slide attack on FF3 based on our own design for 4-round FN attack.

N	M	N^{α}	N^{β}	#trials	Pr[succ]
2	3	1	6	10000	0.00%
2^2	9	1	18	10000	1.40%
2^3	29	2	58	10000	17.99%
2^4	91	2	182	10000	35.35%
2^5	288	2	576	10000	45.89%
2^6	913	2	1826	10000	54.14%
2^7	2897	2	5794	10000	56.85%
2^8	9196	2	18392	5098	56.34%
2^9	29193	3	58386	256	77.73%

Table 3. Experimental success probability in the FF3 attack for various parameters for $\mathsf{L} = 3$ and $\mathsf{M} \sim \mathsf{N}^{\frac{3}{2}} \left(\frac{\mathsf{N}}{2}\right)^{\frac{1}{2\mathsf{L}}}$

This work shows that Feistel schemes with small domains has not well understood yet. We showed a known plaintext attack on 4-round Feistel structure to recover entire round functions. More attacks for bigger rounds are possible in small domains.

Acknowledgments. The work was done while the first author was visiting EPFL. The first author was supported in part by NSF grant CNS-1453132. We thank Adi Shamir for the useful comments and Stefano Tessaro for the discussions.

References

1. *Recommendation for Block Cipher Modes of Operation: Methods for Format Preserving Encryption.* National Institute of Standards and Technology, 2016.
2. Mihir Bellare, Viet Tung Hoang, and Stefano Tessaro. Message-recovery attacks on Feistel-based Format Preserving Encryption. In *23th CCS Proceedings*, 2016.
3. Mihir Bellare, Thomas Ristenpart, Phillip Rogaway, and Till Stegers. Format-preserving encryption. In Michael J. Jacobson, Vincent Rijmen, and Reihaneh Safavi-Naini, editors, *Selected Areas in Cryptography: 16th Annual International Workshop, SAC 2009, Calgary, Alberta, Canada, August 13-14, 2009, Revised Selected Papers*, volume 5867, pages 295–312. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009.
4. Mihir Bellare, Phillip Rogaway, and Terence Spies. The FFX mode of operation for format-preserving encryption. Draft 1.1. Submission to NIST, Feb. 2010. `http://csrc.nist.gov/groups/ST/toolkit/BCM/documents/proposedmodes/ffx/ffx-spec.pdf`.
5. Alex Biryukov, Gaëtan Leurent, and Léo Perrin. Cryptanalysis of feistel networks with secret round functions. In Orr Dunkelman and Liam Keliher, editors, *Selected Areas in Cryptography - SAC 2015: 22nd International Conference, Sackville, NB, Canada, August 12-14, 2015, Revised Selected Papers*, volume 9566, pages 102–121. Springer International Publishing, 2016.

6. Eric Brier, Thomas Peyrin, and Jacques Stern. BPS: a Format-Preserving Encryption Proposal. http://csrc.nist.gov/groups/ST/toolkit/BCM/documents/proposedmodes/bps/bps-spec.pdf.
7. Michael Brightwell and Harry E. Smith. Using Datatype-Preserving Encryption To Enchance Data Warehouse Security. Available at: http://csrc.nist.gov/nissc/1997/proceedings/141.pdf, 1997.
8. Michael Luby and Charles Rackoff. How to construct pseudorandom permutations from pseudorandom functions. *SIAM J. Comput.*, 17(2):373–386, April 1988.
9. Jacques Patarin. Security of balanced and unbalanced feistel schemes with linear non equalities. http://eprint.iacr.org/2010/293, 2010.
10. Terence Spies. Format preserving encryption. Unpublished white paper, available at: https://www.voltage.com/wp-content/uploads/Voltage-Security-WhitePaper-Format-Preserving-Encryption.pdf, 2008.