Quantum Cryptanalysis of Farfalle and (Generalised) Feistel Network

S. Hodžić, A. Roy, and E. Andreeva

1 University of Primorska, FAMNIT, Koper, Slovenia, samir.hodzic@famnit.upr.si
2 University of Klagenfurt, Austria, arnab.roy@aau.at
3 Technical University of Vienna, Austria, elena.andreeva@tuwien.ac.at

Abstract. Farfalle is a permutation-based pseudo-random function which has been proposed by G. Bertoni et al. in 2017. In this work, we show that by observing suitable inputs to Farfalle, one can derive various constructions of a periodic function with a period that involves a secret key. As this admits the application of Simon’s algorithm in the so-called Q2 attack model, we further show that in the case when internal rolling function is linear, then the secret key can be extracted under feasible assumptions. Furthermore, using the provided constructions of periodic functions for Farfalle, we show that one can mount forgery attacks on the session-supporting mode for authenticated encryption (Farfalle-SAE) and the synthetic initial value AE mode (Farfalle-SIV). In addition, as the wide block cipher mode Farfalle-WBC is a 4-round Feistel scheme, a quantum distinguisher is constructed in the case when input branches are containing at last two blocks, where length of one block corresponds to the size of a permutation employed in Farfalle (a similar attack can be mounted to Farfalle-WBC-AE). And finally, we consider the problem of extracting a secret round key out of different periods obtained from a (Generalized) Feistel scheme (GFN), which has not been addressed in any of the previous works which consider the application of Simon’s (or Simon-Grover) algorithm to round reduced versions of GFNs. In this part, we assume that the key is added to an input of an inner function utilized in the round function of a given GFN. By applying two different interpolation formulas, we show that one can extract the round key by utilizing amount of different periods which is closely related to the polynomial/algebraic degree of underlying inner function.

Keywords: Simon’s algorithm, Farfalle, Quantum cryptanalysis, Lagrange’s interpolation.

1 Introduction

The application of Simon’s algorithm in post-quantum cryptanalysis of block ciphers has been firstly demonstrated by H. Kuwakado and M. Morii in 2010, where a quantum distinguisher for the 3-round Feistel network has been constructed. Since then, many works have been applying Simon’s algorithm in order to re-analyse the security of block ciphers and various modes on top of
them in the so-called post-quantum setting. In this context, Simon’s algorithm has been usually combined with Grover’s [14] algorithm, an approach introduced by G. Leander and A. May in [22], and it has been shown that many symmetric cryptographic schemes and general block cipher constructions provide much less security than expected. Such examples are the Even-Mansour [20] scheme, the AEZ [2] and LED ciphers [27], the FX construction [4,22], a number of authenticated encryption schemes [3,18,28], tweakable enciphering schemes [12], to name a few. In addition, many recent works such as [7, 12, 16, 29, 30] (and references therein), deal with post-quantum cryptanalysis on reduced-round versions of Generalized Feistel schemes by applying Simon’s algorithm or the Simon-Grover combination [22].

The main idea behind an attack based on the Simon’s algorithm is to construct a periodic function $f$ using the elements of the underlying cryptographic scheme, such that the values of $f$ are available to the adversary via an oracle (as a black-box). The security model in which the adversary has access only to classical encryption (from which the values of $f$ are obtained) or decryption queries to the oracle, is known as the so-called $Q_1$ attack model. In the $Q_2$ model, the adversary has the quantum access to the encryption oracle (i.e. superposition access). Usually, the function $f$ is constructed in such a way that its period is either the secret key (as in the case for the Even-Mansour scheme), or some other unknown value (depending on the underlying cipher), in which case $f$ is used as a quantum distinguisher due to its periodicity property. Note that the periodicity of a random permutation is not expected with very high probability. If $f$ has only one non-trivial period, then Simon’s algorithm finds the period of $f$ in quantum polynomial time $O(n)$ ($n$ is the dimension of the domain of $f$), which is significantly less than the classical computational complexity $O(2^{n/2})$ that would be required. Whenever the construction of $f$ gives rise to unwanted periods (under the assumption that there are not many periods), one can apply the results by M. Kaplan et al. [18] in order to recover the wanted period.

Contributions

In this work, we apply the Simon’s algorithm under the $Q_2$ attack model to the pseudorandom function Farfalle and its modes [5]: a session-supporting mode for authenticated encryption (Farfalle-SAE), a synthetic initial value AE mode (Farfalle-SIV) and a wide block cipher mode (Farfalle-WBC). The main weakness which admits the application of Simon’s algorithm to these schemes is the structure of an internal sum which is a periodic function with respect to the given input blocks. Based on this observation, we firstly show that one can provide different constructions of a periodic function with a period which contains a secret key (Section 3). More precisely, we firstly show that one can consider a message of two blocks only (both blocks are equal and represent a variable), and construct a periodic function via Construction 1 (Section 3.1). Then, this construction can easily be extended by any number of blocks, where one fixes the two blocks which are equal and represent a variable, and the rest of the blocks are constant. Another approach is to consider a message with more than two blocks,
such that it has multiple pairs of equal blocks and represent different variables (eventually a constant can be added to these pairs). These two construction methods are given by Construction 2 (Section 3.1). In general, the design of Farfalle suggests that one can utilize lightweight permutations internally (the roll_c permutation, cf. Figure 1), and these are taking a secret value as an input. In the special case when the permutation roll_c is a linear mapping (such as in Kravatte, cf. [5, Section 7]), we show (Section 3.2) that one can extract the secret key under certain reasonable assumptions, which are essentially satisfied when a sum of defining matrices of different powers of roll_c is an invertible mapping (or if it has a larger rank, then it may significantly reduce the key space).

Furthermore, in Section 3.3 we show that Constructions 1 and 2 can be used to mount forgery attacks on the Farfalle modes SAE, SIV and WBC. More precisely, for the Farfalle-SAE mode we demonstrate that a forgery can be done in the case when the length of the plaintext is zero, in which case one only manipulates with metadata/associated data blocks. Regarding the Farfalle-SIV mode, the forgery is possible by manipulating both metadata and/or plaintext. And finally, as the Farfalle-WBC (similarly goes for Farfalle-WBC-AE) scheme is a 4-round Feistel scheme, then a distinguishing attack can be mounted based on the fact that the input branches may contain at least two blocks (whose length is as the size of the permutation p_c in Farfalle). Here, we assume that the PRFs in Farfalle-WBC are taken to be the Farfalle pseudorandom functions.

We conclude this paper by Section 4 in which we analyse the period obtained from reduced-round versions of General Feistel networks (GFNs) in many recent papers. Best to our knowledge, the extraction of a secret round key from obtained period in GFNs has not been addressed so far. In this context, assuming that inner function F_k of a given GFN is defined as F_k(x) = F(x ⊕ k) (x is an input block, k is a round key, F is a publicly known function), then we firstly show that one can utilize the Lagrange’s interpolation formula (e.g., see [24, Subsection 2.1.7.3]) in order to extract a secret key. In addition, we show that one can use also another interpolation formula which requires the knowledge of output values given at certain specific inputs. Overall, both interpolation methods require multiple application of the attack (either Simon’s algorithm only, or Simon-Grover combination) by which one obtains different periods (which are used for the interpolation process). The number of this applications strongly depends on the polynomial/algebraic degree of F (depending on the method), which is usually very small in most of the existing schemes, and thus makes the extraction quite feasible. At this point, we note that the Simon-Grover combination is an algorithm which has been firstly introduced by G. Leander and A. May [22], and since then it has been applied to various GFNs for certain round-reduced versions which do not admit the application of Simon’s algorithm only.

Outline

The article is organized as follows. In Section 2 we provide an overview of Simon’s algorithm along with some basic notation. In Section 3.1 we demonstrate the constructions of periodic functions to Farfalle: Constructions 1 and 2. Then,
in Section 3.2 we consider the extraction of a secret key from different Farfalle periods. The application of Constructions 1 and 2 is further utilized in Section 3.3, where forgery attacks are given for Farfalle SAE and SIV, as well as a construction of a quantum distinguisher for the WBC mode. The extraction of a secret round key from different periods obtained from GFNs is shown in Section 4. We give our concluding remarks in Section 5.

2 Preliminaries

The vector space $\mathbb{F}_2^n$ is the space of all $n$-tuples $x = (x_1, \ldots, x_n)$, where $x_i \in \mathbb{F}_2$. The all-zero vector is denoted by $0_n$. A quantum register is a collection of $n$ qubits (the classical basis states $|0\rangle$ and $|1\rangle$), and formally we denote it as $|x\rangle = |c_1\rangle \otimes \cdots \otimes |c_n\rangle$, where $c_i \in \mathbb{F}_2$ (and thus $x \in \mathbb{F}_2^n$). An operator $U_f$, which implements a function $f : \mathbb{F}_2^n \to \mathbb{F}_2^r$ quantumly, uses the all-zero register $|0_r\rangle$ with auxiliary qubits as $U_f : |x\rangle|0_r\rangle \to |x\rangle|0_r \oplus f(x)\rangle = |x\rangle|f(x)\rangle$.

Throughout the paper, when $f$ is a function which uses a secret value (e.g. a secret key), then $U_f$ will denote a quantum oracle (as unitary operator) that provides the values of $f$.

In relation to Simon’s quantum algorithm described later on, we will use the Hadamard transform $H^{\otimes n}$, $n \geq 1$ (also known as the Sylvester-Hadamard matrix), which is defined recursively as

$$H^{\otimes 1} = 2^{-1/2} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}; \quad H^{\otimes n} = 2^{-1/2} \begin{pmatrix} H^{\otimes (n-1)} & H^{\otimes (n-1)} \\ H^{\otimes (n-1)} & -H^{\otimes (n-1)} \end{pmatrix}.$$

Throughout the article, by $|0\rangle$ we will denote the all-zero quantum register whose size will be clear from the context.

2.1 On Simon’s algorithm

Suppose that a Boolean (vectorial) function $f : \mathbb{F}_2^n \to \mathbb{F}_2^r$ ($r \geq 1$) has a unique (secret) period $s$, that is

$$f(x) = f(y) \iff x \oplus y \in \{0_n, x \oplus s\}.$$

This problem is solved efficiently by Simon’s algorithm [26], which extracts $s$ in $O(n)$ quantum oracle queries. Classically, solving this problem requires exponential complexity $O(2^{n/2})$. In applications of this algorithm in post-quantum
cryptanalysis of block ciphers (and its modes), one usually works in an environment in which a given function is not expected to have more than one period due to its complexity, or even if it has more periods, then some other techniques are applied to extract a desired period. For instance, Theorems 1 and 2 given by M. Kaplan et al. in [18] show that one can still extract (with high probability) a desired shift just by performing more queries. On the other hand, if there exist other periods with probability \( \frac{1}{2} \), then one can apply a classical distinguishing attack based on higher order differentials with probability \( \frac{1}{2} \) (cf. [18]).

Thus, in our work we will assume that an observed function does not have many periods in general, while on the other hand, it is known that a random (vectorial) function has periods with negligible probability.

As our work utilizes Simon’s algorithm as a main tool, we recall its computation steps below.

**Simon’s algorithm [26]:**

1) Prepare the state \( 2^{-n/2} \sum_{x \in \mathbb{F}_2^n} |x\rangle |0\rangle \), where the second all-zero register is of size \( \tau \) (\( x \in \mathbb{F}_2^n \)).

2) Apply the operator \( U_f \) which implements the function \( f : \mathbb{F}_2^n \rightarrow \mathbb{F}_2^\tau \) in order to obtain the state \( 2^{-n/2} \sum_{x \in \mathbb{F}_2^n} |x\rangle |f(x)\rangle \).

3) Measuring the second register (the one with values of \( f \)), the previous state is collapsed to \( |\Omega_a\rangle = 2^{-n/2} \sum_{x \in \Omega_a} (-1)^{x \cdot y} |y\rangle \), where \( \Omega_a = \{ x \in \mathbb{F}_2^n : f(x) = a \} \), for some \( a \in \text{Im}(f) \).

4) Apply the Hadamard transform \( H^\otimes n \) to the state \( |\Omega_a\rangle = 2^{-n/2} \sum_{x \in \Omega_a} (-1)^{x \cdot y} |y\rangle \) in order to obtain \( |\varphi\rangle = 2^{-n/2} \sum_{y \in \mathbb{F}_2^n} \sum_{x \in \Omega_a} (-1)^{x \cdot y} |y\rangle \).

5) Measure the state \( |\varphi\rangle \):
   
   i) If \( f \) does not have any period, the output of the measurement are random values \( y \in \mathbb{F}_2^n \).
   
   ii) If \( f \) has a period \( s \), the output of measurement are vectors \( y \) which are strictly orthogonal to \( s \), since the amplitudes of \( y \) are given by \( 2^{-(n+1)/2} \sum_{x \in \Omega_a} (-1)^{x \cdot y} = 2^{-(n+1)/2}((-1)^{y \cdot s} + (-1)^{y \cdot (s^\otimes n) \cdot y}) \), where the term \( 2^{-(n+1)/2} \) comes from the assumption that \( |\Omega_a| = 2 \) for any \( a \in \mathbb{F}_2^n \).

6) If \( f \) has a period, repeat the previous steps until one collects \( n-1 \) linearly independent vectors \( y_i \). Then, solve the homogeneous system of equations \( y_i \cdot s = 0 \) (for collected values \( y_i \)) in order to extract the unique period \( s \).

**Remark 1.** The last step above has complexity \( O(n^3) \), which stands for the Gaussian elimination procedure (used for solving linear systems). For complexity estimates throughout the paper, it will be neglected. Also, by Simon’s function we will call a periodic function which is constructed for an observed scheme, whether it has one or more periods. Without explicitly stating it, whenever we construct a periodic function, it will be clear that one can apply Simon’s algorithm in order to obtain its period(s) (where we assume that observed scheme does not have many periods, as discussed earlier).

A common technique to construct a Simon’s function \( f \) (used in many works) is to concatenate two suitable functions as shown below.
Proposition 1. Let the function $f : \mathbb{F}_2 \times \mathbb{F}_2^m \rightarrow \mathbb{F}_2^n$ be defined as

$$f(b, x) = \begin{cases} g(x), & b = 0 \\ h(x), & b = 1 \end{cases},$$

(1)

where $g, h : \mathbb{F}_2^n \rightarrow \mathbb{F}_2^m$. Then, if $h(x \oplus s) = g(x)$ holds for all $x \in \mathbb{F}_2^n$, then $f$ has period $(1, s) \in \mathbb{F}_2 \times \mathbb{F}_2^n$.

Remark 2. Note that if $g$ and $h$ in (1) are permutations, then using the same arguments as in the proof of Lemma 1 in [19], one can show that $(1, s)$ is an unique period of $f$.

In what follows we recall two examples of constructions of $f$ provided in previous works, which are special cases of Proposition 1.

Example 1. In [19] authors analysed the distinguishability of 3-round Feistel cipher, and in Section III-A defined the function $f : \mathbb{F}_2 \times \mathbb{F}_2^m \rightarrow \mathbb{F}_2^n$ by

$$f(b, x) = \begin{cases} F_2(x \oplus F_1(\alpha)) \oplus (\alpha \oplus \beta), & b = 0 \\ F_2(x \oplus F_1(\beta)) \oplus (\alpha \oplus \beta), & b = 1 \end{cases},$$

(2)

where $\alpha, \beta \in \mathbb{F}_2^m$ are different fixed vectors. Denoting by $g(x) = f(0, x)$ and $h(x) = f(1, x)$, it is easily verified that $h(x \oplus (F_1(\alpha) \oplus F_1(\beta))) = g(x)$, and thus $f$ has a linear structure $(1, s) = (1, F_1(\alpha) \oplus F_1(\beta)) \in \mathbb{F}_2 \times \mathbb{F}_2^n$, and thus this construction corresponds to the case (ii) of Proposition 1.

Remark 3. Note that there exist several other works who use the similar ideas to construct the Simon’s function $f$ as above (for instance, [9, Section 3.2], [18, Section 5.1], [27], etc.).

On the other hand, there are constructions of the function $f$ which are not based on concatenation method. For instance, in [20] the Even-Mansour cipher has been broken (in the setting which uses two keys), which is given by $E_{k_1, k_2}(x) = P(x \oplus k_1) \oplus k_2$, by constructing the following function $f : \mathbb{F}_2^n \rightarrow \mathbb{F}_2^n$ by

$$f(x) = E_{k_1, k_2}(x) \oplus P(x) = P(x \oplus k_1) \oplus P(x) \oplus k_2.$$

One can easily verify that $f(x \oplus k_1) = f(x)$ holds for all $x \in \mathbb{F}_2^n$ (i.e., we have that $s = k_1$). Further analysis and recovering of the keys $k_1$ and $k_2$ one can find in [18, Section 3.2].

Another example is LRW construction [23] analysed in [18, Section 3.2], which is defined as $E_{t, k}(x) = E_k(x \oplus d(t)) \oplus d(t)$, where $d$ is a universal hash function. Constructing the function $f : \mathbb{F}_2^n \rightarrow \mathbb{F}_2^n$ by

$$f(x) = E_{k}(x \oplus d(t_0)) \oplus d(t_0) \oplus E_k(x \oplus d(t_1)) \oplus d(t_1),$$

where $t_0, t_1$ are two arbitrary distinct tweaks, we have that $f$ is periodic in $s = d(t_0) \oplus d(t_1)$. 
Remark 4. In Section 3, we will be mainly utilizing the construction approach similar to the one used for the LRW scheme, as the pseudorandom function Farfalle internally admits the same weakness. Essentially, we see that the LRW construction admits the swapping of (secret) terms present with the variable $x$ (that is $d(t_0)$ and $d(t_1)$). Note that the same construction is used for other schemes (e.g. PMAC, OCB, etc.), see for instance [18].

3 Applying Simon’s algorithm to Farfalle

Farfalle is a permutation-based construction for building a pseudorandom function (PRF) which has been introduced by G. Bertoni et al. [5] in 2017. An instance of this construction known as Kravatte was presented in [5], which is based on Keccak-$p[1600, n_r]$ permutations. The construction of Farfalle is given in Figure 1, while the precise description of the scheme is given in algorithm 1.

In general, the construction of Farfalle employs (a lightweight) $b$-bit permutations $p_b, p_c, p_d$ and $p_e$. In addition, the rolling functions $roll^i_c$ and $roll^i_e$ (also permutations) are used for compression and expansion respectively, where by construction these two functions should be chosen such that an adversary not knowing the value $k$ shall not be able to predict the mask values $roll^i_c(k)$ for any $i$ (in a reasonable range), nor the difference between any pair of mask values $roll^i_c(k) \oplus roll^j_e(k)$ for any $i \neq j$. In what follows, we consider different cases in which one can construct a periodic function (to which Simon’s algorithm applies), which in our case will be output blocks $z_j$.

Remark 5. For convenience, in our constructions we will assume that all message blocks $m_i$ are directly corresponding to $roll^i_c$ functions in Farfalle, i.e. there are

![Fig. 1: The Farfalle construction.](image-url)
Algorithm 1 The Farfalle algorithm. [5]

Parameters: \(b\)-bit permutations \(p_b, p_c, p_d, p_e\) and rolling functions \(\text{roll}_c, \text{roll}_e\)

Input:
- key \(K \in \mathbb{Z}_2^b, |K| \leq b - 1\)
- input string sequence \(M^{(m-1)} \circ \ldots \circ M^{(0)} \in (\mathbb{Z}_2^*)^+\)
- length \(n \in \mathbb{N}\) and offset \(q \in \mathbb{N}\)

Output: string \(Z \in \mathbb{Z}_2^n\)

\[
K' \leftarrow \text{pad}10^b(K)
\]
\[
k \leftarrow p_b(K')
\]
\[
x \leftarrow 0^b, I \leftarrow 0
\]

\[
\text{for } 0 \ldots m - 1 \text{ do}
\]
\[
M \leftarrow \text{pad}10^b(M^{(j)})
\]
\[
\text{Split } M \text{ in } b\text{-bit blocks } m_I \text{ to } m_{I+\mu-1}
\]
\[
x \leftarrow x + \sum_{i=1}^{I+\mu-1} p_c(m_i + \text{roll}_c^i(k))
\]
\[
I \leftarrow I + \mu + 1
\]
\[
\text{end for}
\]
\[
k' \leftarrow \text{roll}_e^i(k), y \leftarrow p_d(x)
\]
\[
\text{while all the requested } n \text{ bits are not yet produced do}
\]
\[
\text{produce } b\text{-bit blocks as } z_j = p_e(\text{roll}_e^i(y)) + k'
\]
\[
\text{end while}
\]
\[
Z \leftarrow n \text{ successive bits from } z_0||z_1||z_2\ldots \text{ starting from bit with index } q
\]
\[
\text{return } Z = 0^n + F(M^{(m-1)} \circ \ldots \circ M^{(0)}) \ll q
\]

no blank indices which are not contributing to accumulators (cf. [5 Figure 2]). Note that this assumption does not affect our construction methods in general, since the messages could always be chosen suitably so that they yield periodicity of output blocks \(z_j\).

3.1 Constructing periodic functions

We present the following constructions:

Construction 1: a) Let \(M\) be a message which contains only two \(b\)-bit blocks \(m_i\), namely \(M = m_0||m_1\). Following the notation of the Farfalle algorithm, we have that the internal value (denoted by \(Y\)) is given as

\[
Y = p_d[p_c(m_0 \oplus \text{roll}_c^0(k)) \oplus p_c(m_1 \oplus \text{roll}_c^1(k))].
\]

Now, if we set the blocks \(m_0\) and \(m_1\) to be equal and represent the same variable, that is \(m_0 = m_1 = m \in \mathbb{F}_2^b\) (note that \(p_c : \mathbb{F}_2^b \rightarrow \mathbb{F}_2^b\)), we have that any output block \(z_j\) \((j \geq 0)\), is given as

\[
z_j(m) = p_e(\text{roll}_c^j(Y)) \oplus k' = p_e(\text{roll}_c^j(p_d[p_c(m \oplus \text{roll}_c^0(k)) \oplus p_c(m \oplus \text{roll}_c^1(k))]) \oplus k',
\]

where \(k'\) is a constant since it depends on the constant \(k\) and the number of input blocks (which is related to \(i\)). We note that a value \(j\) can be chosen
arbitrarily, which falls into the range of the first $q$ bits in the last step of the Farfalle algorithm.

Now, we observe that the function $z_j = z_j(m)$ is periodic with the period $s = \text{roll}_c^0(k) \oplus \text{roll}_c^1(k)$, i.e. it holds that

$$z_j(m \oplus \text{roll}_c^0(k) \oplus \text{roll}_c^1(k)) = z_j(m), \quad \forall m \in \mathbb{F}_2^b.$$ Consequently, by applying Simon’s algorithm to $z_j(m)$ one is able to extract the value $s$ in quantum polynomial time $\mathcal{O}(b)$. We note that $s$ is not expected to be equal to 0, due to the design criterions for the $\text{roll}_c$ permutation.

**b)** Similarly, we consider the message $M = (m \oplus \alpha)|(m \oplus \beta)$ with $m$ being a variable and arbitrary fixed values $\alpha, \beta \in \mathbb{F}_2^b$, in which case any output function $z_j(m)$ is given as

$$z_j(m) = p_c(\text{roll}_c^1(p_d[p_c(m \oplus \alpha \oplus \text{roll}_c^0(k)) \oplus p_c(m \oplus \beta \oplus \text{roll}_c^1(k))]) \oplus k'.$$

Clearly, $z_j(m)$ has the period $\alpha \oplus \beta \oplus \text{roll}_c^0(k) \oplus \text{roll}_c^1(k)$, which can be extracted by Simon’s algorithm in polynomial time $\mathcal{O}(b)$.

**Construction 2:** The previous construction can be extended to the case when the message $M$ contains more blocks. Consider the following cases:

**i)** For instance, one may take two blocks to be equal and represent a variable, while the remaining blocks are all arbitrary constants. More precisely, such a message is given by

$$M = m_0 || m_1 || \alpha_1 || \ldots || \alpha_t = m || m || \alpha_1 || \ldots || \alpha_t, \quad m \in \mathbb{F}_2^b,$$

where $\alpha_u \in \mathbb{F}_2^b$ are arbitrary constants. In this case, the internal value $Y$ is given as

$$Y = p_d(p_c(m \oplus \text{roll}_c^0(k)) \oplus p_c(m \oplus \text{roll}_c^1(k)) \oplus \bigoplus_{i=0}^t p_c(\alpha_i \oplus \text{roll}_c^i(k))),$$

which again implies the periodicity of the output function (block) $z_j(m)$ (for any $j$), with the period $\text{roll}_c^0(k) \oplus \text{roll}_c^1(k)$. This holds due to the fact that the sum $\bigoplus_{i=0}^t p_c(\alpha_i \oplus \text{roll}_c^i(k))$ is constant and does not affect the periodicity of $z_j(m)$.

**ii)** Another approach here is to consider a message given as

$$M = (m_0 \oplus \alpha_0)||(m_0 \oplus \alpha_0)||(m_1 \oplus \alpha_1)||(m_1 \oplus \alpha_1),$$

where $m_r \in \mathbb{F}_2^b$ are variables and $\alpha_r \in \mathbb{F}_2^b$ ($r = 0, 1$) are arbitrary constants. In this case, due to the periodicity of values

$$Y_j = p_c(m_r \oplus \alpha_r \oplus \text{roll}_c^{0+2r}(k)) \oplus p_c(m_r \oplus \alpha_r \oplus \text{roll}_c^{1+2r}(k)), \quad r = 0, 1,$$
we have that the output function
\[ z_j(m_0, m_1) = p_e(\text{roll}_j_2(Y_0 \oplus Y_1)) \oplus k' \]
is periodic, with periods \((\text{roll}_{i_0}^0(k) \oplus \text{roll}_{i_1}^1(k), 0_b)\) and \((0_b, \text{roll}_{i_0}^2(k) \oplus \text{roll}_{i_1}^3(k))\), i.e. \(z_j(m_0, m_1)\) is periodic in both arguments.

**Remark 6.** Clearly, the construction above can be generalized by taking more than two distinct variables which will be present two times in the message (in \(M\) given above we have two times \(m_0\) and two times \(m_1\)).

The demonstrated constructions show that:

- The periodicity of functions \(z_j(m)\) in all previous constructions does not depend on the choice of rolling functions \(\text{roll}_c\) and \(\text{roll}_e\), nor on the choice of permutations \(p_b, p_c, p_d\) and \(p_e\).
- One can choose a suitable message block \(M = \alpha \oplus m \oplus \beta \oplus m\) (\(\alpha, \beta\) fixed constants of certain lengths) which contains the variable block \(m\) placed such that it corresponds to two different indices, say \(i\) and \(j\) \((i < j)\), in which case the period of any output function \(z_t(m)\) \((t \geq 0)\) is given as \(s = \text{roll}_{i_0}^0(k) \oplus \text{roll}_{i_1}^1(k)\). This fact plays an important role in the process of the key extraction (as shown in Section 3.2).

### 3.2 Extracting the secret key \(K\) in Farfalle

As discussed in [6], the permutation \(p_e\) can be instantiated by some lightweight functions (say, with degree equal to 2). In this context, its resistance to higher-order differential attacks has been discussed in [5, Section 8.1], where it has been noted that whenever an adversary wants to construct four message blocks \(m_{i_1}, m_{i_2}, m_{i_3}, m_{i_4}\) such that \(m_{i_1} \oplus m_{i_2} \oplus m_{i_3} \oplus m_{i_4} = 0\), then the difficulty of it relies on the property that the righthand side of the equality

\[ m_{i_1} \oplus m_{i_2} \oplus m_{i_3} \oplus m_{i_4} = \text{roll}_{i_1}^i(k) \oplus \text{roll}_{i_2}^j(k) \oplus \text{roll}_{i_3}^k(k) \oplus \text{roll}_{i_4}^l(k) \]

should not result in a linear mapping whose defining matrix has a low degree.

More precisely, considering that \(\text{roll}\) is a linear function defined as \(\text{roll}(k) = M \times k\), the righthand side can be written as \((M^{i_1} \oplus M^{i_2} \oplus M^{i_3} \oplus M^{i_4}) \times k\), and thus guessing its value becomes more difficult if the matrix \(M^{i_1} \oplus M^{i_2} \oplus M^{i_3} \oplus M^{i_4}\) has full degree (or higher degree in general), for any four pairwise indices \(i_1, i_2, i_3, i_4\) in some reasonable range which limits the maximum number of blocks in Farfalle. Consequently, this gives a design requirement for the rolling function \(\text{roll}_c\) in general.

Regarding our **Constructions 1 and 2**, we note that the extraction of the secret value \(k\) is in a trade-off with the previous requirement. Namely, by applying the given constructions, one can deduce the periods of the form

\[ s_{i,j} = \text{roll}_{i_0}^i(k) \oplus \text{roll}_{i_1}^j(k), \quad i \neq j, \]
which corresponds to pairwise distinct variable blocks with indices $i$ and $j$. Assuming that $roll_c$ is an invertible linear function defined as $roll_c(k) = M \times k$, then one can combine different periods $s_{i_t,j_t}$ in order to derive the sum

$$s_{i_1,j_1} \oplus \ldots \oplus s_{i_p,j_p} = \bigoplus_{t=1}^{p} (M^{i_t} \oplus M^{j_t}) \times k, \quad i_t \neq j_t. \quad (3)$$

For instance, one may derive the period $s_{0,1} = roll_0^0(k) \oplus roll_1^1(k)$, or $s_{0,2} = roll_0^0(k) \oplus roll_2^2(k)$, etc. Recall that any period $s_{i_t,j_t}$ is obtained in quantum polynomial time $O(b)$ from Farfalle, by applying Simon’s algorithm. Consequently, we have that $k$ is a solution of the linear system (3), and thus a unique $k$ exists if $\bigoplus_{t=1}^{b} (M^{i_t} \oplus M^{j_t})$ is an invertible matrix. Clearly, the space of possible solutions of (3) may be reduced depending on the rank of the matrix $\bigoplus_{t=1}^{b} (M^{i_t} \oplus M^{j_t})$.

As the function $p_b$ is a permutation in Farfalle, this discussion is formalized in the following result.

**Proposition 2.** Let $s_{i_t,j_t} = roll_{i_t}^0(k) \oplus roll_{j_t}^1(k)$ ($1 \leq i_t < j_t \leq p$) be periods obtained from Farfalle by applying Simon’s algorithm. Suppose that the $roll_c(k)$ is a linear function defined as $roll_c(k) = M \times k$. Then, then secret value $k$ is solution of the linear system of equations (3). Consequently, the secret key $K$ in Farfalle can be determined from $K||10^* = p_b^{-1}(k)$.

**Remark 7.** Note that the complexity of obtaining $p$ periods $s_{i_t,j_t}$ is $O(pb)$ (cf. Remark 1), which clearly depends on the requirement that $\bigoplus_{t=1}^{b} (M^{i_t} \oplus M^{j_t})$ has a full rank. In this context, one may further consider the work [25], which analyses the invertibility of a sum of two nonsingular matrices.

### 3.3 Attacking authenticated encryption modes based on Farfalle

Based on the previously presented constructions we now consider attacks on certain authenticated encryption modes based on Farfalle, namely Farfalle-SAE and Farfalle-SIV. These modes apply a pseudorandom function (PRF) $F$ that is instantiated as the Farfalle PRF. Without mentioning explicitly, the extraction of the secret key $K$ will be possible whenever one meets the requirements of Proposition 2.

**Farfalle-SAE** is a session-supporting authenticated encryption scheme, where the initialization and wrapping steps are defined as in algorithm 2.

Now, following the wrapping procedure of Farfalle-SAE, let us assume that $|A| > 0$ and $|P| = 0$, i.e. we are considering the case when there is no message. Still, in this case we assume that the nonce $N$ changes with every new metadata $A$. In the case when $|A| > 0$ and $|P| = 0$, we have that $history$ updates as $history \leftarrow A||0 \circ history = A||0 \circ F_K(N)$, and thus let us assume that the metadata $A$ is given by

$$A = a_0||a_1 = a||a, \quad a \in \mathbb{F}_2^b.$$
Algorithm 2 Farfalle-SAE\([F, t, \ell]\)

**Parameters:** PRF \(F\), tag length \(t \in \mathbb{N}\), and alignment unit length \(\ell \in \mathbb{N}\)

**Initialization** takes \(K \in \mathbb{Z}_2^*\), nonce \(N \in \mathbb{Z}_2^*\) and returns tag \(T \in \mathbb{Z}_2^t\)

- offset = \(\lfloor \frac{t}{\ell} \rfloor\): smallest multiple of \(\ell\) not smaller than \(t\)
- history ← \(N\)
- \(T ← 0^t + F_K(\text{history})\)

**Return** \(T\)

**Wrap** takes metadata \(A \in \mathbb{Z}_2^*\), plaintext \(P \in \mathbb{Z}_2^*\), returns ciphertext \(C \in \mathbb{Z}_2^{|P|}\) and tag \(T \in \mathbb{Z}_2^t\)

- \(C ← P + F_K(\text{history}) \ll \text{offset}\)
- \(\text{if } |A| > 0 \text{ OR } |P| = 0 \text{ then}\)
  - history ← \(A || 0 \circ \text{history}\)
- \(\text{end if}\)
- \(\text{if } |P| > 0 \text{ then}\)
  - history ← \(C || 1 \circ \text{history}\)
- \(\text{end if}\)
- \(T ← 0^t + F_K(\text{history})\)

**Return** \(C, T\)

where \(b\) is length of the \(p_c\) permutation used in Farfalle. In the last step, we have that the tag \(T\) is updated as

\[ T ← F_K(A || 0 \circ F_K(N)), \]

which is clearly periodic in \(s = \text{roll}_0^0(k) \oplus \text{roll}_1^1(k)\) (cf. Construction 1). This is due to the fact that the period \(s\) does not depend on the “old” value of \(\text{history} = F_K(N)\), i.e. \(T(a) = T(a \oplus s)\) holds for any \(a\) even when the nonce value changes with respect to \(A\). Note that similar scenario is present in other modes as well, see for instance [18].

**Mounting a forgery attack (the case \(|A| > 0\) and \(|P| = 0\):**

I) Query the Farfalle-SAE oracle \(O_{Farfalle-\text{SAE}}\) with \(A = a_0 || a_1 = a||a\) and no message sufficiently many times until a period \(s = \text{roll}_0^0(k) \oplus \text{roll}_1^1(k)\) is extracted by Simon’s algorithm from any block \(z_j(a)\) (for any \(j\)) of the tag function \(T(a) = F_K(A || 0 \circ F_K(N))\). As any \(z_j(a)\) has the same period, one can consider the whole output value of \(T(a)\). This stage requires \(O(b)\) queries, and admits the nonce value \(N\) to be different with every new value of \(a\).

II) For an arbitrary (fixed) \(a \in \mathbb{F}_2^b\), construct a valid tag \(T'\) for the metadata \(A' = (a \oplus s)||((a \oplus s)\).

In the case of the forgery attack, it is clear that the existence of more periods (except those that are expected) simply means new/different forgeries (i.e., it does not affect the success of the attack in a negative way). This is also the case for forgery attacks presented later for the Farfalle-SIV mode.
Farfalle-SIV is an authenticated encryption schemes which can securely encipher different plaintexts with the same key without requiring the overhead of nonce management (algorithm [5]). As it uses the tag computed over the message as a nonce for the encryption function, then the security narrows down to the case when two messages have the same tag. Let us consider the following attack

\begin{algorithm}
\caption{Farfalle-SIV algorithm [5]}
\begin{algorithmic}
\State \textbf{Parameters:} a PRF \text{ } F \text{ and tag length } t \in \mathbb{N}
\State \text{Wrap takes metadata } A \in \mathbb{Z}_2^t \text{ and plaintext } P \in \mathbb{Z}_2^t, \text{ and returns ciphertext } C \in \mathbb{Z}_2^t \text{ and tag } t \in \mathbb{Z}_2^t
\State T \leftarrow 0^t + F_K(P \circ A)
\State C \leftarrow P + F_K(T \circ A)
\Return C, T
\EndState
\State \text{Unwrap takes metadata } A \in \mathbb{Z}_2^t, \text{ ciphertext } C \in \mathbb{Z}_2^t \text{ and tag } T \in \mathbb{Z}_2^t \text{ and returns plaintext } P \in \mathbb{Z}_2^{|C|} \text{ or an error}
\State P \leftarrow C + F_K(T \circ A)
\State T' \leftarrow 0^t + F_K(P \circ A)
\If {T' = T}
\Return P
\Else
\Return error
\EndIf
\end{algorithmic}
\end{algorithm}

which is targeting the tag \( T \).

As it has been shown in \textbf{Constructions 1} and \textbf{2}, one can construct different messages \( P_1 \) and \( P_2 \), as well as the corresponding metadata blocks \( A_1 \) and \( A_2 \) such that \( F(P_1 \circ A_1) = F(P_2 \circ A_2) \). For instance, considering \textbf{Construction 1}, one may choose plaintexts \( P_i \) (\( i = 1, 2 \)) as functions of \( m \in \mathbb{F}_2^t \) such that

\[
P_1 = m||m, \quad A_1 = a||a,
\]

\[
P_2 = (m \oplus s')||(m \oplus s')', \quad A_2 = (a \oplus s'')||(a \oplus s''),
\]

(4)

where \( m, a \in \mathbb{F}_2^t \) are considered to be variables, \( s' = \text{roll}_0^0(k) \oplus \text{roll}_1^1(k) \) and \( s'' = \text{roll}_2^2(k) \oplus \text{roll}_3^3(k) \). Consequently, we have that the internal sums \( S_{P_i \circ A_1} \) and \( S_{P_2 \circ A_2} \) are given as

\[
S_{P_1 \circ A_1} = p_c(m \oplus \text{roll}_c^0(k)) \oplus p_c(m \oplus \text{roll}_c^1(k)) \oplus p_c(a \oplus \text{roll}_c^2(k)) \oplus p_c(a \oplus \text{roll}_c^3(k)),
\]

and

\[
S_{P_2 \circ A_2} = p_c(m \oplus s' \oplus \text{roll}_c^0(k)) \oplus p_c(m \oplus s' \oplus \text{roll}_c^1(k)) \oplus p_c(a \oplus s'' \oplus \text{roll}_c^2(k)) \oplus p_c(a \oplus s'' \oplus \text{roll}_c^3(k)).
\]

Thus, any output block \( z_j(m, a) \) of \( F(P_i \circ A_i) \) (\( i = 1, 2 \)) is given as

\[
z_{P_i \circ A_1}^j(m, a) = p_c(\text{roll}_c^j[p_d(S_{P_i \circ A_1}))] \oplus k' = p_c(\text{roll}_c^j[p_d(S_{P_2 \circ A_2}))] \oplus k' = z_{P_2 \circ A_2}^j(m, a),
\]
since the vectors $s'$ and $s''$ are periods of $z_j^{P_1 \circ A_1}(m, a)$, i.e. it holds that

$$z_j^{P_1 \circ A_1}(m, a) = z_j^{P_1 \circ A_1}(m \oplus s', a \oplus s'') = z_j^{P_2 \circ A_2}(m, a).$$

Moreover, it holds that $z_j^{P_1 \circ A_1}(m, a) = z_j^{P_1 \circ A_1}(m \oplus a, a)$ and $z_j^{P_1 \circ A_1}(m, a) = z_j^{P_1 \circ A_1}(m, a)$, i.e. $z_j^{P_1 \circ A_1}(m, a)$ is periodic in both arguments.

**Remark 8.** It is well known that if a given function has some periods, say $s_1, \ldots, s_r$, then the same function is periodic in any value from the space spanned by $s_1, \ldots, s_r$. In context of the previous construction, the function $z_j^{P_1 \circ A_1}(m, a)$ is periodic in every element of the set $\langle (s', 0_b), (0_b, s'') \rangle = \{0_{2b}, (s', 0_b), (0_b, s''), (s', s'') \}$, where $0_{2b}$ is clearly the trivial period.

**On attack feasibility:** In general, the previous construction of the same tag for different inputs $P_i \circ A_i$ is possible if the attacker knows at least one of the vectors $s'$ or $s''$ (depending on whether we want to manipulate plaintext parts or metadata parts respectively). There exist two possible approaches in recovering $s'$ or $s''$:

*I)* In this approach, let us assume that the input plaintext $P_1$ and $A_1$ have the form given by (1). The attacker may firstly apply Simon’s algorithm to the output function $z_j^{P_1 \circ A_1}(m, a)$, which is periodic in both arguments, and then he may construct an input $P_2 \circ A_2$ (as in (1)) for which the tags $T_{P_1 \circ A_1}$ and $T_{P_2 \circ A_2}$ are equal.

**Mounting a forgery attack ($P$ and $A$ defined as in (4)):**

*I)* Query the Farfalle-SIV oracle $O_{Farfalle-SIV}$ with inputs $P = m||m$ and $A = a||a$ sufficiently many times until the space of periods $\langle (s', 0_b), (0_b, s'') \rangle$ is obtained by Simon’s algorithm from any block $z_j(m, a)$ (for any $j$) of the tag function $T(m, a) = F_K(P \circ A)$.

*II)* For arbitrary (fixed) blocks $m, a \in \mathbb{F}_2^b$, construct a valid tag $T'$ for the input $P' \circ A' = (m \oplus s')||(m \oplus s')||(a \oplus s'')||(a \oplus s'')$.

*II)* Now, if we assume that $P_1 = m_0||m_1$ is any plaintext which consists of two blocks (here $m_0$ and $m_1$ are not necessarily the same), then one may assume that for all such messages the metadata $A_1$ has the form $A_1 = a||a$. Then, any output block $z_j(m_0, m_1, a)$ has the period $(0_b, 0_b, s'') = (0_b, 0_b, roll^2_1(k) \oplus roll^2_2(k))$, i.e. it holds that

$$z_j(m_0, m_1, a) = z_j(m_0, m_1, a \oplus s''), \; \forall a, m_0, m_1 \in \mathbb{F}_2^b.$$

**Mounting a forgery attack (Any $P$ and $A$ is defined as in (4)):**

*I)* Query the Farfalle-SIV oracle $O_{Farfalle-SIV}$ with $A = a||a$ and arbitrary inputs $P = m_0||m_1$ sufficiently many times until the period $s''$ is obtained by Simon’s algorithm from any block $z_j(a)$ (for any $j$) of the tag function $T(a) = F_K(P \circ A)$.
II) For arbitrary (fixed) block $a \in \mathbb{F}_2^2$, construct a valid tag $T'$ for the input $P \circ A' = m_0 || m_1 || (a \oplus s'') || (a \oplus s'')$.

Remark 9. Note that the versatility of the previously given forgery attacks can be achieved by placing variable blocks of $P$ and $A$ to correspond to different indices, which in turn gives different periods (as discussed at the end of Subsection 3.1).

Farfalle-WBC is a tweakable wide block cipher (based on two PRFs) whose construction represents an instantiation of the HHFHFH mode, which was presented in [1]. Essentially, it is a 4-round Feistel scheme which processes an arbitrary-length plaintext (algorithm 4). In order to provide a general analysis of the security of Farfalle-WBC in terms of Simon’s algorithm, for convenience we consider a 4-round Feistel scheme (Figure 2) with the following setting:

- Inner functions $F_i$ ($i = 1, \ldots, 4$) represent $H_K$ and $G_K$ such that $F_1(x) = H_K(x||0)$, $F_4(x) = H_K(x||1)$, $F_2(x) = G_K(x||W||1)$, $F_3(x) = G_K(x||W||0)$, where $W \in \mathbb{F}_2^*$ is a tweak.

\begin{algorithm}
\caption{Farfalle-WBC[H, G, ℓ] \cite{5}}
\textbf{Parameters:} PRFs $H, G$ and alignment unit length $\ell \in \mathbb{N}$
\begin{algorithmic}
  \STATE $P \leftarrow$ first split($|P|$) of $P$, and $R$ gets the remaining bits
  \STATE $R_0 \leftarrow R_0 + H_K(L||0)$, where $R_0$ is the first min($b, |R|$) bits of $R$
  \STATE $L \leftarrow L + G_K(R||1 \circ W)$
  \STATE $R \leftarrow R + G_K(L||0 \circ W)$
  \STATE $L_0 \leftarrow L_0 + H_K(R||1)$ where $L_0$ is the first min($b, |L|$) bits of $L$
  \STATE return $C = L||R$
\end{algorithmic}
\end{algorithm}

Fig. 2: The 4-round Feistel network with keyed inner functions $F_i$. 

Algorithm 4 Farfalle-WBC[H, G, ℓ] \cite{5}
– Inner functions $G_K$ and $H_K$ used in Farfalle-WBC algorithm are taken to be Farfalle functions.
– An attack that we present later assumes that the size of input blocks $P_i$ ($i = 1, 2$) is equal to two, i.e. both branches are containing two $b$-blocks ($b$ is length of the inner permutation $p_c$ in Farfalle). It is important to note that this assumption is based on the definition of the $split[b, \ell]$ function given by [5, Algorithm 4], which admits more than one $b$-bit block per branch (for suitable underlying parameters).

With respect to the previous assumptions, we describe the Farfalle-WBC algorithm in a convenient way. Now, by observing the output value $C_2$, which is given as

$$C_2(P_1, P_2) = P_2 \oplus F_1(P_1) \oplus F_3(P_1 \oplus F_2(P_2 \oplus F_1(P_1))),$$

we can construct a Simon’s function $f$ as follows. Taking that $P_2 = X = m||m$ ($m \in \mathbb{F}_2^b$) is a variable and $P_1 = \alpha \in \mathbb{F}_2^b$ is a fixed constant, we define the function $f : \mathbb{F}_2^b \rightarrow \mathbb{F}_2^\tau$ ($\tau$ is length of the output of $H_K$ or $G_K$ in Farfalle-WBC) as

$$f(m) = P_2 \oplus C_2(\alpha, X) = F_1(\alpha) \oplus F_3(\alpha \oplus F_2(X \oplus F_1(\alpha)))$$

$$= F_1(\alpha) \oplus F_3(\alpha \oplus F_2(m \oplus \beta_1||m \oplus \beta_2)),$$

where $F_1(\alpha) = \beta_1||\beta_2$. We deduce the following:

1) We have that the period of the internal function

$$m \rightarrow F_2(m \oplus \beta_1||m \oplus \beta_2) = G_K(m \oplus \beta_1||m \oplus \beta_2||W||1)$$

is given by $s = \beta_1 \oplus \beta_2 \oplus \text{roll}^2(k) \oplus \text{roll}^3(k)$. Note that the blocks of $P_2$ correspond to indices 2 and 3 in Farfalle. This is due to the fact that every output block of the Farfalle function is periodic with the same period $s$ (visible on both Constructions 1 and 2), and moreover, the tweak value $W$ (considered to be a constant) by Constructions 2-(i) does not affect the value of the period.

2) Consequently, as other parts of the function $f$ are constant (referring to $F_1(\alpha)$ and $\alpha$ inside $F_3$), we have that the function $f$ is periodic in $s$, i.e., for every $m \in \mathbb{F}_2^b$ it holds that $f(m \oplus s) = f(m)$. The value $s$ can be extracted by Simon’s algorithm in quantum polynomial time $O(b)$.

3) Hence, the function $f$ can be used as an efficient quantum distinguisher. In general, the key extraction is possible in the case when the rolling function $\text{roll}_c$ is linear, in which case one applies Proposition 2.

Remark 10. Note that the similar construction and arguments can be applied to Farfalle-WBC-AE (given by [5, Algorithm 6]), where one can manipulate the plaintext and/or metadata blocks (in terms of variables and constants).
4 Extracting a secret value from Simon’s period in GFNs

In many recent papers (for instance, see the references in [7, 12, 16, 29, 30]), the application of Simon’s algorithm or Simon-Grover algorithm to different (Generalized) Feistel schemes provides a period of the form

\[ s = F_k(\alpha) \oplus F_k(\beta) \oplus C, \]

where \( \alpha \) and \( \beta \) are known different constants, \( C \) may be known or unknown constant which does not depend on \( \alpha \) and \( \beta \), and \( k \) is a secret value. For instance, such an \( s \) has been obtained in [19] from the 3-round Feistel scheme (as shown in Example 1 earlier), Type 1, 2 and 3 GFNs in [9, 10, 15–17, 30], etc. Usually, the value \( k \) in considered GFNs is a round key with \( C \) usually being the all-zero vector. So far, no method has been proposed which extracts \( k \). In this section we provide a method for extracting the value \( k \), when the given function \( F_k \) is defined as \( F_k(z) = F(z \oplus k) \) with \( F \) being publicly known function. Clearly, in the case when \( F_k \) is defined as \( F_k(z) = F(z) \oplus k \), then \( s \) does not depend on \( k \) at all (unless if it is involved in \( C \)).

Let us assume that \( F \) is a function in \( n \)-bits, i.e. \( F : \mathbb{F}_2^n \to \mathbb{F}_2^n \), defined as \( F_k(z) = F(z \oplus k) \), \( z \in \mathbb{F}_2^n \). In order to extract the value \( k \) from \( s = F_k(\alpha) \oplus F_k(\beta) \oplus C \), we firstly notice that the value of \( s \) can be viewed as a function in \( \alpha \) and \( \beta \), i.e. we have that \( s = s(\alpha, \beta) \). Recall that by certain amount of applications of Simon’s algorithm (until one obtains \( n - 1 \) linearly independent vectors orthogonal to \( s \)), we obtain only a particular value of \( s \) for given \( \alpha \) and \( \beta \). Clearly, for higher values of \( n \), it is not feasible to obtain all values of \( s(\alpha, \beta) \) due to large input space. As one can choose \( \alpha \) and \( \beta \) to be arbitrary and different, let us assume that

\[ (\alpha, \beta) = (x, x \oplus \sigma), \]

where \( \alpha = x \) and \( \sigma \in \mathbb{F}_2^n \) is a non-zero fixed and known constant. Thus, we are considering the function \( s : \mathbb{F}_2^n \to \mathbb{F}_2^n \) given by

\[ s(x) = F_k(x) \oplus F_k(x \oplus \sigma) \oplus C, \quad x \in \mathbb{F}_2^n. \]

Since \( F \) is publicly known function, let us now consider the function \( \Delta : \mathbb{F}_2^n \to \mathbb{F}_2^n \) defined by

\[ \Delta(x) = s(x) \oplus F(x) \oplus F(x \oplus \sigma) = [F(x \oplus k) \oplus F(x)] \oplus [F(x \oplus \sigma \oplus k) \oplus F(x \oplus \sigma)] \oplus C. \]

It is not difficult that see that the function \( \Delta \) has (at least) the non-trivial periods \( \{k, \sigma, k \oplus \sigma\} \). However, the problem which remains to be solved is how to implement the function \( \Delta \) efficiently in quantum/classical environment, without performing infeasible amount of measurements (or applications of Simon’s algorithm). In order to solve this problem, one may consider the following two approaches.
4.1 Utilizing the Lagrange interpolation formula

One may employ the so-called Lagrange interpolation formula (see e.g. [24, Subsection 2.1.7.3]) which is described as follows. Let \( x_1, \ldots, x_n, y_1, \ldots, y_n \) be elements of a field \( F \), where \( x_i \), for \( i \in [1, n] \), are pairwise distinct. Then there exists a unique polynomial, say \( h : F \to F \), with polynomial degree at most \( n - 1 \) such that \( h(x_i) = y_i, i \in [1, n] \), which is given by

\[
h(x) = \sum_{i=1}^{n} y_i \prod_{1 \leq j \leq n, j \neq i} \frac{x - x_j}{x_i - x_j}.
\]

**Remark 11.** Note that in the case when \( F : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n} \) (\( F = GF(2^n) = \mathbb{F}_{2^n} \)) is given by its univariate representation in \( \mathbb{F}_{2^n} [x] \) as \( F(x) = \sum_{i=0}^{2^n-1} a_i x^i \), then its polynomial degree \( \deg(F) \) is equal to maximal \( i \) for which \( a_i \neq 0 \).

As the vector \( \sigma \) and \( F \) are known, then clearly one has to recover (interpolate) the function \( s(x) \) in order to obtain the function \( \Delta(x) \), since \( \Delta(x) = s(x) \oplus F(x) \oplus F(x \oplus \sigma) \).

In order to apply Lagrange interpolation formula to \( s(x) \), we recall that the degree of any Boolean function \( g : \mathbb{F}_2^n \to \mathbb{F}_2 \) and its first derivative \( D_0g(x) = F(x) \oplus F(x \oplus a) \ (a \in \mathbb{F}_2^n) \) are related by the inequality

\[
\deg(g) \geq \deg(D_0g) + 1.
\]

If the publicly known function \( F \) is of degree \( d \), then \( \deg(s) = d - 1 \) and thus one can interpolate the function \( s \) if we have \((d - 1) + 1 = d\) input-output pairs \((x_i, s(x_i))\), \(i = 1, \ldots, d\). Note that this can be done even without knowing the secret value \( k \). Consequently, by Lagrange Interpolation formula one can recover the function \( \Delta : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n} \), if we have \( d \) input-output pairs \((x_i, s(x_i))\), \(i = 1, \ldots, d\).

Note that in the case when \( F \) is of higher degree, then the interpolation by Lagrange formula may become infeasible due to the large amount of terms that may occur in \( s \). However, in the case of a GFN, the inner function is usually a function with lower degree (e.g. even considering that \( F \) is a double Substitution-Permutation Network), which means that feasible amount of input-output pairs \((x_i, s(x_i))\) is needed in order to interpolate \( \Delta \). The previous discussion is formalized with the following result.

**Proposition 3.** Suppose that \( s(x) = F(x \oplus k) \oplus F(x \oplus \sigma \oplus k) \oplus C \ (s : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}, \ C \) is a constant, \( \sigma \neq 0 \) is known) is a period obtained by applying Simon’s algorithm to some given function (whose inputs involve \( x \)), where \( F : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n} \) is a publicly known function with \( \deg(F) = d \). In addition, assume that the values \( s(x_1), \ldots, s(x_d) \) have been extracted by taking any pairwise different inputs \( x_1, \ldots, x_d \in \mathbb{F}_{2^n} \). Then:

1) One can recover \( s(x) \) without the knowledge of \( k \), i.e.,

\[
s(x) = \sum_{i=1}^{d} s(x_i) \prod_{1 \leq j \leq n, j \neq i} \frac{x - x_j}{x_i - x_j}.
\]
2) By implementing the function $\Delta(x) = s(x) \oplus F(x) \oplus F(x \oplus \sigma)$, one can recover $k$ by Simon’s algorithm.

As Simon’s algorithm has the polynomial complexity in $n$, then the complexity of implementing the function $\Delta$ is feasible only if the degree of $F$ is not large. Also, if $F$ has a multiple periods, then $k$ can be still extracted by performing sufficiently many measurements, cf. [18, Theorems 1 and 2]. On the other hand, by applying Simon’s algorithm sufficiently many times, vectors obtained by measurements will be orthogonal to the space $\{0_n, k, \sigma, k \oplus \sigma\}$. In the case that $\Delta$ has not other periods than $\{k, \sigma, k \oplus \sigma\}$, which is highly expected, then the value $k$ can be deduced easily as the vector $\sigma$ is known. We conclude this subsection with the following two remarks:

i) In [13], it has been pointed out that the choice of different irreducible polynomials used to construct the field $F_{2^n}$ may affect the polynomial degree of a function defined on $F_{2^n}$. In this context, in order to reduce the number of required input-output pairs $(x_i, s(x_i))$ required in Proposition 3, one may firstly choose a suitable irreducible polynomial by which the polynomial degree of $F$ is minimized (which consequently minimizes $\deg(s)$). It has been concluded in [13] that a linear transformation on the output coordinates affects the coefficients of the exponents that belong to the same cyclotomic cosets of the exponent in the original univariate function representation. On the other hand, a linear transformation on the input coordinates (or changing the irreducible polynomial) affect only the coefficients of the exponents with Hamming weight less than or equal to the maximum Hamming weight of the exponents in original univariate function representation.

ii) In this subsection, we have considered the function $\Delta(x)$ which is a mapping from $F_{2^n}$ to $F_{2^n}$. However, one may further reduce the implementation costs (as well as the number of required number pairs $(x_i, s(x_i))$) by interpolating the function $\lambda \cdot s(x)$, for a suitable $\lambda \in F_{2^n}$ which minimizes the degree $\deg(\lambda \cdot F)$. Consequently, the function $\Delta(x)$ would be defined as $\Delta(x) = \lambda \cdot (s(x) \oplus F(x) \oplus F(x \oplus \sigma))$.

iii) Note that one can apply quantum algorithm for interpolation particularly when degree $d$ of $F$ is large. An optimal quantum algorithm [8] for interpolation requires $O(d/2)$ queries.

### 4.2 Utilizing another interpolation formula

It is well-known that any Boolean function $g : F_{2^n} \rightarrow F_2$ can be uniquely represented by its associated algebraic normal form (ANF) as follows:

$$g(x_1, \ldots, x_n) = \bigoplus_{u \in F_{2^n}} \lambda_u \left( \prod_{i=1}^{n} x_i^{u_i} \right),$$

where $x_i, \lambda_u \in F_2$ and $u = (u_1, \ldots, u_n) \in F_{2^n}$. The support of $g$ is defined as $\text{supp}(g) = \{ x \in F_{2^n} : g(x) = 1 \}$, and the algebraic degree of $g$ is defined as
\text{deg}(g) = \max\{\text{wt}(u) : \lambda_u \neq 0\}, \text{where wt}(u) \text{ denotes the number of non-zero coordinates of } u \in \mathbb{F}_2^n.

Let us now assume that the function \( g : \mathbb{F}_2^n \to \mathbb{F}_2 \) has an algebraic degree \( \text{deg}(g) \leq d < n \), and that the values of \( g \) are known on the set \( S_d = \{ y \in \mathbb{F}_2^n \mid \text{wt}(y) \leq d \} \). Then, according to [6, see page 37], this can be used to recover correctly the whole function \( g \) using the formula
\[
g(x) = \bigoplus_{y \leq x, \ y \in S_d} g(y) \left[ \left( \sum_{i=0}^{d-\text{wt}(y)} \binom{\text{wt}(x) - \text{wt}(y)}{i} \right) \mod 2 \right]. \tag{5}
\]

If we want to utilize [5] in order to recover \( k \) from \( s(x) = F_k(x) \oplus F_k(x \oplus \sigma) \oplus C, x, C \in \mathbb{F}_2^n \), we consider the following setting. Firstly, in terms of remarks given in Subsection [4.1], let us assume that a vector \( \lambda \in \mathbb{F}_2^n \) is taken such that it minimizes the the algebraic degree of \( \lambda \cdot F \) (recall that \( F \) is known). Then, one would need to have \#\( S_t \) input-output pairs \( (x_i, s(x_i)) \), where \( S_t = \{ y \in \mathbb{F}_2^n \mid \text{wt}(y) \leq t = \text{deg}(\lambda \cdot s)\} \). Clearly, knowing the pairs \( (x_i, s(x_i)) \) we can directly compute the values \( \lambda \cdot s(x_i) \), and moreover, by \( t \leq d - 1 \) we have that
\[
\#S_t = \sum_{j=0}^{t} \binom{n}{j} \leq \sum_{j=0}^{d-1} \binom{n}{j}.
\]

As \( t = \text{deg}(\lambda \cdot s) \) may not be known, then it is sufficient to require the amount of pairs equal to the sum on the right-hand side in the inequality given above. Note that one still uses the formula [5], as it is allowed that the degree of \( \lambda \cdot s \) is smaller than \( d - 1 \).

Consequently, using [6] we are able to obtain the function \( \lambda \cdot s \), and thus we are able to implement the function \( \Delta = \lambda \cdot (s(x) \oplus F(x) \oplus F(x \oplus \sigma)) \) and proceed with the extraction of \( k \) by applying Simon’s algorithm. Similarly as in the case of Proposition [4] we have the following result.

**Proposition 4.** For a vector \( \lambda \in \mathbb{F}_2^n \), let \( d = \text{deg}(\lambda \cdot F) \) be the algebraic degree (minimized by suitable \( \lambda \)) of a publicly known function \( F : \mathbb{F}_2^n \to \mathbb{F}_2^n \). Suppose that for all vectors \( x \in S = \{ y \in \mathbb{F}_2^n \mid \text{wt}(y) \leq d - 1 \} \) one knows the values of periods \( s(x) = F(x \oplus k) \oplus F(x \oplus \sigma \oplus k) \oplus C \) \( (s : \mathbb{F}_2^n \to \mathbb{F}_2^n, C \) is a constant, \( \sigma \in \mathbb{F}_2^n \setminus \{0_n\} \) which are obtained by applying Simon’s algorithm to some given function. Then:

1) One can recover the function \( \lambda \cdot s(x) \) without the knowledge of \( k \) by using formula [5].
2) By implementing the function \( \Delta(x) = \lambda \cdot [s(x) \oplus F(x) \oplus F(x \oplus \sigma)] \), one can recover \( k \) by Simon’s algorithm.

Similarly as in the case of Lagrange interpolation formula, Proposition [4] loses its efficiency if the algebraic degree of \( \lambda \cdot F \) is high. For instance, the result can be used for some smaller values of \( d = \text{deg}(\lambda \cdot F) \), which is usually the case when \( F \) represents an inner function of some GFN. For \( n = 64 \) and \( d = \text{deg}(\lambda \cdot F) = 4 \), we have that the amount of required pairs \( (x_i, s(x_i)) \) in Proposition [4] is estimated to be \( \sum_{j=0}^{d} \binom{n}{j} \approx 2^{15.4} \).
5 Conclusions

In this paper we show that the pseudorandom function Farfalle admits an application of Simon’s algorithm in various settings. Several scenarios have been shown by Constructions 1 and 2, where much more similar combinations is clearly possible. Based on the provided constructions, we show that forgery attacks are possible to mount on Farfalle-SAE and SIV modes, as well as a construction of a quantum distinguisher for the Farfalle-WBC mode. The presented attacks indicate that the main weakness of Farfalle is actually the one which may potentially admit higher order differential attacks, as discussed in [5, Section 8].

In context of the Kravatte instance (which is based on Keccak-p permutation), we note that the authors of Farfalle did not claim the quantum resistant against the attacker who can make quantum superposition queries. At the end, we show that one can extract a secret round key by applying two different interpolation formulas by using a reasonable amount of different periods obtained by applying Simon’s or Simon-Grover algorithm to reduced-round versions (Generalized) Feistel networks in many recent papers. This shows that the existing attacks on GFNs do not only provide efficient quantum distinguishers (excluding Grover’s search of certain round keys), but also one is able to derive some information related to the secret (round) key just by considering obtained periods.

References

1. D. J. Bernstein. Some challenges in heavyweight cipher design. Presented at Dagstuhl seminar on Symmetric Cryptography, Schloss Dagstuhl, 2016.
2. X. Bonnetain. Quantum key-recovery on full AEZ. Security and Cryptology, Selected Areas in Cryptography – SAC 2017, 24th International Conference, Ottawa, ON, Canada, DOI 10.1007/978-3-319-72565-9, 16–18, pp. 394–406, 2017.
3. X. Bonnetain, G. Leurent, M. N.-Plasencia, A. Schrottenloher. Quantum linearization attacks. Advances in Cryptology – ASIACRYPT 2021, LNCS vol, 13090, pp. 422–452, 2021.
4. X. Bonnetain, A. Hosoyamada, M. N.-Plasencia, Y. Sasaki, A. Schrottenloher. Quantum Attacks without superposition queries: The offline Simon algorithm. Advances in Cryptology – ASIACRYPT 2019, LNCS vol. 11921, pp. 552–583, 2019.
5. G. Bertoni, J. Daemen, S. Hoffert, M. Peeters, G. Van Assche, R. V. Keer. Farfalle: parallel permutation-based cryptography. IACR Transactions on Symmetric Cryptology, vol. 2017, no. 4, pp. 1–38, 2017.
6. C. Carlet. Boolean functions for cryptography and coding theory. New York, Cambridge University Press, https://doi.org/10.1017/9781108606806, 2020.
7. C. Cid, A. Hosoyamada, Y. Liu, S. M. Sim. Quantum cryptanalysis on contracting Feistel structures and observation on related-key settings. Progress in Cryptology – INDOCRYPT 2020: 21st International Conference on Cryptology in India, Bangalore, India, December 13–16, pp. 373–394, 2020.
8. A. M. Childs, W. van Dam, S.-H. Hung, I. E. Shparlinski. Optimal quantum algorithm for polynomial interpolation. 43rd International Colloquium on Automata, Languages, and Programming, ICALP 2016 - Rome, Italy, LIPIcs vol. 55, pp. 1–13, 2016.
9. X. Dong, B. Dong, X. Wang. Quantum attacks on some Feistel block ciphers. Designs, Codes and Cryptography, vol. 88, no. 6, pp. 1179–1203, 2020.
10. X. Dong, Z. Li, X. Wang. Quantum cryptanalysis on some generalized Feistel schemes. Science China Information Sciences, 62: 22501, 2019.
11. H. Feistel, W. A. Notz, J. L. Smith. Some cryptographic techniques for machine-to-machine data communications. Proceedings of the IEEE, vol. 63, no. 11, pp. 1545–1554, 1975.
12. S. Ghosh, P. Sarkar. Breaking tweakable enciphering schemes using Simon’s algorithm. Designs, Codes and Cryptography, vol. 89, pp. 1907–1926, 2021.
13. A. M. Yousef, G. Gong. On the interpolation attacks on block ciphers. Proceedings of the 7th International Workshop on Fast Software Encryption, pp. 109–120, 2000.
14. L. K. Grover. A fast quantum mechanical algorithm for database search. Proceedings of the twenty-eighth annual ACM symposium on Theory of Computing, Philadelphia, Pennsylvania, USA, May 22 - 24, pp. 212–219, 1996.
15. S. Hodžić, L. R. Knudsen. A quantum distinguisher for 7/8-round SMS4 block cipher. Quantum Information Processing, vol. 19, no. 11, 2020.
16. S. Hodžić, L. R. Knudsen, A. B. Kidmose. On quantum distinguishers for Type-3 Generalized Feistel Network based on separability. International Conference on Post-Quantum Cryptography, PQCrypto 2020, LNCS vol. 12100, pp. 461–480, 2020.
17. G. Ito, A. Hosoyamada, R. Matsumoto, Y. Sasaki, T. Iwata. Quantum chosen-ciphertext attacks against Feistel ciphers. Cryptographers’ Track at the RSA Conference, CT-RSA 2019: Topics in Cryptology – CT-RSA 2019, LNCS vol. 11405, pp. 391–411, 2019.
18. M. Kaplan, G. Leurent, A. Leverrier, M. N.-Plasencia. Breaking symmetric cryptosystems using quantum period finding. CRYPTO 2016: Advances in Cryptology – CRYPTO 2016, Springer, Berlin, Heidelberg, LNCS vol. 9815, pp. 207–237, 2016.
19. H. Kuwakado, M. Morii. Quantum distinguisher between the 3-round Feistel cipher and the random permutation. IEEE International Symposium on Information Theory, DOI: 10.1109/ISIT.2010.5513654, 2010.
20. H. Kuwakado, M. Morii. Security on the quantum-type Even-Mansour cipher. International Symposium on Information Theory and its Applications, October 28-31, Honolulu, HI, USA, 2012.
21. X. Lai. Higher Order Derivatives and Differential Cryptanalysis. Communications and Cryptography, SECS vol. 276, pp. 227–233, 1994.
22. G. Leander, A. May. Grover Meets Simon – Quantumly attacking the FX-construction. Advances in Cryptology – ASIACRYPT 2017, International Conference on the Theory and Application of Cryptology and Information Security, LNCS, vol. 10625, pp. 161–178, 2017.
23. M. Liskov, R. L. Rivest, D. Wagner. Tweakable block ciphers. Journal of Cryptology, vol. 24, no. 3, pp. 588–613, 2011.
24. G. L. Mullen, D. Panario. Handbook of finite fields. Published by Chapman and Hall/CRC, ISBN 9781439873786, July 18, 2013.
25. K. S. Miller. On the inverse of the sum of matrices. Mathematics Magazine, vol. 54, no. 2, pp. 67–72, 1981.
26. D. R. Simon. On the power of quantum computation. *SIAM Journal on Computing*, vol. 26, no. 5, pp. 1474–1483, 1997.

27. L. Xu, J. Guo, J. Cui, M. Li. Key-recovery attacks on LED-like block ciphers. *Tsinghua Science and Technology*, vol. 24, no. 5, pp. 585–595, 2019.

28. Y. Xu, W. Liu, W. Yu. Quantum forgery attacks on COPA, AES-COPA and marble authenticated encryption algorithms. *Quantum Information Processing*, vol. 20, no. 4, 2021.

29. Z. Zhongya, W. Wenling, S. Han, W. Bolin. Quantum attacks on Type-3 generalized Feistel scheme and unbalanced Feistel scheme with expanding functions. *Chinese Journal of Electronics*, vol. 31, no. 4, pp. 1–9, 2022.

30. B.-M. Zhou, Z. Yuan. Quantum key-recovery attack on Feistel constructions: Bernstein–Vazirani meet Grover algorithm. *Quantum Information Processing*, vol. 20, no. 330, 2021.