On Quantum Slide Attacks

Xavier Bonnetain, María Naya-Plasencia, André Schrottenloher

On Quantum Slide Attacks

Xavier Bonnetain[1,2], María Naya-Plasencia[2] and André Schrottenloher[2]

[1] Sorbonne Université, Collège Doctoral, F-75005 Paris, France
[2] Inria de Paris, France

Abstract. At Crypto 2016, Kaplan et al. proposed the first quantum exponential acceleration of a classical symmetric cryptanalysis technique: they showed that, in the superposition query model, Simon's algorithm could be applied to accelerate the slide attack on the alternate-key cipher. This allows to recover an n-bit key with $\mathcal{O}(n)$ quantum time and queries.
In this paper we propose many other types of quantum slide attacks, inspired by classical techniques including *sliding with a twist*, *complementation slide* and *mirror slidex*. These slide attacks on Feistel networks reach up to two round self-similarity with modular additions inside branch or key-addition operations. With only XOR operations, they reach up to four round self-similarity, with a cost at most quadratic in the block size. Some of these variants combined with whitening keys (FX construction) can also be successfully attacked.
Furthermore, we show that some quantum slide attacks can be composed with other quantum attacks to perform efficient key-recoveries even when the round function is a strong function classically.
Finally, we analyze the case of quantum slide attacks exploiting cycle-finding, that were thought to enjoy an exponential speed up in a paper by Bar-On et al. in 2015, where these attacks were introduced. We show that the speed-up is smaller than expected and less impressive than the above variants, but nevertheless provide improved complexities on the previous known quantum attacks in the superposition model for some self-similar SPN and Feistel constructions.

Keywords: quantum cryptanalysis, slide attacks, Feistel networks, Simon's algorithm, Kuperberg's algortihm, slidex attacks, cycle finding

1 Introduction

For a long time, symmetric primitives were believed easy to protect against quantum adversaries, by simply doubling the key length. As Grover's algorithm allows to perform an exhaustive search in the square root of the classical time, this counter measure was supposed to provide an equivalent ideal security as before. However, little was known on the attacks a quantum adversary could perform.

Indeed, many new results have recently appeared in this direction, like quantum generic meet-in-the-middle attacks on iterative block ciphers [Kap14], quantum linear and differential attacks [KLLN16b], an analysis of the FX construct against quantum adversaries [LM17], improved algorithms for collisions or multicollisions [CNS17, HSX17] and [KM10, KM12, Bon18] that respectively analyze the security of 3-round Feistel schemes, the Even-Mansour construction and quantumly break the AEZ primitive for authenticated encryption.

Related work. In [KLLN16a], Kaplan et al. considered the superposition query model and, amongst other results, provided for the first time an exponential acceleration of a classical cryptanalysis. Using Simon's algorithm [Sim94] the complexity of quantum

slide attacks on the alternate-key cipher with bit-wise additions was shown to be of $\mathcal{O}(n)$ in that model. In [AR17] some ideas for countering these attacks were proposed. The most interesting one, using modular addition, has recently been studied in detail in [BNP]. This paper provides detailed cost estimates of attacks built over Kuperberg's algorithm [Kup05], that applies to modular additions (while Simon's algorithm only concerns bit-wise additions). The authors also propose an algorithm for the case of several parallel modular additions and estimate the cost thereof.

In an independent and very recent result [DDW18], quantized versions of some advanced slide attacks on 2k- and 4k-Feistel schemes were proposed. They correspond respectively to Section 3.4 and a small part of Section 5. The authors also propose a quantum attack on GOST, which is not linked to quantum slide attacks. In [DW17], the three round distinguisher of [KM10] is exploited to reduce by three the number of keys to search for in a Feistel cipher with independent keys. In [HS17], some meet-in-the-middle attacks on Feistel constructions are proposed, in a more restricted model, and the same observation as [DW17] is presented.

Motivation and Summary of our results. In this paper we propose the quantized version of several advanced slide attacks proposed in [BW00], like slide attacks on Feistel constructions, the *complementation slide* attack and the *slide with a twist* technique. We also present situations where quantum attacks can be composed, allowing for instance to perform efficient key-recovery attacks even when the round functions are clasically strong. We provide the complexities of these attacks when the key is inserted with bitwise-addition (where Simon applies), or with modular additions.

We also propose a quantum version of mirror slide attacks from [DDKS15], and quantum slide attacks exploiting cycle finding, as was proposed in [BOBDK18]. We show that the quantum speedup of the latter is much smaller than expected by the authors.

All these attacks contribute to a better understanding of the post-quantum security of symmetric primitives.

We display in Section 1 all quantum improvements of existing slide attacks that we know of, including our new results.

Organization. The paper is organized as follows. Section 2 presents some preliminaries, as the quantum algorithms used in the paper, and the general principles of slide attacks. Section 3 proposes new quantum advanced slide attacks, considering XOR and modular additions with respect to the ones from [BW00] and [DKS15], mainly considering Feistel networks. Section 4 presents new attacks composing quantum algorithms even when the round function is classically hard to attack, with an improved version of encased algorithm when the key and branch transformations are the same. Section 5 combines the principles of the previous sections to attack 4-round self-similar Feistel ciphers. Section 6 describes our quantum slide attacks when exploiting cycle finding, and the paper is concluded in Section 7.

2 Preliminaries

In this section we provide a brief introduction to classical and quantum slide attacks, the considered quantum adversary model, and some quantum algorithms used throughout the paper. A description of the results from [KLLN16a] regarding quantum slide attacks using Simon's algorithm is also given. A summary of the classical attacks we considered can be found in the Appendix.

Table 1: Quantum slide attacks

Cipher	Attack details	Queries	Decryption oracle	Source
1k-Feistel (XOR)	Basic slide	$n/2$		[DDW18]
1k-Feistel (additions)	Basic slide	$2^{1.2\sqrt{n}}$		Section 3.2
1k-Feistel (any)	Composed slide	$n/2$ to $2^{2.4\sqrt{n}}$		Section 4
2k-Feistel (XOR)	Complementation	$n2^{n/4}$		Section 3.3
2k-Feistel (XOR)	Sliding with a twist	$n/2$	Yes	[DDW18]
2k-Feistel (XOR)	Composed slide	n		Section 4
2k-Feistel (additions)	Complementation	$2^{1.2\sqrt{n}+n/4}$		Section 3.3
2k-Feistel (additions)	Composed slide	$2^{1.8\sqrt{n}}$		Section 4
3k-Feistel (any)	Cycle finding	$2^{n/2}$		Section 6.3
4k-Feistel (XOR)	Complementation, sliding with a twist	$4n^2$	Yes	Section 5
4k-Feistel (XOR)	Enhanced reflection	$2^{n/4}$		Section 5.3
4k-Feistel (any)	Cycle finding	$2^{n/2}$		Section 6.3
4k-Whitened Feistel (DESX) (XOR)	Complementation, sliding with a twist	$n^2 2^{n/4}$ $n2^{n/4}$ (Distinguish)	Yes	Section 5
2k-Whitened Feistel (DESX)	Mirror slidex	$n2^{n/4}$		Section 5
4k-Whitened Feistel (DESX) (variant) (XOR)	Mirror slidex	n (Distinguish)	Yes	Section 5
1k-SPN (XOR)	Basic slide	n		[KLLN16a]
1k-SPN (additions)	Basic slide	$2^{1.8\sqrt{n}}$		Section 3.1
2k-SPN (any)	Grover-meet-Simon	$n2^{n/2}$		[LM17]
2k-SPN (any)	Cycle finding	$2^{n/2}$		Section 6.3

2.1 Classical Slide Attacks

Slide attacks were introduced in [BW99]. In their common principle, they are applied independently from the number of rounds, by exploiting structural properties of the cipher, especially self-similarity.

Notations. We consider a cipher $E_k : \{0,1\}^n \to \{0,1\}^n$, constructed from a round function $F(x,k) : \{0,1\}^n \times \{0,1\}^m \to \{0,1\}^n$ applied a certain number of times. There are r rounds and they use round subkeys $k_1, \dots k_r$ derived from the master key K of the cipher. In such constructions, we consider mainly two group operations: the bitwise xor addition, denoted $\oplus$, and the modular addition, denoted $+$. By abuse of notation, a general group law will also be denoted $+$.

Assumptions. We assume that F is a *weak* function, in the sense that given a pair of equations $F(x_1,k) = y_1$ and $F(x_2,k) = y_2$, it is computationally easy to retrieve the key k. Notice that in some settings, F may be more difficult to attack; in which case we will need more of these equations. Moreover, the notion of weakness is broader in a quantum setting. This is developed in Section 4.

Basic slide property. Suppose that all the round subkeys are equal: $k_1 = k, \dots k_r = k$; i.e the scheme is *one-round self-similar*. From the structure of the cipher, which is r similar applications of the same permutation $x \mapsto F(x,k)$, we may write a simple equality, the

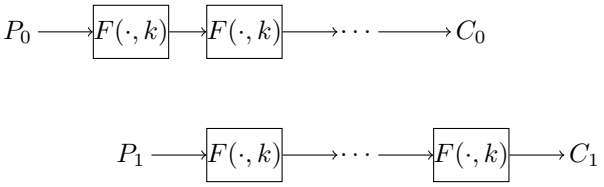

Figure 1: Illustration of the basic slide property

slide property:

$$E_K(F(x,k)) = F(E_K(x),k) \tag{1}$$

Basic slide attack. The goal of the attacker is to find two pairs x, y satisfying $F(x,k) = y$. The birthday paradox implies that, among $\mathcal{O}\left(2^{n/2}\right)$ plaintext-ciphertext couples P, C, there exists a *slide pair*: P_0, C_0 and P_1, C_1 such that $F(P_0, k) = P_1$. In that case, we also have: $F(C_0, k) = C_1$. Since we suppose F weak, these two equations suffice to retrieve the key k. Hence, the simplest attack setting consists in performing $\mathcal{O}\left(2^{n/2}\right)$ queries, then checking for each pair P_0, C_0 and P_1, C_1 if it is a slide pair; and in that case returning the key k. This requires $\mathcal{O}\left(2^{n/2}\right)$ memory and $\mathcal{O}(2^n)$ time independently of the length of k (as soon as the slide equations allow to retrieve it).

Example of weak round function. In the case of a keyed permutation $F(x,k) = k \oplus \Pi(x)$, as shown in Figure 2, a slide pair $(P_0, C_0), (P_1, C_1)$ satisfies $F(P_0, k) = P_1$ i.e $P_1 = k \oplus \Pi(P_0)$, which is equivalent to $C_1 = k \oplus \Pi(C_0)$. Hence it suffices to check if $P_1 \oplus C_1 = \Pi(P_0) \oplus \Pi(C_0)$.

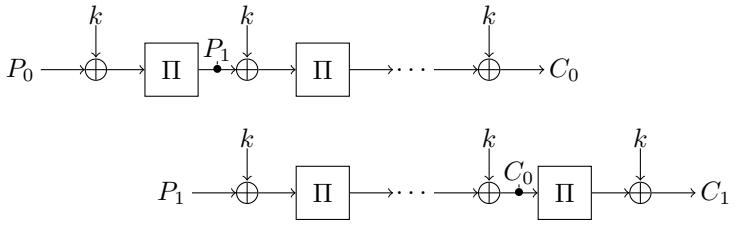

Figure 2: Example of slide attack on a cipher with a weak round function

Slide attacks have been successfully applied to the TREYFER cipher, variants of DES and Blowfish and Feistel constructions [BW99]. We do not study stream ciphers in this paper and focus on block ciphers.

2.2 Some Definitions

We define quickly some generic constructions mentioned in the rest of this paper.

rk-Feistel and rk-SPN. We name rk-Feistel and rk-SPN versions of those ciphers where the round keys form a periodic sequence of period r. In particular, 1k-Feistel denotes the case where all subkeys are equal. It may be useful to see those schemes as a composition of the same, repeated, but more intricate, round function $F = F(k_1) \circ \ldots \circ F(k_r)$.

Whitened Feistels. We also consider what has been called the DESX construction, which consists in adding additional independent whitening keys to a Feistel cipher, that is:

$$WFeistel : \begin{array}{ccccccc} \{0,1\}^k & \times & (\{0,1\}^n)^3 & & \to & & \{0,1\}^n \\ k & , & k_x, k_y, m &) & \mapsto & & k_y \oplus Feistel(k, m \oplus k_x) \end{array}.$$

This is an example of the FX construction, which consists of applying key whitenings to a block cipher.

In this paper, the notation rk-WFeistel refers to a rk-Feistel scheme augmented by key whitenings.

2.3 The Attack Model

In this paper we consider the well introduced and defined model of superposition quantum queries. It means that the adversary is not only allowed to perform local computations on a quantum computer, but is also allowed superposition queries to a remote quantum cryptographic oracle. Given a superposition of inputs to this oracle, it returns the superposition of the outputs.

This setting has been described in [DFNS13, BZ13, Zha12]. The considered attacks will be denoted as superposition attacks. This model, though strong, is simple and non-trivial, and security in this model implies security in any other scenario. It is a well accepted model and, for discussions on its importance, we refer to [GHS16] and [Gag17].

2.3.1 Cost Metric.

The cost metric we use is the number of quantum encryption/decryption queries we perform, and the number of quantum gates we have to use. Our quantum time unit is the cost of one query. We estimate that one quantum query costs $\Omega(n^2)$ quantum gates, as we are considering block ciphers, for which we want a dependency between all the input and output bits. Hence, we estimate that an algorithm with n^3 operations has a cost similar as n queries.

2.4 Quantum Slide Attacks

Quantum Slide Attacks are built upon classical ones and they reduce drastically their cost, thanks to the use of quantum algorithms and the model of superposition queries. Classically, the general layout of a slide attack is to perform a certain amount of queries, among which, thanks to the analysis, we expect one or more slide pairs to occur. Sometimes these pairs can be detected; sometimes one merely tries all possibilities. Using these pairs, one can then break a smaller component of the cipher (e.g its weak round function $F(\cdot, k)$) and retrieve some secret material (e.g some subkeys).

In the quantum setting, an exponential speedup can be obtained by the use of a quantum Hidden Shift Algorithm. In turn, this puts heavier constraints on the structure of the attack. Instead of merely promising that some slide pairs will be found, we must ensure that a Hidden Shift property holds, that must be valid for all inputs. When rewriting classical into quantum slide attacks, we will call this property a *slide-shift* property.

Generically, it consists in a functional equality, $g_0(x) = g_1(x + s)$, with two functions g_0 and g_1 depending on the encryption scheme and some publicly known parameters or components, and s a secret value (generally a subkey) that we seek. This is not always doable (in particular, it requires that all the inputs belong to a slide pair), which explains why some slide attacks have a more efficient quantum counterpart than others. Moreover, the cost depends on the group law $+$. For the bitwise XOR additions, we rely on the polynomial-time Simon's quantum algorithm [Sim94], and for modular additions, the subexponential-time Kuperberg's quantum algorithm [Kup05], which are described below.

Quantum Slide Attacks were first introduced in [KLLN16a], where the authors show that such rewriting is doable for Figure 2. Their slide-shift property is the equality, for all x:

$$\Pi(E_k(x)) \oplus k = E_k(\Pi(x \oplus k)) \ .$$

Hence $g_0(x) = g_1(x \oplus k)$ where $g_0(x) = \Pi(E_k(x))$ and $g_1(x) = E_k(\Pi(x)) \oplus x$, and since superposition queries to E_k are authorized, Simon's algorithm can be applied.

2.4.1 Composing Algorithms

The previous method recovers a value s, which depends on the instance. This value can be fixed, but it can depend on some context parameters, becoming $s(y)$ for a given y, independent of the inputs of g_0 and g_1. In that case, we can see the quantum algorithm as a classical oracle to the function $y \mapsto s(y)$. Classically, the function s is often assumed to be weak, that is, the secrets we seek can be found with a few input-output pairs.

But we can go further, and do all the computation reversibly on the quantum computer, which means we can make a *quantum* oracle to the function s, and apply any quantum algorithm on it. The relevant security notion for s becomes its *quantum security*, and not only the classical one.

As we want to make a proper unitary that computes the function $y \mapsto s(y)$, the cost is doubled, as all the intermediate computations need to be uncomputed. This is a case of quantum algorithm composition, as the Grover-meets-Simon technique [LM17]. Exemples of compositions are presented in Section 4 and 5.

2.5 Quantum Hidden Shift Algorithms

2.5.1 Simon's Algorithm

Simon's algorithm [Sim94] is a quantum algorithm designed to solve the following problem:

Problem 1 (Simon's Problem)**.** *Let $G : \{0,1\}^n \to \{0,1\}^n$. Given the promise that there exists $s \in \{0,1\}^n$ such that for any $(x, y) \in \{0,1\}^n$, $[G(x) = G(y)] \Leftrightarrow [x \oplus y \in \{0^n, s\}]$, find s.*

This problem is a *hidden subgroup problem* in the group $((\mathbb{Z}/(2))^n, \oplus)$. Due to the structure of the group, it can also solve a problem of *hidden shift* in it, which is as follows:

Problem 2 (Simon's Problem, Hidden Shift Version))**.** *Let $g_0, g_1 : \{0,1\}^n \to \{0,1\}^n$ two permutations such that there exists $s \in \{0,1\}^n$ such that, for all x, $g_0(x) = g_1(x \oplus s)$. Find s.*

Proof.

$$\text{Let } G : \begin{array}{ccc} \{0,1\} \times \{0,1\}^n & \to & \{0,1\}^n \\ (b, x) & \mapsto & \begin{cases} g_0(x) & \text{if } b = 0 \\ g_1(x) & \text{if } b = 1 \end{cases} \end{array}.$$

G satisfies $G(b_0, x_0) = G(b_1, x_1)$ if and only if $(b_0 \oplus b_1, x_0 \oplus x_1) \in \{(0,0), (1, s)\}$. □

Simon's algorithm finds a value v that satisfies $v \cdot s = 0$ in only one query. Hence, one will have enough information to retrieve s with little more than n queries.

In order to retrieve the secret, we then need to solve a linear system. The cost of this part is in $\mathcal{O}(n^3)$ gates. As one query is estimated to cost $\Omega(n^2)$ gates, the time cost will also be n. In the case of a purely quantum implementation, we estimate the cost to be $2n$, to take into account the uncomputing of the intermediate values.

2.5.2 Use in Cryptanalysis.

In cryptanalysis, the function G under study is built from a block cipher and possibly public components; we suppose that all can be queried in quantum superposition. The implication: $[G(x) = G(y)] \Rightarrow [x \oplus y \in \{0^n, s\}]$ might not be true; indeed, some additional, random collisions can occur.

As shown in [KLLN16a, Bon18, LM17], Simon's algorithm is still very efficient in that case. There is a fixed multiplicative overhead estimated to be close to 3 [KLLN16a], around 2 [LM17] or less than 1.2 [Bon18], the algorithm being less efficient if some differentials occurs with a high probability.

We will consider here a cost of n queries for Simon's algorithm, as the overhead does not change much on the complexity. The failure rate of this algorithm decreases exponentially with the number of queries once the threshold is passed, hence we will also neglect the failure probability of the algorithm when we use it as a routine for another quantum algorithm.

2.5.3 Kuperberg's Algorithm and Variants

Kuperberg's Algorithm aims at solving the *abelian* hidden shift problem, defined as:

Problem 3 (Abelian Hidden shift problem)**.** *Let $(\mathbb{G}, +)$ an abelian group, $g_0, g_1 : \mathbb{G} \to \mathbb{G}$ two permutations such that there exists $s \in \mathbb{G}$ such that, for all x, $g_0(x) = g_1(x + s)$. Find s.*

Many variants of Kuperberg's algorithm have been proposed [Kup05, Kup13, Reg04]. As we consider symmetric primitives, we're mainly interested in the case of addition modulo a power of two, which is fairly common. Moreover, we want concrete estimates for given parameters. This case has been studied in [BNP], where various algorithms and explicit costs for groups of the form $(\mathbb{Z}/(2^w))^p$ are proposed.

Concretely, the cost estimate (in quantum memory, time and queries) is $2^{\sqrt{2\log_2(3)n}} \simeq 2^{1.8\sqrt{n}}$ for the group $\mathbb{Z}/(2^n)$. The authors also propose an algorithm with a cost of $2(p/2+1)^w$ for the group $(\mathbb{Z}/(2^w))^p$, which is especially interesting for small w, as it is polynomial in p, and generalize these algorithms to any group of the form $(\mathbb{Z}/(2^w))^p$. As the worst case is for $\mathbb{Z}/(2^n)$, we will consider this complexity in our cost estimates. As before, for a purely quantum implementation, we estimate that the cost is doubled.

3 Quantum Slide-shift Attacks

In this section, we first present a quantum slide attack on an iterated keyed permutation, similar to the one in [KLLN16a], but with modular additions instead of xor, therefore using the precise results from [BNP] for applying Kuperberg's algorithm. Next, we present new advanced quantum slide attacks on Feistel networks, with slide attacks based on one-round self-similarity and advanced sliding techniques from [BW00], applied to Feistel variants. We consider all the possible combinations of modular additions and XORs. We also use some ideas from [DKS15] (mirror slidex attack). The round function is assumed weak.

3.1 Key-alternating Cipher with Modular Additions

The slide attack on the key-alternating cipher is described in Figure 3 (the primitive in itself is the first line of the figure). $\boxplus$ denotes a modular addition. The round permutation is denoted Π and the whole encryption function E_k.

As with the Simon attack, we can define the following function:

$$G : \{0,1\} \times \{0,1\}^n \to \{0,1\}^n$$
$$b, x \quad \mapsto \begin{cases} g_0(x) = \Pi(E_k(x)) - x & \text{if } b = 0, \\ g_1(x) = E_k(\Pi(x)) - x & \text{if } b = 1. \end{cases}$$

Also as in the previous attack, we know that all x satisfy $\Pi(E_k(x)) + k = E_k(\Pi(x + k))$ because of the sliding property. Then we can see that G verifies the conditions of the

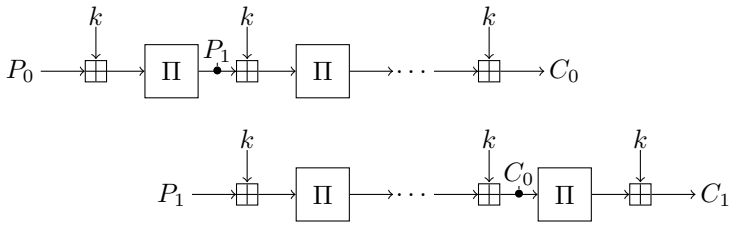

Figure 3: Slide attack on the key-alternating cipher with modular additions

hidden shift problem as $g_0(x) = g_1(x + k)$:

$$G(0, x) = \Pi(E_k(x)) - x = E_k(\Pi(x + k)) - k - x = G(1, x + k).$$

We reasonably assume that both $E_k \circ \Pi$ and $\Pi \circ E_k$ are indistinguishable from a random permutation, and we can apply Kuperberg's algorithm. This way we can recover k with a complexity of $2^{1.8\sqrt{n}}$ quantum time and queries.

3.2 Feistel Scheme with One Round Self-similarity and Modular Additions

We consider from now on Feistel schemes, like the one represented in Figure 4. The block message is decomposed in two branches, one branch has a round key added, and a round function applied to it, and is next added (by a xor or a modular addition) to the other branch before the two branches swap places. This is repeated a number of times before generating the ciphertext.

For a Feistel construction, if we consider a slide attack over one round, the right part of the first plaintext R (left side in Figure 4) will be the same as the left part of the second plaintext, L'. Classical adversaries could use a fixed right part, and take random plaintexts for the left part.

In our quantum attack, we will consider a fixed and known value for these variables $R_0 = R = L'$. As the value of $R = R_0$ is fixed, we can consider in our attack the variable $k' = f(R_0 + k)$, represented in Figure 4, as an equivalent key, and this will be the value retrieved by Kuperberg's algorithm. We denote one encryption function as E_k, we denote by $Trunc_L$ and $Trunc_R$ the function that truncates a Feistel state to only its left or right part respectively, and the round function as f, as represented in Figure 4. We first describe the function that we will use in our attack:

$$\begin{aligned} G : \{0,1\} \times \{0,1\}^{n/2} &\to \{0,1\}^{n/2} \\ b, x \quad &\mapsto \begin{cases} g_0(x) = Trunc_R(E_k(x, R_0)) & \text{if } b = 0, \\ g_1(x) = Trunc_L(E_k(R_0, x)) & \text{if } b = 1. \end{cases} \end{aligned}$$

From Figure 4 we can verify the slide shift equation $g_0(x) = g_1(x + k')$. By applying Kuperberg's algorithm we will recover the value of $k' = f(R_0 + k)$ for the fixed value of R_0 that we fixed in the beginning. Since we know R_0, we can retrieve the actual value of k when f is weak. The cost of this attack, that considers modular additions, is $2^{1.2\sqrt{n}}$. If instead we had xors, the analysis is quite similar, and the time complexity is reduced to around n. This basic attack is generalized to any non-denegerate function f in Section 4 (it works as long as the key k is added or xored as on Figure 4).

3.3 The Quantum Complementation Slide Attack

We illustrate the *complementation slide attack* [BW00] on a Feistel cipher with 2-round self-similarity, like for instance 2k-DES, introduced in [BW99]. We consider the 3 possible

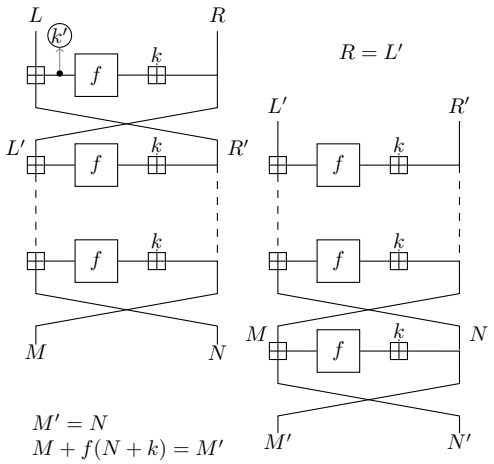

Figure 4: Slide attack on Feistel scheme with one round self-similarity and modular addition

ways of adding the keys when determining the attack complexities, but we only describe the attack considering key insertions and round combinations by modular additions. In the next section, we will consider the case where both operations are different.

The main idea of this attack, described in Figure 5, is to consider not two rounds for the slide pair, but just one. This implies that the round-keys of the middle rounds are not the same, which can be compensated by adding a relative difference between the complemented slide pairs. We will describe here the attack for the case when the key and the Feistel combination are done with modular additions. As can be seen in the figure, if we denote by $\Delta = k_0 - k_1$ and if the following equations are verified: $R = L' - \Delta$ and $L + f(R + k_0) = R' + \Delta$, then, the outputs of both plaintexts, (M, N) and (M', N') verify $N = M' - \Delta$ and $N' = M + f(k_1 + N + \Delta) - \Delta$. It is easy to verify that, with such inputs, all the transformations through the f functions will be the same pairwise through all the slid pairs. The combination of both halves ensures that the input round differences stays as wanted, and the same property can be found at the end of the functions.

In this case, in order to quantize the attack, we propose to combine an exhaustive search with Grover and Kuperberg's algorithm for solving the hidden shift problem. Here, for the sake of simplicity, we limit ourselves to describing the generic lines of the attack.

We perform an exhaustive search on Δ. The size of a block and of the whole key is n. Considering the value of R fixed to a chosen and known one, we can redefine an equivalent round-key $k'_0 = f(R + k_0)$: the correspondence between k_0 and k'_0 is bijective, and when we recover k'_0 we can immediately deduce k_0. The exhaustive search can be combined with Kuperberg's algorithm applied to the following function (keeping in mind that R and Δ are known to the attacker):

$$G : \{0,1\} \times \{0,1\}^{n/2} \to \{0,1\}^{n/2}$$
$$b, x \quad \mapsto \begin{cases} g_0(x) = Trunc_R(E_k(x, R)) + \Delta & \text{if } b = 0, \\ g_1(x) = Trunc_L(E_k(R - \Delta, x + \Delta)) & \text{if } b = 1. \end{cases}$$

From Figure 5 we can verify that $g_0(x) = g_1(x + k'_0)$. For each of the tested values for Δ, by applying Kuperberg's algorithm we will recover the value of k'_0 for the fixed value of R that we fixed in the beginning, R_0. From k'_0 and R_0 we directly recover k_0 because f is weak, and with Δ, this implies the value of k_1. When the tested value for Δ is the correct one, we should also obtain a collision given by $N' = M + f(k_1 + N + \Delta) - \Delta$, which happens with a random probability 2^{-n}. When this is the case, this implies that we

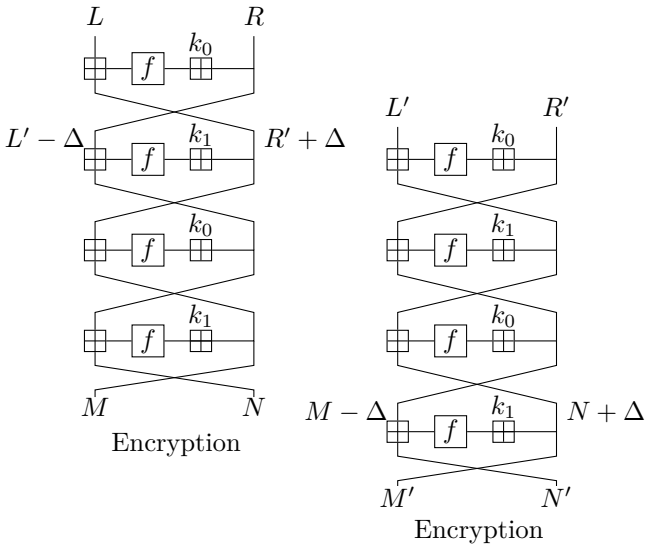

Figure 5: Representation of the complementation slide attack

have recovered the correct values of k_0 and k_1.

The cost for this if all the transformations were xors would be of $n2^{n/4}$, compared to $2^{n/2}$ from the quantum accelerated exhaustive search of the generic attack. If we have modular additions instead, the cost becomes $2^{1.2\sqrt{n}+n/4}$, which is still better than generic exhaustive search of the key. We can however do better if the key and branch addition are exactly the same. This is developed in Section 4.

3.4 Sliding with a Twist

A further improved variant of the slide attacks is the *sliding with a twist*, also introduced in [BW00], that can be applied against some Feistel constructions when we have superposition access to the decryption oracle.

This attack applies to ciphers with the same structure as in Figure 6, and small variants can also be applied to DESX or to reduced-round versions of GOST. The quantum version can be applied as long as the two branches are added with a xor. The round key can be added also by modular addition or multiple modular additions. We will describe the attack considering key insertions and round combinations by xor, the addition of the key being irrelevant for the complexity.

The main idea of this attack is to consider that encryption of a two-round self similarity Feistel cipher is a slid version of its decryption, modulo the final twists, that are easily taken into account. For (L, R), (M, N) as inputs and outputs of the encryption function and $(M'N')$, (L', R') as inputs and outputs of the decryption one, we have that if $R = N'$ and $M' = L \oplus f(R \oplus k_0)$, then $R' = N$ and $L' = M \oplus f(N \oplus k_0)$.

We can consider now $R = N'$ as a fixed chosen value. Like the previous attack, if we consider an equivalent key $k'_0 = f(R \oplus k_0)$, we can apply Simon's algorithm. Let us denote the decryption function D_k.

$$G : \{0,1\} \times \{0,1\}^{n/2} \to \{0,1\}^{n/2}$$
$$b, x \quad \mapsto \begin{cases} g_0(x) = Trunc_R(E_k(x, R)) = N & \text{if } b = 0, \\ g_1(x) = Trunc_R(D_k(x, R)) = R' & \text{if } b = 1. \end{cases}$$

From Figure 6 we can verify the slide shift equation $g_0(x) = g_1(x \oplus k'_0)$. Simon's algorithm recovers the value of k'_0, and from it, also the one of k_0 with negligible complexity

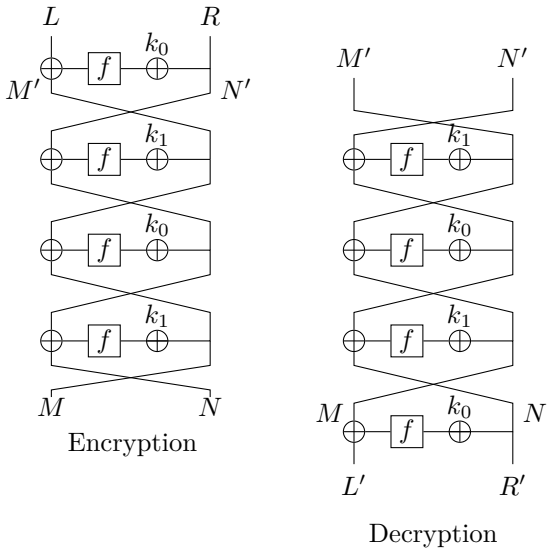

Figure 6: Representation of the sliding with a twist technique

because f is easy to invert. We can repeat a similar attack peeling off one layer in order to recover also k_1 with comparable complexity.

The cost when all the transformations are xors is n, compared to $2^{n/2}$ for the quantum accelerated exhaustive search. If we have modular additions between branches, this attack does not apply, as the decryption scheme has subtractions instead of additions.

4 Composing Quantum Attacks: Key-Recovery with Strong Round Functions

In this section, we propose efficient key-recovery attacks on self-similar Feistel constructions when using some classically strong round functions. We show that as long as the round function is vulnerable to quantum attacks, the key will be recovered with a complexity at most the cost of the quantum round attack multiplied by the cost of the quantum attack on the one-round self-similar Feistel (that will be around n if the branch addition is bitwise, or around $2^{1.2\sqrt{n}}$ if it is a modular addition), by composing both algorithms. This composition allows to recover the key and perform an attack in cases that are classically sound.

In what follows, we describe the generic case and show an even more efficient attack when the round function has the form $f(x + k)$ (with f strong as, for instance, a hash function) and the branch and key additions are the same.

4.1 General Attack

In the attacks presented in the previous section, the round function was supposed to be weak. If it was not, we would still be able to recover the values of $k' = f_k(R_0)$. The problem next would be to recover k, which would not be directly possible.

We show here how to quantumly attack the round function in some common cases. In the previous attacks, the value of R_0 was fixed to a random chosen value. This means that we can repeat the same operation for several values R_i.

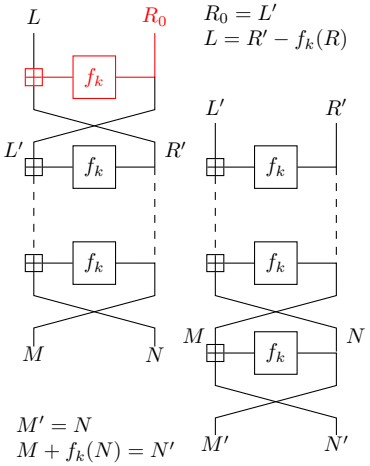

Figure 7: Slide attack on a generic one-round self-similar Feistel

Case f(x+k). We consider here that the round function is a key addition composed with a public function f, as this is a fairly common design. We use the attack of Section 3.2, which, given a value R_0, produces $f(R_0 + k)$. As presented in Section 2.4, we consider this attack as a quantum black box H, that takes an input x, and produces $f(x + k)$. Given this H, we can construct the following function G:

$$G : \{0,1\} \times \{0,1\}^{n/2} \to \{0,1\}^{n/2}$$
$$b, x \quad \mapsto \begin{cases} g_0(x) = H(x) = f(x+k) & \text{if } b = 0, \\ g_1(x) = f(x) & \text{if } b = 1. \end{cases}$$

It is now easy to identify k as the hidden shift of G.

The cost of the attack is summarized in Table 2 for a Feistel scheme with one-round self-similarity, and it is easy to extend it to other scenarios. For the case of two modular additions we can still have different operators if they use, for instance, a different bit order.

Generalization of the attack. We described the attack for a Feistel round function of the form $f(x + k)$, but we can use this attack for any keyed function vulnerable to quantum key recovery, like Even-Mansour (see Figure 8), a 3-round Feistel, a self-similar Feistel...

Using the same principle as above, we manage to craft a quantum oracle to the inner round function. Now, we can attack it using any quantum algorithm that needs a quantum superposition access on it, without any additional constraint. The total cost of the attack is the cost to attack the round function multiplied by the cost of the slide attack. This means that the one-round self-similar Feistel construct offers essentially the security of its inner round function. This is not the case classically, as the cost to find a slide pair is $2^{n/2}$; it would generally be the cost of the total attack, as the inner round function operates on $n/2$ bits.

For the example in Figure 8, the exhaustive search is classically in $2^{n/2}$, quantumly in $2^{n/4}$. The classical slide attack can find a pair $(x, f_k(x))$ in $2^{n/4}$ queries. Once this is done, we can easily retrieve some other slide pairs using the slide pairs at the output of the cipher, and do the standard slide attack to break Even-Mansour in $2^{n/4}$ more queries and time. The quantum slide attack performs in $n2^{1.2\sqrt{n}}$ queries.

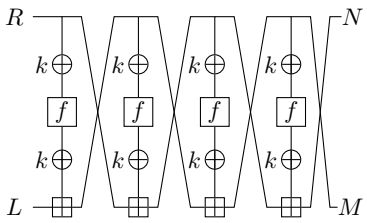

Figure 8: Example of a vulnerable Feistel

4.2 With the Same Branch and Key Addition

If the round function is $f(x+k)$ and the branch and the key addition both use the same law, the attack can be made more efficient. We can consider a Feistel construct E_k composed of a certain number of iterations of the function $f_k(x,y) = y, x + f(y+k)$. In that case, we have $f_k(x,y) + (k,k) = f_0(x+k, y+k)$, hence $E_k(x,y) + (k,k) = E_0(x+k, y+k)$. We can then consider

$$G : \{0,1\} \times \{0,1\}^{n/2} \to \{0,1\}^n$$
$$b, x \quad \mapsto \begin{cases} g_0(x) = E_k(x,x) + (x,x) & \text{if } b = 0, \\ g_1(x) = E_0(x,x) + (x,x) & \text{if } b = 1. \end{cases}$$

The hidden shift is k here.

Case $f(x+k)$, 2-rounds Self-Similarity. It can also be generalized if the Feistel has two alternating keys (k_0, k_1), we can use

$$G : \{0,1\} \times \left(\{0,1\}^{n/2}\right)^2 \to \{0,1\}^n$$
$$b, x, y \quad \mapsto \begin{cases} g_0(x) = E_{k_0,k_1}(x,y) + (x,y) & \text{if } b = 0, \\ g_1(x) = E_{0,0}(x,y) + (x,y) & \text{if } b = 1. \end{cases}$$

As we have the same property $E_{k_0,k_1} + (k_0,k_1) = E_0(x+k_0, y+k_1)$, the hidden shift is (k_0, k_1).

Table 2 presents a summary of the costs of this attack, depending on the group operation for the branch addition, and the key addition. + can represent a modular addition, or multiple additions in parallel, as long as it is the same operation at each round. It can be different for the branch and the key, except for the sixth and last attack in the table.

5 Advanced Quantum Slide Attacks on Feistels

In this section, we give our best attacks on self-similar Feistel schemes, most of them based on *slide shift* properties. These attacks reach up to four rounds of self-similarity, they run very efficiently and also concern whitened versions, but the operations used (branch or key addition, whitenings) are limited to xors.

5.1 Mirror Slidex Attack on 2k-WFeistel.

The *mirror slidex* attack [DKS15] applies to any cipher that can be decomposed as $E = E_2 \circ E_1 \circ E_0$ where E_1 is an involution ($E_1 \circ E_1 = Id$). It is in fact a generalization of the *sliding with a twist*, which corresponds to the case where $E_2 = Id$.

Table 2: Summary of the slide attack costs, n is the block size.

Branch, Key add.	Round function	quantum query cost	Remark
$\oplus$, any	Weak	$n/2$	
$\oplus, \oplus$	$f(x \oplus k)$	$n/2$	
$\oplus, \oplus$	$f_k(x)$	$n^2/2$	Simon calls purely quantum Simon
$\oplus, +$	$f_k(x)$	$n2^{1.2\sqrt{n}}$	Kuper. calls purely quantum Simon
$+$, any	Weak	$2^{1.2\sqrt{n}}$	
$+, +$	$f(x + k)$	$2^{1.2\sqrt{n}}$	Need the exact same addition
$+, \oplus$	$f_k(x)$	$n2^{1.2\sqrt{n}}$	Simon calls purely quantum Kuper.
$+, +$	$f_k(x)$	$2^{2.4\sqrt{n}+1}$	Kuper. calls purely quantum Kuper.
$\oplus, \oplus$	$f(x \oplus k)$	n	2-Round self-similarity
$+, +$	$f(x + k)$	$2^{1.8\sqrt{n}}$	2-Round self-similarity, Need the exact same addition

In that case, the equation that we can write is: $E_0^{-1} \circ E_2^{-1} \circ E(P) = E^{-1} \circ E_2(E_0(P))$. If E_0 and E_2 have a good form, we can turn it into a slide shift property, and quantumly attack using Simon's or Kuperberg's algorithms. A classical mirror slidex does not necessarily give rise to a quantum one.

Consider a 2k-WFeistel scheme as in [DKS15, Section 7.3] (2K-DESX in the paper). The authors combine the mirror slidex with the complementation slide. This is also what we do, although we prefer writing the attack as an updated slide shift. Let us denote

$$E(x) = k_{post} \oplus E_k(k_{pre} \oplus x)$$

where E is the full cipher, k_{pre} and k_{post} are the pre and post-whitening keys, of size n each (they are xored to the full state of the cipher), and E_k is a 2k-Feistel scheme of alternating keys k_0, k_1 of size $\frac{n}{2}$ each. We have $E_k(x) = k_{post} \oplus E(k_{pre} \oplus x)$. We also denote $k^L_{post}, k^R_{post}, k^L_{pre}, k^R_{pre}$ the respective left and right sides of the whitening keys. The whole key of E is $k_0, k_1, k_{pre}, k_{post}$ of size $3n$. We note $\Delta = k_0 \oplus k_1$. Recall the slide shift property coming from the *complementation slide* technique:

$$Trunc_R(E_k(x, R)) \oplus \Delta = Trunc_L(E_k(R \oplus \Delta, x \oplus f(R \oplus k_0) \oplus \Delta)) \ .$$

We replace calls to E_k by E, which is our oracle, and obtain that:

$$Trunc_R(E(x \oplus k^L_{pre}, R \oplus k^R_{pre})) \oplus k^R_{post} \oplus \Delta = \\ Trunc_L(E(R \oplus \Delta \oplus k^L_{pre}, x \oplus k^R_{pre} \oplus f(R \oplus k_0) \oplus \Delta)) \oplus k^L_{post} \ .$$

Let us denote $\Delta' = \Delta \oplus k^L_{pre} \oplus k^R_{pre}$ as an adapted value of Δ, we rewrite:

$$Trunc_R(E(x, R)) \oplus \Delta' \oplus (k^L_{pre} \oplus k^R_{pre} \oplus k^L_{post} \oplus k^R_{post}) = \\ Trunc_L(E(R \oplus \Delta', x \oplus \Delta' \oplus f(R \oplus k_0))) \ .$$

Our new slide function is:

$$G : \{0,1\} \times \left(\{0,1\}^{n/2}\right)^2 \to \{0,1\}^{n/2}$$

$$b, x, y \quad \mapsto \begin{cases} g_0(x, y) = Trunc_R(E_k(x, R)) \oplus y & \text{if } b = 0, \\ g_1(x, y) = Trunc_L(E_k(R \oplus \Delta', x \oplus \Delta')) \oplus \Delta' \oplus y & \text{if } b = 1. \end{cases}$$

where we guess the value of Δ'. Then G verifies the slide shift property $g_0(x, y) = g_1(x \oplus f(R \oplus k_0), y \oplus (k^L_{pre} \oplus k^R_{pre} \oplus k^L_{post} \oplus k^R_{post}))$.

Now, as we did in Section 4, we can consider this whole slide attack as a black box, that, given an input R, outputs $f(R \oplus k_0)$. As f is public, we can create a quantum oracle to the function $G'(R) = f(R) \oplus f(R \oplus k_0)$. This function has k_0 as a hidden period, which can be recovered, by reusing Simon's algorithm.

Since Δ' is guessed, we deduce $k_1 \oplus k^L_{pre} \oplus k^R_{pre}$. There now remains only one unknown subkey in the Feistel. We can apply the quantum attack on the FX construction [LM17] to recover it and the whitening keys in $\mathcal{O}\big(n2^{n/4}\big)$ queries and time.

If different operations are used (e.g, modular additions for the whitenings and a xor inside the mirror property), this attack does not seem to work anymore, as the whitening and the slide operation would not commute. Furthermore, using decryption queries (as in the *sliding with a twist* attack on 2k-Feistel) does not seem to allow the complexity to decrease, due to the presence of the whitenings. A guess of $\frac{n}{2}$ bits of information, with Grover, seems still mandatory.

5.2 Attacking 4k-Feistel and 4k-WFeistel

Combining twist and complementation slides enables the authors in [BW00] to attack a 4k-Feistel. In this section we show how to efficiently quantize this attack and extend it to 4k-WFeistel. The main idea is that encryption and decryption can be compared in this way (see Figure 9):

$$k_0 k_1 k_2 k_3 k_0 k_1 \dots$$
$$k_3 k_2 k_1 k_0 \dots$$

So that the keys k_0 and k_2 always coincide, whereas the keys k_3 and k_0 correspond to rounds having a constant difference.

The slide pairs will have a difference $\Delta = k_1 \oplus k_3$ as can be seen in Figure 9, similarly to the *complementation slide* technique. We gather from [BW00] that a slide pair $(P, C) = (L, R), (M, N)$ (in input to the encryption function), $(P', C') = (L', R'), (M', N')$ (in input to the decryption function) satisfies the following properties:

$$L', R' = M \oplus \Delta \oplus f(k_0 \oplus N), N$$
$$M', N' = L \oplus f(R \oplus k_0) \oplus \Delta, R$$

For all $P = (x, R)$ we write $D_k(M', N') = (L', R')$, $M = Trunc_L(E_k(x, R))$, $N = Trunc_R(E_k(x, R))$, which gives:

$$D_k(x \oplus f(R \oplus k_0) \oplus \Delta, R) = M \oplus \Delta \oplus f(k_0 \oplus N), N$$
$$\implies Trunc_R(D_k(x \oplus f(R \oplus k_0) \oplus \Delta, R)) = Trunc_R(E_k(x, R))$$

Hence, for a fixed R, we have the following function G:

$$G : \{0,1\} \times \{0,1\}^{n/2} \to \{0,1\}^{n/2}$$
$$b, x \mapsto \begin{cases} g_0(x) = Trunc_R(D_k(x, R)) & \text{if } b = 0, \\ g_1(x) = Trunc_R(E_k(x, R)) & \text{if } b = 1. \end{cases}$$

The hidden shift between g_0 and g_1 is $f(R \oplus k_0) \oplus \Delta$. As in Section 4, we remark that the function $R \mapsto f(R \oplus k_0) \oplus \Delta \oplus f(R)$ has k_0 as hidden period, which can be recovered with Simon's algorithm.

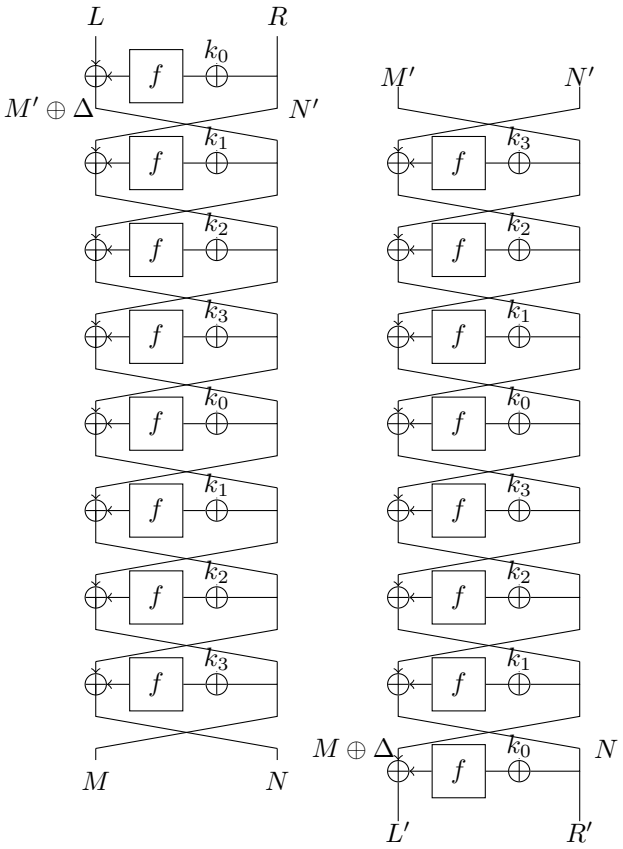

Figure 9: Complementation and twist combined on a 4k-Feistel scheme

The total cost is around $n^2/2$ queries (the input space is of size $n/2$, the inner Simon needs uncomputation), for recovering k_0. Once k_0 is known, we can directly use the first attack to get $\Delta = k_1 \oplus k_3$, at a negligible cost of $n/2$.

Moreover, we can "shift" the cipher, that is, suppress the first round of encryption, and add it at the last round, in order to get access to the same cipher, but with keys (k_1, k_2, k_3, k_0) instead of (k_0, k_1, k_2, k_3). Hence, we can repeat the same attack, to get k_1, and $\Delta' = k_2 \oplus k_0$. With k_1 and Δ, we obtain k_3, and with k_0 and Δ', we obtain k_2. The total cost is around n^2, for a total key size of $2n$ bits.

As this attack uses the *sliding with a twist* technique, it does not transpose well to modular additions (the decryption oracle would use subtractions instead of additions).

Attacking 4k-WFeistel. Building on the 4k-Feistel attack of above, we can further extend it to 4k-WFeistel, with key whitenings k_{pre} and k_{post}. Indeed, if we rewrite the slide shift equation:

$$Trunc_R(D_k(x \oplus f(R \oplus k_0) \oplus \Delta, R)) = Trunc_R(E_k(x, R))$$

using the whole primitive $E(\cdot) = k_{post} \oplus E_k(k_{pre} \oplus \cdot)$, we obtain:

$$Trunc_R(D(x \oplus k^L_{post} \oplus f(R \oplus k_0) \oplus \Delta, R \oplus k^R_{post})) \oplus k^R_{pre} = Trunc_R(E(x \oplus k^L_{pre}, R \oplus k^R_{pre})) \oplus k^R_{post}$$

which we rewrite:

$$Trunc_R(D(x \oplus \delta_L \oplus f(R \oplus k_0 \oplus k^R_{post}), R)) = Trunc_R(E(x, R \oplus \delta_R)) \oplus \delta_R$$

where $\delta_R = k^R_{pre} \oplus k^R_{post}$ and $\delta_L = k^L_{pre} \oplus k^L_{post} \oplus k_1 \oplus k_3$.

Now, we can use the function

$$G : \{0,1\} \times \left(\{0,1\}^{n/2}\right)^2 \to \{0,1\}^{n/2}$$
$$b, x, y \quad \mapsto \begin{cases} g_0(x,y) = Trunc_R(D(x \oplus f(y \oplus (k^R_{post} \oplus k_0)), y) \oplus y & \text{if } b = 0, \\ g_1(x,y) = Trunc_R(E(x,y)) & \text{if } b = 1. \end{cases}$$

We have $g_0(x,y) = g_1(x \oplus \delta_L, y \oplus \delta_R)$. However, to implement g_0, we need to know the value $(k^R_{pos} \oplus k_0)$. Hence, we first guess the value of $(k^R_{post} \oplus k_0)$ using Grover's algorithm, and combine it with Simon's algorithm as studied in [LM17]. In time and queries $2n2^{n/4}$, this attack recovers δ_L, δ_R and $(k^R_{pos} \oplus k_0)$. Now, we can perform the analogue of the classical attack: swapping the encryption and the decryption, we obtain $k^L_{pre} \oplus k^L_{post}$ and $k^R_{pre} \oplus k^R_{post} \oplus k_0 \oplus k_2$, which allows to obtain $k_0 \oplus k_2$, $k_1 \oplus k_3$ and $k_{pre} \oplus k_{post}$. This step costs $\mathcal{O}\left(n2^{n/4}\right)$ time.

But we can also move differently δ_R in the slide shift property:

$$Trunc_R(D(x \oplus \delta_L \oplus f(R \oplus k_0 \oplus k^R_{pre}), R \oplus \delta_R)) = Trunc_R(E(x,R)) \oplus \delta_R \ .$$

We now interpret this as a hidden shift equation on x only and, given an arbitrary R, we can recover $f(R \oplus k_0 \oplus k^R_{pre})$. In turn, this gives us the value $k_0 \oplus k^R_{pre}$ for a total of n^2 queries.

Since we have already guessed $(k^R_{pos} \oplus k_0)$, we deduce $k_0, k^R_{pre}, k^R_{post}$, and k_2 from $k_0 \oplus k_2$. Only $\frac{n}{2}$ bits of key material remain unknown in the Feistel, as $k_1 \oplus k_3$ has been previously obtained. We can break the complete cipher and finish the key-recovery by applying the Grover-meet-Simon attack on the FX construction of [LM17], in time $\mathcal{O}\left(n2^{n/4}\right)$.

As this attack uses the *sliding with a twist* technique, it does not transpose well to modular additions.

A variant of 4k-WFeistel. We follow [DKS15, Section 7.4] and consider the 4k-WFeistel scheme (4K-DESX in the paper), with two whitening keys k_{pre} and k_{post} and four alternating keys k_0, k_1, k_2, k_3 scheduled in $4m+1$ rounds as: $(k_0, k_1, k_2, k_3)^m, k_0$.

Slide pairs $(P, C), (P', C')$ have the properties:

$$P \oplus C' = k_{pre} \oplus k_{post} \oplus (k_1 \oplus k_3 || 0)$$
$$E_k(P \oplus k_{pre}) = E_k^{-1}(C' \oplus k_{post}) \oplus (k_1 \oplus k_3 || 0)$$

Where the term $\Delta = k_1 \oplus k_3$ intervenes as in the *complementation slide* to correct the inversion of E_k. We can rewrite this as a slide shift equation holding for all input x:

$$E_k(x) = E_k^{-1}(x \oplus (\Delta||0)) \oplus (\Delta||0)$$
$$\implies E(x \oplus k_{pre}) \oplus k_{post} = E^{-1}(x \oplus k_{post} \oplus (\Delta||0)) \oplus k_{pre} \oplus (\Delta||0)$$
$$\implies E(x) \oplus x = E^{-1}(x \oplus \Delta') \oplus x \oplus \Delta'$$

where $\Delta' = k_{pre} \oplus k_{post} \oplus (\Delta||0)$, a slide shift equation holding for all x. We can retrieve Δ' in only n queries, achieving a very efficient distinguisher. Obtaining the rest of the key seems more difficult.

5.3 Enhanced Reflection Attack

The *enhanced reflection attack* was introduced in [DDKS15], as an improvement of the mirror slidex attack, for ciphers of the form $E = E_2 \circ E_1 \circ E_0$ where E_1 is an involution. It requires to find P such that $E_0(P)$ is a *fixpoint* of E_1. This happens with probability $2^{-n/2}$. In this case we get directly $C = E_2(E_0(P))$.

In the case of 4-key-alternating Feistels, in this section, multiple pairs (P, C) are needed but the fixpoints can be detected. Notice that these attacks do not use a slide shift property as above.

Enhanced reflection attack on 4k-Feistel. In [DDKS15], the authors use a reflection attack on 4k-Feistel (which they name 4k-DES), where the four alternating subkeys are denoted k_0, k_1, k_2, k_3. They prove (4.1.3, Property 3) that if (P, C) is a plaintext-ciphertext pair for $4m$-round 4k-Feistel, such that in the encryption process of P we have $X^L_{2m-1} = X^L_{2m+1} \oplus \Delta$ where $\Delta = k_1 \oplus k_3$, then $P^L = C^L$ and $P^R = C^R \oplus Out_{4m} \oplus \Delta$.

Out_{4m} is the output of the internal mixing function f at the $4m$-th step, so $Out_{4m} = f(X^R_{4m} \oplus k_3) = f(C^R \oplus k_3)$.

Reflection points, that satisfy these properties, can be detected by $P^L = C^L$. Only $\mathcal{O}\big(2^{n/2}\big)$ known plaintexts are required. Given at least three of them, the adversary guesses Δ, then tries to obtain k_3 from the equation: if Δ is good, this works and both k_1 and k_3 can be obtained.

To complete the attack, the authors note that a similar reflection property holds with the equation $P^R = C^R$ to detect reflection points, and $P^L = C^L \oplus Out_1 \oplus (k_0 \oplus k_2)$.

In a quantum setting, only $\mathcal{O}\big(2^{n/4}\big)$ superposition queries are enough, to retrieve the reflection points, using Grover search over all plaintexts P. Again, trying all values of Δ requires $\mathcal{O}\big(2^{n/4}\big)$ work. All subkeys are recovered in time $\mathcal{O}\big(2^{n/4}\big)$.

Enhanced reflection attack on 4k-WFeistel. The attack above on 4k-Feistel, using reflection points, can be turned into an attack for 4k-WFeistel. Suppose that $E(x) = k_{post} \oplus E'(k_{pre} \oplus x)$ where E' is a 4k-Feistel procedure. The adversary first has to guess $k^L_{pre} \oplus k^L_{post}$. To do this, remark that reflection points for E' of the form P', C' turn into reflection points for E that satisfy $P^L \oplus C^L = k^L_{pre} \oplus k^L_{post}$.

In this, the correct value of $k^L_{pre} \oplus k^L_{post}$ appears with probability $2 \cdot 2^{-n/2}$, whereas all incorrect values have probability $2^{-n/2}$ of appearance. This allows to retrieve $k^L_{pre} \oplus k^L_{post}$ using $\mathcal{O}\big(n2^{n/2}\big)$ memory and time, the bottleneck of the attack. Indeed, reformulated in an abstract way, the problem that we are solving is:

Problem 4 (Most frequent value)**.** *Given a function $f : [N^2] \to [N]$ such that any value in $[N]$ appears $\frac{N^2}{N+1}$ times in the images of f, except a single value y, which appears $\frac{2N^2}{N+1}$ times (twice as much); find y.*

Assume that we make $c(2^{n/2} + 1)$ queries for some c. Due to Chernoff and union bounds, given $\delta, \gamma < 1$, the probability that the number of y among the queries is below than $(1 - \delta)2c$ is less than $\exp(-\delta^2 c)$. The joint probability that the count of some output, different from y, exceeds $(1 + \gamma)c$, is less than $2^{n/2} \exp(-\gamma^2 c/3)$. We tailor c, δ and γ so that these probabilities become negligible in n, avoiding the case of a failure. If we take $c = 2n$, $\delta = \frac{1}{8}$ and $\gamma = \frac{3}{4}$, we find that it suffices to look for the element which appears more than $\frac{7}{2}n$ times.

As soon as the value of $k^L_{pre} \oplus k^L_{post}$ is obtained, the reflection points can be detected

via the equation $P^L \oplus C^L = k^L_{pre} \oplus k^L_{post}$, and the second equation:

$$\begin{aligned} P^R &= C^R \oplus Out_{4m} \oplus (k_1 \oplus k_3) \oplus k^R_{pre} \oplus k^R_{post} \\ &= C^R \oplus f(C^R \oplus k_3) \oplus (k_1 \oplus k_3) \oplus k^R_{pre} \oplus k^R_{post} \end{aligned}$$

For example, the adversary guesses $(k_1 \oplus k_3) \oplus k^R_{pre} \oplus k^R_{post}$ and checks that two reflection points indeed yield the same value for k_3.

A quantization of this attack would require to solve efficiently the problem of finding the value of $k^L_{pre} \oplus k^L_{post}$. This seems actually difficult, and we don't know of any quantum algorithm for Problem 4 that would improve by more than a constant factor the number of queries required. The main difficulty, due to which the well-known algorithms for element distinctness [Amb16], collision [BHT16] or multicollision [HSX17] cannot be applied, seems to be that the most frequent value is not known beforehand. We leave this as an open question.

6 On Quantum Attacks Exploiting Cycle Finding

A generic framework of cycle-based slide attacks is presented in [BOBDK18, Section 4.1]. The authors suggest that it could be accelerated in a similar way as the slide attacks from [KLLN16a], expecting for instance exponential speedups. In this section we find much smaller improvements than expected, as the attack do not seem to have a slide-shift structure. We are nevertheless able to propose some improved-over-classical quantum associated attacks.

6.1 Definition of a Cycle Slide Attack

We suppose that $E_k = f_k^\ell$ for some function f_k, which happens to be immune to simpler slide attacks such as those presented above for 1k-, 2k- and 4k-Feistel schemes. Consider a message P and the cycle built from P using E_k:

$$P, E_k(P), E_k^2(P), \dots$$

Let m_2 be the period of this cycle. Let also m_1 be the period of the f_k-cycle, that is, the smallest integer such that $f_k^{m_1}(P) = P$. Then one has $m_2 = m_1/gcd(m_1, \ell)$. Moreover, suppose that $gcd(m_1, \ell) = 1$, then $m_1 = m_2 = m$. This condition cannot be checked directly by the attacker, since he does not have access to f_k.

By Bezout's theorem, there exists d_1, d_2 such that $d_1 m - d_2 \ell = 1$. This gives:

$$\begin{aligned} f_k^{d_1 m - d_2 \ell + 1}(P) &= P \\ f_k^{d_1 m + 1} &= f_k^{d_2 \ell}(P) = E_k^{d_2}(P) \\ f_k(P) &= E_k^{d_2}(P) \end{aligned}$$

Hence $(P, E_k^{d_2}(P))$ is a slide pair. Moreover, $(E_k^t(P), E_k^{d_2+t}(P))$ is one for every t. This gives a certain number of slide pairs "for free", depending on the length of the cycle. Once they have been obtained, we can use them to perform an attack on f_k and try to recover the key material.

General cycle size. We assume that E_k is a random permutation. In that case, the i-th largest cycle has size $e^{-i+1}\,(1 - 1/e)\,2^n$ (on average), the largest having size $(1 - 1/e)\,2^n$. In particular, on average, half of the points lie on the largest cycle. Finding a cycle of E_k then requires $c2^n$ chosen plaintext queries for some $c < 1$, which is a little less than the entire codebook.

The main interest is the time complexity: suppose that the attack on f_k needs time $\mathcal{O}(t)$, then the total time complexity is $\mathcal{O}(t + 2^n)$ and not $\mathcal{O}(t2^n)$ as would require a standard slide attack (see [BOBDK18, Section 4.1]).

Combining cycles. It is important to note that the probability of success (i.e, m_1 is prime with ℓ) is strictly smaller than one, exactly $\phi(\ell)/\ell$ where ϕ is Euler's totient function. The only way to check that the slide pairs obtained were good is to try to attack f_k. Hence, it may be difficult to combine (when needed) the data obtained from multiple cycles.

In particular, if one is not able to tell if a given cycle is a good one (i.e, m_1 is prime with ℓ), the complexity can increase dramatically, since we would require all cycles to be good at the same time: it happens only with probability $(\phi(\ell)/\ell)^t$ if there are t of them.

Classical Examples and Applications. The cycle slide attack is applied in [BOBDK18] against a palindromic Feistel scheme, where the keys are scheduled as:

$$k_0, k_1 \dots k_r, k_r, k_{r-1}, \dots k_0, k_0, k_1, \dots$$

The function f_k is a Feistel scheme with $2r$ keys scheduled as $k_0, k_1 \dots k_r, k_r, k_{r-1}, \dots k_0$. In that case, the average cycle length of a plaintext P is of order $\mathcal{O}\left(2^{n/2}\right)$, limiting considerably the cost of the attack and making it possible. But since the cycle is of length $\mathcal{O}\left(2^{n/2}\right)$, it gives only $\mathcal{O}\left(2^{n/2}\right)$ slide pairs: the distinguishing attack on f_k required should need only $\mathcal{O}\left(2^{n/2}\right)$ plaintext-ciphertext pairs.

In [BOBDK18, section 5], the authors present attacks on GOST. First:

- A distinguishing attack on 8-round GOST with unknown S-boxes, using 2^{36} known plaintexts and time (it relies on differentials);
- A key-recovery attack on 8-round GOST with unknown S-boxes, based on a differential attack, using $2^{36.5}$ known plaintexts, data and time.

This allows to attack the 24 first rounds of GOST, that correspond to f_k^3 where f_k is an 8-round GOST, efficiently using the cycle-based slide attack. This requires 2^{63} chosen plaintext and time.

The authors also attack the palindromic full GOST: having a palindromic key, it is in fact f_k^4 where f_k is a palindromic Feistel scheme. Hence the cycle-based attack works more efficiently (the cycles have small size). The distinguishing attack allows to retrieve the good slide pairs, and once there are enough, the key-recovery attack is launched on f_k. The total needs 2^{40} chosen plaintexts, memory and time.

6.2 Quantization of a Cycle-based Slide Attack

At the end of [BOBDK18, Section 4.1], the authors suggest that a quantum period-finding algorithm could be applied to cycle-based slide attacks. To the best of our knowledge, this seems not possible. Indeed, given a point P, the period that is of interest here is the one of the function:

$$G \ : d \in \mathbb{Z} \mapsto E_k^d(P)$$

If the function $G(d) = E_k^d(P)$ was implemented using a quantum circuit, we could indeed use Shor's period-finding algorithm to retrieve the cycle length. But this would require to call E_k^d in superposition over d, which seems highly difficult. The encryption oracle E_k is actually only accessed through queries $x \to E_k(x)$, in superposition over x. To compute E_k^d for some d, one still needs to perform d *successive calls* to E_k. A superposition oracle for G is indeed constructible, but it would have the same time complexity as the classical one (the highest value of d in the input). On the contrary, Shor's algorithm uses an oracle for $f(r) = a^r \mod n$, and fast exponentiation allows to compute it efficiently.

Quantization. It is however possible, using Grover search, to improve on the classical complexity of some cycle slide attacks. Finding a point that lies on a cycle of length smaller than d, if there are k such points, can be done in $\mathcal{O}\left(\sqrt{2^n/k}\right)$ calls to a testing subprocedure G, implemented using a quantum algorithm, that computes: $G(P) = 1 \iff \exists u \leq d, E_k^u(P) = P$. As in the classical case, such a procedure will make successive calls to E_k, in total d of them, running in overall time $\mathcal{O}(d)$ (with d superposition queries to E_k).

In a random permutation, there are on average $1/d$ cycles of length d; and d points of period less than or equal to d. So using Grover's algorithm, one can find a superposition of points that lie on a cycle of length less than d in $\mathcal{O}\left(\sqrt{d2^n}\right)$ calls to E_k. In the classical case, we do not have much choice: the cycle slide pairs found will fall on a large cycle, with high probability. On the contrary, in the quantum setting, we can *specifically* look for points lying on a short cycle. Surprisingly, finding fixed points, or points on very short cycles, costs less than finding points on bigger cycles, due to the cost of iterating E_k. But the smallest the cycle, the least slide pairs we can get from it.

If the attack on f_k needs time $\mathcal{O}(t)$ and $\mathcal{O}(s)$ slide pairs, we conclude that the quantized version of cycle-based sliding costs $\mathcal{O}\left(t + \sqrt{s2^n}\right)$ quantum and classical computations alike. When s becomes large, we arrive at $\mathcal{O}(t + 2^n)$ and there is no improvement w.r.t the classical setting.

To summarize, cycle-based slide attacks seem to be eligible to an interesting quantum speedup when the cipher $E_k = f_k^r$ does not enjoy a slide-shift property as before, but has a sufficiently weak round function f_k, so that a small number of slide pairs suffice to get the subkey material.

Example. For example, this would speed up (but not much) the attack on 24-round GOST of [BOBDK18], since we can tailor the cycle length to our needs.

The attack on f_k requires $2^{36.5}$ slide pairs. Using Grover search, one can find a cycle of this length (or little more) in around $\sqrt{2^{36.5}2^{64}}$) time and queries. This diminishes the amount of work towards around 2^{50} superposition queries instead of 2^{64} classical.

6.3 Quantum Cycle-based Slide Attacks

We are inspired by [BOBDK18] and the attacks against the SA construction and weak variants of AES. In the classical as in the quantum versions, most of the computation time required is due to finding the actual slide pairs (via the cycle).

Two Keys and Two Permutations. Consider a cipher with alternating keys k_0, k_1, xored or modularly added, and two permutations Π_1, Π_2. In the case of a SPN, $\Pi_1 = \Pi_2 = \Pi$ are the same.

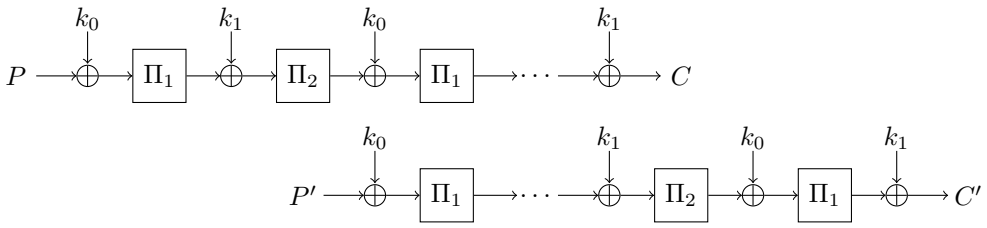

Figure 10: Slide attack against a key- and permutation-alternating cipher

This scheme resists to the basic slide attack, but we can write $E_k \circ \Pi_2 = f_k^r(x)$ where $f_k(x) = \Pi_2(k_1 \oplus \Pi_1(k_0 \oplus x))$, and apply the cycle-finding technique. In $\mathcal{O}\left(2^{n/2}\right)$

superposition queries to E_k and computations, we can recover a small number of slide pairs, say two, from small cycles of $E_k \circ \Pi_2$. Recall that n is the block size here; the key length is $2n$. Therefore we obtain two equations:

$$y = \Pi_2(k_1 \oplus \Pi_1(k_0 \oplus x))$$
$$y' = \Pi_2(k_1 \oplus \Pi_1(k_0 \oplus x'))$$

Since the permutations can be inverted, we find:

$$\Pi_2^{-1}(y) \oplus \Pi_2^{-1}(y') = \Pi_1(k_0 \oplus x) \oplus \Pi_1(k_0 \oplus x')$$

Solving this equation on k_0, if Π_1 has no specific property, can be done in $\mathcal{O}\big(2^{n/2}\big)$ time using Grover's algorithm, the same complexity as the first stage. This improves on the Grover-meets-Simon technique of [LM17], which would perform in $\mathcal{O}\big(n2^{n/2}\big)$ queries and more time (the Grover oracle requires to solve linear systems in superposition).

Attacking 3k-SPN. Cycle-finding can further be applied on a 3k-SPN construction, where there is a unique permutation $\Pi = A \circ S$, with A a linear layer and S a non-linear layer of S-Boxes. Still using $\mathcal{O}\big(2^{n/2}\big)$ queries, we now write the slide equations as:

$$y = \Pi(k_2 \oplus \Pi(k_1 \oplus \Pi(k_0 \oplus x)))$$
$$y' = \Pi(k_2 \oplus \Pi(k_1 \oplus \Pi(k_0 \oplus x')))$$
$$\implies \Pi^{-1}(y) \oplus \Pi^{-1}(y') = \Pi(k_1 \oplus \Pi(k_0 \oplus x)) \oplus \Pi(k_1 \oplus \Pi(k_0 \oplus x'))$$

To solve efficiently this equation in k_0 and k_1, we first guess k_0 using Grover's algorithm. The equation on k_1 becomes:

$$A^{-1}(\Pi^{-1}(y) \oplus \Pi^{-1}(y')) = S(k_1 \oplus \Pi(k_0 \oplus x)) \oplus S(k_1 \oplus \Pi(k_0 \oplus x'))$$

Furthermore, we may consider each S-Box separately and solve the equation on k_1, S-Box by S-Box. if s is the bit size of an S-Box, the final complexity of this attack is $\mathcal{O}\big(2^{(n+s)/2}\big)$ computations, with $\mathcal{O}\big(2^{n/2}\big)$ oracle queries.

Attacking 4k-AES. In the case of AES, we can add one more round. Suppose that, by the cycle, we obtain four equations of the form:

$$A^{-1}(\Pi^{-1}(y) \oplus \Pi^{-1}(y')) = S(k_2 \oplus \Pi(k_1 \oplus \Pi(k_0 \oplus x))) \oplus S(k_2 \oplus \Pi(k_1 \oplus \Pi(k_0 \oplus x')))$$

We use the fact that a column of $\Pi(x)$ does only depend on a diagonal of x. Since we need only to guess k_2 byte per byte, we need also only to guess k_1 column by column, assuming that the full k_0 is guessed. The cycle step has a complexity of approximately 2^{64} queries (usually, queries to an AES-like black-box should cost a non-negligible quantum time). The equation step has a complexity of approximately $2^{64} \times \big(2^{16}(2^4 \times 4) \times 4\big) \simeq 2^{84}$ calls to Π: each guess of k_0 is tested by searching the good k_1 (column by column) and k_2 (byte per byte).

Against 3k-Feistel. A Feistel scheme with a mixing function f, alternating three keys k_0, k_1, k_2, xored or modularly added, is immune to the *complementation slide* and *sliding with a twist* techniques. It seems difficult to write a slide shift property for this cipher. Let us write the round function g as:

$$L, R \mapsto R + f(k_1 + L + f(k_0 + R)), L + f(k_0 + R) + f(k_2 + f(k_1 + L + f(k_0 + R)))$$

and suppose that we can invert f. In $\mathcal{O}\big(2^{n/2}\big)$ queries, we can find two slide equations $g(L, R) = L', R'$, which imply $f(k_1 + L + f(k_0 + R)) = L' - R$. Regardless of the

function f, we can invert it in time $\mathcal{O}\left(2^{n/4}\right)$ using Grover and recover two equations $k_1 + L + f(k_0 + R) = X$. We take the difference (or sum if we replace + by $\oplus$) to eliminate k_1, and we can solve the remaining equation on k_0 using Grover in $\mathcal{O}\left(2^{n/4}\right)$ time. Once this is done, k_1 can be found via the relation $k_1 = f^{-1}(L' - R) - L - f(k_0 + R)$ and k_2 via $L + f(k_0 + R) + f(k_2 + f(k_1 + L + f(k_0 + R))) = R'$.

The whole attack requires $\mathcal{O}\left(2^{n/2}\right)$ time and queries due to the cycle finding, with any function f.

Against 4k-Feistel. If we append one more round key k_3, the round function g becomes:

$$L, R \mapsto L + f(k_0 + R) + f(k_2 + f(k_1 + L + f(k_0 + R))),$$
$$R + f(k_1 + L + f(k_0 + R)) + f(k_3 + L + f(k_0 + R) + f(k_2 + f(k_1 + L + f(k_0 + R))))$$

Again, we can find some slide equations $g(L, R) = L', R'$ from a cycle in $\mathcal{O}\left(2^{n/2}\right)$ queries. We guess the subkey k_0. For each guess, we can rewrite the equations as if there were only 3 subkeys, and solve them in time $\mathcal{O}\left(2^{n/4}\right)$ using multiple Grover instances, as seen above, regardless of the properties of f. The whole attack requires $\mathcal{O}\left(2^{n/2}\right)$ time and queries, the two steps (cycle finding and solving equations) are now balanced. The time complexity is greater than the other 4k-Feistel attacks seen above, but there is no restriction on the function f and the operations used; furthermore, we only use encryption queries, not decryption queries (which is the case of the twist).

7 Conclusion

In this paper, we presented various quantum slide attacks, which target the two main families of block ciphers: SPN and Feistels. The concrete cost of these attacks vastly depends on the structure, that is, is there a slide-shift property or not, and in that case, what is the group law used, whether it is XOR or modular addition (or parallel modular additions); dependencies which do not occur classically. However, for usual parameters in symmetric schemes, slide-shift properties remain extremely competitive when compared to exhaustive search.

We also showed that the relevant security notion for the inner function in a quantum slide-shift attack is its quantum security, which demonstrates a very powerful chaining property of some quantum attacks. We used this chaining property to attack 4-round self-similar Feistel ciphers, at a quadratic cost.

We provided a detailed analysis of cycle-based slide attacks, thought to be quantumly very efficient, showing a smaller than expected improvement with respect to the classical attacks.

Classical slide attacks have shown the importance of a good key schedule, as self-similarity in a cipher allows for powerful breaks. In the quantum setting, these results seem to put even more weight on this design principle, as the attacks become much more efficient. Furthermore, as their cost is very low, a possible future direction for improvement would be to consider new attack patterns, intrinsically infeasible in a classical setting, with stronger functions relating the slide pairs.

Also, using modular additions instead of bitwise additions as suggested in [AR17] does not seem to counter the attacks, as already pointed out in [BNP]. While the quantum complexity of an attack goes from polynomial to subexponential, it would require unrealistic state sizes to assert enough security. Protecting against slide attacks, classical and quantum, using e.g a good key schedule, seems a much more desirable option.

References

[Amb16] Andris Ambainis. Quantum algorithm for element distinctness. In *Encyclopedia of Algorithms*, pages 1646–1651. 2016.

[AR17] Gorjan Alagic and Alexander Russell. Quantum-Secure Symmetric-Key Cryptography Based on Hidden Shifts. In Jean-Sébastien Coron and Jesper Buus Nielsen, editors, *EUROCRYPT (3)*, volume 10212 of *LNCS*, pages 65–93, 2017.

[BHT16] Gilles Brassard, Peter Høyer, and Alain Tapp. *Quantum Algorithm for the Collision Problem*. Springer New York, New York, NY, 2016.

[BNP] Xavier Bonnetain and María Naya-Plasencia. Hidden shift quantum cryptanalysis and implications. To appear in: Advances in Cryptology - ASIACRYPT 2018 - 24th Annual International Conference on the Theory and Application of Cryptology and Information Security, Brisbane, Australia, December 02-06, 2018.

[BOBDK18] Achiya Bar-On, Eli Biham, Orr Dunkelman, and Nathan Keller. Efficient Slide Attacks. *Journal of Cryptology*, 31(3):641–670, Jul 2018.

[Bon18] Xavier Bonnetain. Quantum key-recovery on full AEZ. In *Selected Areas in Cryptography - SAC 2017 - 24th International Conference, Ottawa, ON, Canada, August 16-18, 2017, Revised Selected Papers*, volume 10719 of *Lecture Notes in Computer Science*, pages 394–406. Springer, 2018.

[BW99] Alex Biryukov and David Wagner. Slide Attacks. In Lars R. Knudsen, editor, *Fast Software Encryption, 6th International Workshop, FSE '99, Rome, Italy, March 24-26, 1999, Proceedings*, volume 1636 of *LNCS*, pages 245–259. Springer, 1999.

[BW00] Alex Biryukov and David Wagner. Advanced Slide Attacks. In Bart Preneel, editor, *Advances in Cryptology - EUROCRYPT 2000, International Conference on the Theory and Application of Cryptographic Techniques, Bruges, Belgium, May 14-18, 2000, Proceeding*, volume 1807 of *LNCS*, pages 589–606. Springer, 2000.

[BZ13] Dan Boneh and Mark Zhandry. Secure Signatures and Chosen Ciphertext Security in a Quantum Computing World. In *Advances in Cryptology - CRYPTO 2013 - 33rd Annual Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2013. Proceedings, Part II*, pages 361–379, 2013.

[CNS17] André Chailloux, María Naya-Plasencia, and André Schrottenloher. An efficient quantum collision search algorithm and implications on symmetric cryptography. In Takagi and Peyrin [TP17], pages 211–240.

[DDKS15] Itai Dinur, Orr Dunkelman, Nathan Keller, and Adi Shamir. Reflections on slide with a twist attacks. *Des. Codes Cryptography*, 77(2-3):633–651, 2015.

[DDW18] Xiaoyang Dong, Bingyou Dong, and Xiaoyun Wang. Quantum attacks on some feistel block ciphers. Cryptology ePrint Archive, Report 2018/504, 2018. https://eprint.iacr.org/2018/504.

[DFNS13] Ivan Damgård, Jakob Funder, Jesper Buus Nielsen, and Louis Salvail. Superposition Attacks on Cryptographic Protocols. In Carles Padró, editor, *Information Theoretic Security - 7th International Conference, ICITS 2013,*

Singapore, November 28-30, 2013, Proceedings, volume 8317 of *LNCS*, pages 142–161. Springer, 2013.

[DKS15] Orr Dunkelman, Nathan Keller, and Adi Shamir. Slidex attacks on the even-mansour encryption scheme. *J. Cryptology*, 28(1):1–28, 2015.

[DW17] Xiaoyang Dong and Xiaoyun Wang. Quantum key-recovery attack on feistel structures. Cryptology ePrint Archive, Report 2017/1199, 2017. https://eprint.iacr.org/2017/1199.

[Gag17] Tommaso Gagliardoni. *Quantum Security of Cryptographic Primitives*. PhD thesis, Darmstadt University of Technology, Germany, 2017.

[GHS16] Tommaso Gagliardoni, Andreas Hülsing, and Christian Schaffner. Semantic Security and Indistinguishability in the Quantum World. In Matthew Robshaw and Jonathan Katz, editors, *Advances in Cryptology - CRYPTO 2016 - 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part III*, volume 9816 of *LNCS*, pages 60–89. Springer, 2016.

[HS17] Akinori Hosoyamada and Yu Sasaki. Quantum meet-in-the-middle attacks: Applications to generic feistel constructions. Cryptology ePrint Archive, Report 2017/1229, 2017. https://eprint.iacr.org/2017/1229.

[HSX17] Akinori Hosoyamada, Yu Sasaki, and Keita Xagawa. Quantum multicollision-finding algorithm. In *ASIACRYPT (2)*, volume 10625 of *Lecture Notes in Computer Science*, pages 179–210. Springer, 2017.

[Kap14] Marc Kaplan. Quantum attacks against iterated block ciphers. *CoRR*, abs/1410.1434, 2014.

[KLLN16a] Marc Kaplan, Gaëtan Leurent, Anthony Leverrier, and María Naya-Plasencia. Breaking Symmetric Cryptosystems Using Quantum Period Finding. In Matthew Robshaw and Jonathan Katz, editors, *Advances in Cryptology - CRYPTO 2016 - 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part II*, volume 9815 of *LNCS*, pages 207–237. Springer, 2016.

[KLLN16b] Marc Kaplan, Gaëtan Leurent, Anthony Leverrier, and María Naya-Plasencia. Quantum Differential and Linear Cryptanalysis. *IACR Trans. Symmetric Cryptol.*, 2016(1):71–94, 2016.

[KM10] H. Kuwakado and M. Morii. Quantum distinguisher between the 3-round Feistel cipher and the random permutation. In *Information Theory Proceedings (ISIT), 2010 IEEE International Symposium on*, pages 2682–2685, June 2010.

[KM12] H. Kuwakado and M. Morii. Security on the quantum-type Even-Mansour cipher. In *Information Theory and its Applications (ISITA), 2012 International Symposium on*, pages 312–316, Oct 2012.

[Kup05] Greg Kuperberg. A Subexponential-Time Quantum Algorithm for the Dihedral Hidden Subgroup Problem. *SIAM J. Comput.*, 35(1):170–188, 2005.

[Kup13] Greg Kuperberg. Another Subexponential-time Quantum Algorithm for the Dihedral Hidden Subgroup Problem. In Simone Severini and Fernando G. S. L. Brandão, editors, *8th Conference on the Theory of Quantum Computation,*

Communication and Cryptography, TQC 2013, May 21-23, 2013, Guelph, Canada, volume 22 of *LIPIcs*, pages 20–34. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2013.

[LM17] Gregor Leander and Alexander May. Grover Meets Simon - Quantumly Attacking the FX-construction. In Takagi and Peyrin [TP17], pages 161–178.

[Reg04] Oded Regev. A Subexponential Time Algorithm for the Dihedral Hidden Subgroup Problem with Polynomial Space. *CoRR*, 2004.

[Sim94] Daniel R. Simon. On the Power of Quantum Computation. In *35th Annual Symposium on Foundations of Computer Science, Santa Fe, New Mexico, USA, 20-22 November 1994*, pages 116–123. IEEE Computer Society, 1994.

[TP17] Tsuyoshi Takagi and Thomas Peyrin, editors. *Advances in Cryptology - ASIACRYPT 2017 - 23rd International Conference on the Theory and Applications of Cryptology and Information Security, Hong Kong, China, December 3-7, 2017, Proceedings, Part II*, volume 10625 of *Lecture Notes in Computer Science*. Springer, 2017.

[Zha12] Mark Zhandry. How to Construct Quantum Random Functions. In *53rd Annual IEEE Symposium on Foundations of Computer Science, FOCS 2012, New Brunswick, NJ, USA, October 20-23, 2012*, pages 679–687, 2012.

A Summary of Classical Slide Attacks

We provide in Tables 3 and 4 a (certainly non exhaustive) list of classical slide attacks that we studied for quantum improvements. They are not ordered by efficiency. We refer to the corresponding source for a presentation of the attack principle. Table 3 contains attacks on specific constructions, while Table 4 contains attacks on generic constructions (n is the block size of the cipher attacked; for a Feistel network, round keys have size $n/2$). Note that memory usage and required access to a decryption device play a role in the usefulness of these slide attacks.

Table 3: Classical slide attacks on specific constructions

Cipher attacked	Attack details	Encryption queries	Time	Decryption queries	Memory	Source
2K-DES (64-bit blocks)		2^{32}	2^{50}		2^{32}	[BW99], Section 4
TREYFER (64-bit blocks)		2^{32}	2^{40}		2^{32}	[BW99], Section 5
DESX (64-bit blocks)	Sliding with a twist, known-plaintext only	$2^{32,5}$	$2^{87,5}$		$2^{32,5}$	[BW00], Section 4
GOST (20 rounds)	Sliding with a twist	2^{33}	2^{70}		2^{65}	[BW00], Section 5
2K-AES (128-bit blocks)		2^{69}	2^{69}		2^{69}	[BOBDK18], Section 2.3
3K-AES (128-bit blocks)		2^{81}	2^{81}		2^{81}	[BOBDK18], Section 2.3
24-round GOST with unknown S-Boxes	Slide and truncated differential on 8 rounds	2^{63}	2^{63}		2^{63}	[BOBDK18], Section 5.3
Palindromic GOST with unknown S-Boxes		2^{40}	2^{40}		2^{40}	[BOBDK18], Section 5.4

Table 4: Classical slide attacks on generic constructions. We omit O notations.

Cipher attacked	Remark	Encr. queries	Time	Dec. queries	Memory	Source
One-round self-similar cipher with weak round function	Exhaustive search of slide pairs	$2^{n/2}$	2^n		$2^{n/2}$	[BW99], 2
1k-Feistel	Chosen-plaintext search of slide pairs	$2^{n/4}$	$2^{n/2}$		$2^{n/4}$	[BW99], 3
2k-Feistel	Complementation slide	$2^{n/2}$	$2^{n/2}$		$2^{n/2}$	[BW00], 3.1
2k-Feistel	Sliding with a twist	$2^{n/4}$	$2^{n/4}$	$2^{n/4}$	$2^{n/4}$	[BW00], 3.2
4k-Feistel	Complementation and sliding with a twist	$2^{n/4}$	$2^{n/4}$	$2^{n/4}$	$2^{n/4}$	[BW00], 3.3
Even-Mansour construction	Known-plaintext only	$2^{(n+1)/2}$	$2^{(n+1)/2}$		$2^{(n+1)/2}$	[BW00], 4
1k-SPN		$2^{n/2}$	$2^{n/2}$		$2^{n/2}$	[BOBDK18], 2.1
3k-Feistel		$2^{5n/6}$	$2^{5n/6}$		$2^{2n/3}$	[BOBDK18], 3
Palindromic-scheduled Feistel†		$2^{n/2}$	$2^{n/2}$			[BOBDK18], 4.2
Generic* $E_k = f_k^l$	Cycle structure	2^{n-1}	$t + 2^{n-1}$		2^{n-1}	[BOBDK18], 4.1
2k-WFeistel	Mirror slidex and complementation slide	$2^{n/2}$	$2^{n/2}$	$2^{n/2}$	$2^{n/2}$	[DKS15], 7.3
4k-WFeistel with schedule $(k_a, k_b, k_c, k_d)^m k_a$	Mirror slidex and complementation slide	$2^{n/2}$	$2^{n/2}$	$2^{n/2}$	$2^{n/2}$	[DKS15], 7.4
4k-Feistel	Enhanced reflection	$2^{n/2}$	$2^{n/2}$		n	[DDKS15], 4
4k-WFeistel	Enhanced reflection	$n2^{n/2}$	$n2^{n/2}$		$n2^{n/2}$	[DDKS15], 4

*t is the time needed to attack f_k.
† at best (depends in practice on attacking the palindromic round function)