Quantum Differential and Linear Cryptanalysis

Marc Kaplan, Gaëtan Leurent, Anthony Leverrier, María Naya-Plasencia

Quantum Differential and Linear Cryptanalysis

Marc Kaplan[1,2], Gaëtan Leurent[3], Anthony Leverrier[3] and María Naya-Plasencia[3]

[1] LTCI, Télécom ParisTech, 23 avenue d'Italie, 75214 Paris CEDEX 13, France
[2] School of Informatics, University of Edinburgh,
10 Crichton Street, Edinburgh EH8 9AB, UK
marc.kaplan@telecom-paristech.fr
[3] Inria Paris, France
[anthony.leverrier,gaetan.leurent,maria.naya_plasencia]@inria.fr

Abstract. Quantum computers, that may become available one day, would impact many scientific fields, most notably cryptography since many asymmetric primitives are insecure against an adversary with quantum capabilities. Cryptographers are already anticipating this threat by proposing and studying a number of potentially *quantum-safe* alternatives for those primitives. On the other hand, symmetric primitives seem less vulnerable against quantum computing: the main known applicable result is Grover's algorithm that gives a quadratic speed-up for exhaustive search.
In this work, we examine more closely the security of symmetric ciphers against quantum attacks. Since our trust in symmetric ciphers relies mostly on their ability to resist cryptanalysis techniques, we investigate quantum cryptanalysis techniques. More specifically, we consider quantum versions of differential and linear cryptanalysis. We show that it is usually possible to use quantum computations to obtain a quadratic speed-up for these attack techniques, but the situation must be nuanced: we don't get a quadratic speed-up for all variants of the attacks. This allows us to demonstrate the following non-intuitive result: the best attack in the classical world does not necessarily lead to the best quantum one. We give some examples of application on ciphers LAC and KLEIN. We also discuss the important difference between an adversary that can only perform quantum computations, and an adversary that can also make quantum queries to a keyed primitive.

Keywords: Symmetric cryptography · Differential cryptanalysis · Linear cryptanalysis · Post-quantum cryptography · Quantum attacks · Block ciphers.

1 Introduction

Large quantum computers would have huge consequences in a number of scientific fields. Cryptography would certainly be dramatically impacted: for instance, Shor's factoring algorithm [Sho97] makes asymmetric primitives such as RSA totally insecure in a post-quantum world. Even if quantum computers are unlikely to become widely available in the next couple of years, the cryptographic community has decided to start worrying about this threat and to study its impact. One compelling reason for taking action is that even current pre-quantum long-term secrets are at risk as it seems feasible for a malicious organization to simply store all encrypted data until it has access to a quantum computer. This explains why post-quantum cryptosystems, based for instance on lattices or codes, have become a very hot topic in cryptology, and researchers are now concentrating their efforts in order to provide efficient alternatives that would resist quantum adversaries.

In this paper, we focus on symmetric cryptography, the other main branch of cryptography. Symmetric primitives also suffer from a reduced ideal security in the quantum world,

IACR Transactions on Symmetric Cryptology ISSN 2519-173X, Vol. 2016, No. 1, pp. 71–94
DOI:10.13154/tosc.v2016.i1.71-94

but this security reduction turns out to be much less drastic than for many asymmetric primitives. So far, the main quantum attack on symmetric algorithms follows from Grover's algorithm [Gro96] for searching an unsorted database of size N in $O(N^{1/2})$ time. It can be applied to any generic exhaustive key search, but merely offers a quadratic speed-up compared to a classical attack. Therefore, the current consensus is that key lengths should be doubled in order to offer the same security against quantum algorithms. This was one of the motivations to require a version of AES with a 256-bit key, that appears in the initial recommendations of the European PQCRYPTO project [ABB+15]:

> "Symmetric systems are usually not affected by Shor's algorithm, but they are affected by Grover's algorithm. Under Grover's attack, the best security a key of length n can offer is $2^{n/2}$, so AES-128 offers only 2^{64} post-quantum security. PQCRYPTO recommends thoroughly analyzed ciphers with 256-bit keys to achieve 2^{128} post-quantum security."

Doubling the key length is a useful heuristic, but a more accurate analysis is definitely called for. Unfortunately, little work has been done in this direction. Only recently, a few results have started to challenge the security of some symmetric cryptography constructions against quantum adversaries. In particular, some works have studied generic attacks against symmetric constructions, or attacks against modes of operations.

First, the quantum algorithm of Simon [Sim97], which is based on the quantum Fourier transform, has been used to obtain a quantum distinguisher for the 3-round Feistel cipher [KM10], to break the quantum version of the Even-Mansour scheme [KM12], and in the context of quantum related-key attacks [RS15]. More recently, the same quantum algorithm has been used to break widely used block cipher modes of operations for MACs and authenticated encryption [KLLNP16] (see also [SS16]). All these attacks have a complexity linear in the block size, and show that some constructions in symmetric cryptography are badly broken if an adversary can make quantum queries.

Kaplan [Kap14] has also studied the quantum complexity of generic meet-in-the-middle attacks for iterated block ciphers constructions. In particular, this work shows that having access to quantum devices when attacking double iteration of block ciphers can only reduce the time by an exponent 3/2, rather than the expected quadratic improvement from Grover's algorithm. In consequence, in stark contrast with classical adversaries, double iteration of block ciphers can restore the security against quantum adversaries.

These are important steps in the right direction, providing the quantum algorithms associated to some generic attacks on different constructions. These results also show that the situation is more nuanced than a quadratic speed-up of all classical attacks. Therefore, in order to get a good understanding of the actual security of symmetric cryptography constructions against quantum adversaries, we need to develop and analyze quantum cryptanalytic techniques. In particular, a possible approach to devise new quantum attacks is to *quantize* classical ones.

Security of symmetric key ciphers. While the security of crypto-systems in public key cryptography relies on the hardness of some well-understood mathematical problems, the security of symmetric key cryptography is more heuristic. Designers argue that a scheme is secure by proving its resistance against some particular attacks. This means that only cryptanalysis and security evaluations can bring confidence in a primitive. Even when a primitive has been largely studied, implemented and standardized, it remains vital to carry on with the cryptanalysis effort using new methods and techniques. Examples of standards that turned out to be non-secure are indeed numerous (MD5, SHA1, RC4...). Symmetric security and confidence are therefore exclusively based on this constant and challenging task of cryptanalysis.

Symmetric cryptanalysis relies on a toolbox of classical techniques such as differential or linear cryptanalysis and their variants, algebraic attacks, etc. A cryptanalyst can study the security of a cipher against those attacks, and evaluate the security margin of a design

using reduced-round versions. This security margin (how far the attack is from reaching all the rounds) is a good measure of the security of a design; it can be used to compare different designs and to detect whether a cipher is close to being broken.

Since the security of symmetric primitives relies so heavily on cryptanalysis, it is crucial to evaluate how the availability of quantum computing affects it, and whether dedicated attacks can be more efficient than brute-force attacks based on Grover's algorithm. In particular, we must design the toolbox of symmetric cryptanalysis in a quantum setting in order to understand the security of symmetric algorithms against quantum adversaries. In this paper, we consider quantum versions of cryptanalytic attacks for the first time[1], evaluating how an adversary can perform some of the main attacks on symmetric ciphers with a quantum computer.

Modeling quantum adversaries. Following the notions for PRF security in a quantum setting given by Zhandry [Zha12], we consider two different models for our analysis:

Standard security: a block cipher is *standard secure* against quantum adversaries if no efficient quantum algorithm can distinguish the block cipher from PRP (or a PRF) by making only *classical* queries (later denoted as Q1).

Quantum security: a block cipher is *quantum secure* against quantum adversaries if no efficient quantum algorithm can distinguish the block cipher from PRP (or a PRF) even by making *quantum* queries (later denoted as Q2).

A Q1 adversary collects data classically and processes them with quantum operations, while a Q2 adversary can directly query the cryptographic oracle with a quantum superposition of classical inputs, and receives the superposition of the corresponding outputs. The adversary, in the second model, is very powerful. Nevertheless, it is possible to devise secure protocols against these attacks. In particular, the model was used in [BZ13b], where quantum-secure signatures were introduced. Later, the same authors showed how to construct message authentification codes secure against Q2 adversaries [BZ13a]. It was also investigated in [DFNS13] for secret-sharing schemes. This model is also mathematically well defined, and it is convenient to use it to give security definitions against quantum adversaries, a task that is often challenging [GHS15]. A more practical issue is that even if the cryptographic oracle is designed to produce classical outcomes, its implementation may use some technology, for example optical fibers, that a quantum adversary could exploit. In practice, ensuring that only classical queries are allowed seems difficult, especially in a world in which quantum resources become available. It seems more promising to assume that security against quantum queries is not granted and to study security in this model.

Modes of operation. Block ciphers are typically used in a mode of operation, in order to accommodate messages of variable length and to provide a specific security property (confidentiality, integrity...). In classical cryptography, we prove that modes of operations are secure, assuming that the block cipher is secure, and we trust the block ciphers after enough cryptanalysis has been performed. We can do the same against quantum adversaries, but proofs of security in the classical model do not always translate to proofs of security in the quantum model. In particular, common MAC and AE modes secure in the classical model have recently been broken with a Q2 attack [KLLNP16]. On the other hand, common encryption modes have been proven secure in the quantum model [ATTU16], assuming either a standard-secure PRF or a quantum-secure PRF. In this work, we focus on the security of block ciphers, but this analysis should be combined with an analysis of the quantum security of modes of operation to get a full understanding of the security of symmetric cryptography in the quantum model.

Our results. We choose to focus here on differential cryptanalysis, the truncated differential variant, and on linear cryptanalysis. We give for the first time a synthetic

[1] Previous results as [Kap14, KM10, KM12] only consider quantizing generic attacks.

description of these attacks, and study how they are affected by the availability of quantum computers. As expected, we often get a quadratic speed-up, but not for all attacks.

In this work we use the concept of quantum walks to devise quantum attacks. This framework contains a lot of well known quantum algorithms such as Grover's search or Ambainis' algorithm for element distinctness. More importantly, it allows one to compose these algorithms in the same way as classical algorithms can be composed. In order to keep our quantum attacks as simple as possible, we use a slightly modified Grover's search algorithm that can use quantum checking procedures. This simple trick comes at the cost of constant factors (ignored in our analysis), but a more involved approach, making better use of quantum walks may remove those additional factors.

We prove the following non-obvious results:

- Differential cryptanalysis and linear cryptanalysis usually offer a *quadratic gain* in the Q2 model over the classical model.
- Truncated differential cryptanalysis, however, usually offers *smaller gains* in the Q2 model.
- Therefore, the optimal quantum attack is not always a quantum version of the optimal classical attack.
- In the Q1 model, cryptanalytic attacks might offer *little gain* over the classical model when the key-length is the same as the block length (*e.g.* AES-128).
- But the gain of cryptanalytic attacks in the Q1 model can be quite significant (*similar to the Q2 model*) when the key length is longer (*e.g.* AES-256).

The rest of the paper is organized as follows. We first present some preliminaries on the classical (Section 2) and quantum (Section 3) settings. Section 4 treats differential attacks, while Section 5 deals with truncated differential attacks and Section 6 provides some applications on ciphers LAC and KLEIN. We study linear cryptanalysis in Section 7. In Section 8, we discuss the obtained results. Section 9 concludes the paper and presents some open questions.

2 Preliminaries

In the following, we consider a block cipher E, with a blocksize of n bits, and a keysize of k bits. We assume that E is an iterated design with r rounds, and we use $E^{(t)}$ to denote a reduced version with t rounds (so that $E = E^{(r)}$). When the cipher E is computed with a specific key $\kappa \in \{0,1\}^k$, its action on a block x is denoted by $E_\kappa(x)$. The final goal of an attacker is to find the secret key κ^* that was used to encrypt some data. A query to the cryptographic oracle is denoted $E(x)$, where it is implicitly assumed that E encrypts with the key κ^*, *i.e.*, $E(x) = E_{\kappa^*}(x)$.

Key-recovery attack. The key can always be found using a brute-force attack; following our notations, the complexity of such a generic attack is 2^k. This defines the ideal security, *i.e.* the security a cipher should provide. Therefore, a cipher is considered *broken* if the key can be found "faster" than with the brute-force attack, where "faster" typically means with "less encryptions". Three parameters define the efficiency of a specific attack. The *data complexity* is the number of calls to the cryptographic oracle $E(x)$. The *time complexity* is the time required to recover the key κ^*. We consider that querying the cryptographic oracle requires one unit of time, so that the data complexity is included in the time complexity. The *memory complexity* is the memory needed to perform the attack.

Distinguishers. Another type of attacks, less powerful than key-recovery ones, are distinguishers. Their aim is to distinguish a concrete cipher from an ideal one. A distinguishing attack often gives rise to a key-recovery attack and is always the sign of a weakness of the block cipher.

Our scenario. In this paper, we consider some of the main families of non-generic attacks that can be a threat to some ciphers: differential and linear attacks. We propose their quantized version for the distinguisher and the last-rounds key-recovery variants of linear, simple differentials and truncated differentials. Our aim is to provide a solid first step towards "quantizing" symmetric families of attacks. To reach this objective, due to the technicality of the attacks themselves, and even more due to the technicality of combining them with quantum tools, we consider the most basic versions of the attacks.

Success probability. For the sake of simplicity, in this paper we do not take into account the success probability in the parameters of the attacks. In particular, because it affects in the same way both classical and quantum versions, it is not very useful for the comparison we want to perform. In practice, it would be enough to increase the data complexity by a constant factor to reach any pre-specified success probability. A detailed study of the success probability of statistical attacks can be found in [BGT11].

3 Quantum algorithms

We use a number of quantum techniques in order to devise quantum attacks. Most of them are based on well-known quantum algorithms that have been studied extensively over the last decades. The equivalent to the classical brute-force attack in the quantum world is to search through the key space using a Grover's search algorithm [Gro96], leading to complexity $2^{k/2}$. Our goal is to devise quantum attacks that might be a threat to symmetric primitives by displaying a smaller complexity than the generic quantum exhaustive search.

3.1 Variations on Grover's algorithm

Although Grover's algorithm is usually presented as a search in an unstructured database, we use in our applications the following slight generalization (see [San08] for a nice exposition on quantum-walk-based search algorithms). The task is to find a marked element from a set X. We denote by $M \subseteq X$ the subset of marked elements and assume that we know a lower bound ε on the fraction $|M|/|X|$ of marked elements. A classical algorithm to solve this problem is to repeat $O(1/\varepsilon)$ times: (i) sample an element from X, (ii) check if it is marked.

The cost of this algorithm can therefore be expressed as a function of two parameters: the *Setup cost* S, which is the cost of sampling a uniform element from X, and the *Checking cost* C, which is the cost of checking if an element is marked. The cost considered by the algorithm can be the time or the number of queries to the input. It suffices to consider specifically one of those resources when quantifying the Setup and Checking cost.

Similarly, Grover's algorithm [Gro96] is a quantum search procedure that finds a marked element, and whose complexity can be written as a function of the *quantum Setup cost* S, which is the cost of constructing a uniform superposition of all elements in X, and the *quantum Checking cost* C, which is the cost of applying a controlled-phase gate to the marked elements. Notice that a classical or a quantum algorithm that checks membership to M can easily be modified to get a controlled-phase.

Theorem 1 (Grover)**.** *There exists a quantum algorithm which, with high probability, finds a marked element, if there is any, at cost of order* $\frac{\mathrm{S}+\mathrm{C}}{\sqrt{\varepsilon}}$.

In particular, the setup and the checking steps can themselves be quantum procedures. Assume for instance that the set X is itself a subset of a larger set $\tilde{X}$. Grover's algorithm can then find an element $x \in X$ at a cost $(\tilde{X}/X)^{1/2}$, assuming that the setup and checking procedures are easy. Moreover, a closer look at the algorithm shows that if one ignores the final measurement that returns one element, the algorithm produces a uniform superposition of the elements in X, which can be used to setup another Grover search.

Grover's algorithm can also be written as a special case of amplitude amplification, a quantum technique introduced by Brassard, Høyer and Tapp in order to boost the success probability of quantum algorithms [BHMT02]. Intuitively, assume that a quantum algorithm $\mathcal{A}$ produces a superposition of outputs in a good subspace G and outputs in a bad subspace B. Then there exists a quantum algorithm that calls $\mathcal{A}$ as a subroutine to amplify the amplitude of good outputs.

If $\mathcal{A}$ was a classical algorithm, repeating it $\Theta(1/a)$, where a is the probability of producing a good output, would lead to a new algorithm with constant success probability. Just as Grover's algorithm, the amplitude amplification technique achieves the same result with a quadratic improvement [BHMT02]. The intuitive reason is that quantum operations allow to amplify the amplitudes of good output states, and that the corresponding probabilities are given by the squares of the amplitudes. Therefore, the amplification is quadratically faster than in the classical case.

Theorem 2 (Amplitude amplification)**.** *Let $\mathcal{A}$ be a quantum algorithm that, with no measurement, produces a superposition $\sum_{x\in G}\alpha_x|x\rangle + \sum_{y\in B}\alpha_y|y\rangle$. Let $a = \sum_{x\in G}|\alpha_x|^2$ be the probability of obtaining, after measurement, a state in the* good *subspace G.*

Then, there exists a quantum algorithm that calls $\mathcal{A}$ and $\mathcal{A}^{-1}$ as subroutines $\Theta(1/\sqrt{a})$ times and produces an outcome $x \in G$ with a probability at least $\max(a, 1-a)$.

A variant of quantum amplification amplitude can be used to count approximately, again with a quadratic speed-up over classical algorithms [BHT98].

Theorem 3 (Quantum counting)**.** *Let $F : \{0, \dots N-1\} \to \{0, 1\}$ be a Boolean function, and $p = |F^{-1}(1)|/N$. For every positive integer D, there is a quantum algorithm that makes D queries to F and, with probability at least $8/\pi^2$, outputs an estimate p' to p such that $|p - p'| \leq 2\pi\sqrt{p}/D + \pi^2/D^2$.*

3.2 Quantum search of pairs

We also use Ambainis' quantum algorithm for the element distinctness problem. In our work, we use it to search for collisions.

Theorem 4 (Ambainis [Amb07])**.** *Given a list of numbers $x_1, \dots, x_n$, there exists a quantum algorithm that finds, with high probability, a pair of indices (i, j) such that $x_i = x_j$, if there exists one, at a cost $O(n^{2/3})$.*

The quantum algorithm proposed by Ambainis can easily be adapted to finding a pair satisfying $x_i + x_j = w$ for any given w (when the x_i's are group elements and the "+" operation can be computed efficiently).

Ambainis' algorithm can also be adapted to search in a list $\{x_1, \dots, x_n\}$ for a pair of indices (i, j) such that (x_i, x_j) satisfies some relation R, with the promise that the input contains at least k possible pairs satisfying R. If the input of the problem is a uniformly random set of pairs, it is sufficient, in order to find one, to run Ambainis' algorithm on a smaller random subset of inputs.

Theorem 5. *Consider a list of numbers $x_1, \dots, x_n$ with $x_i \in X$ and a set of pairs $\mathcal{D} \subset X \times X$ such that $\mathcal{D}$ contains exactly k pairs. There exists a quantum algorithm that finds, with high probability, a pair (i, j) such that $(x_i, x_j) \in \mathcal{D}$, at a cost $O(n^{2/3}k^{-1/3})$ on average over uniformly distributed inputs.*

Proof. For a uniformly chosen subset $X' \subset X$ such that $|X'| = n/\sqrt{k}$, there is, with constant probability, at least one pair from $\mathcal{D}$ in $X' \times X'$. According to Theorem 4, the cost of finding this pair is $O(n^{2/3}k^{-1/3})$. Therefore, the quantum algorithm starts by sampling a random X' and then runs Ambainis' algorithm on this subset.

Table 1: Notations used in the attacks.

n	block-size
k	key-size
Δ_{in}	size (log) of the set of input differences
Δ_{out}	size (log) of the set of output differences
Δ_{fin}	size (log) of the set of differences $\mathcal{D}_{\text{fin}}$ after last rounds
h_S	probability ($-\log$) of the differential characteristic ($h_S < n$)
h_T	probability ($-\log$) of the truncated differential characteristic
h_{out}	probability ($-\log$) of generating δ_{out} from $\mathcal{D}_{\text{fin}}$
k_{out}	number of key bits required to invert the last rounds
$C_{k_{\text{out}}}$	cost of recovering the last round subkey from a good pair
$C^*_{k_{\text{out}}}$	quantum cost of recovering the last round subkey from a good pair
ε	bias of the linear approximation
ℓ	number of linear approximations (Matsui's algorithm 1)

Notice that if the algorithm runs on uniformly random inputs, the set X' does not need to be itself chosen at random. Any sufficiently large subset will contain one of the pairs with high probability, with high probability over the distribution of inputs.

Before ending this section on quantum algorithms, we make a remark on the outputs produced by quantum-walk-based algorithms, such as Ambainis' or Grover's algorithm. In our applications, we use these not necessarily to produce some output, but to prepare a superposition of the outputs. Similarly to Grover's algorithm, this can be done by running the algorithm without performing the final measurement. However, since Ambainis' algorithm uses a quantum memory to maintain some data structure, the superposition could in principle include the data from the memory. This issue does not happen with Grover's algorithm precisely because it does not require any data structure.

In our case, the algorithm ends in a superposition of nodes containing at least one of the searched pairs. It has no consequence for our application, because we are nesting this procedure in Grover's algorithm. Alternatively, it is possible to use amplitude amplification afterwards in order to amplify the amplitude on the good nodes. However, this could be an issue when nesting our algorithm in an arbitrary quantum algorithm. For a discussion on nested quantum walks, see [JKM13].

4 Differential Cryptanalysis

Differential cryptanalysis was introduced in [BS90] by Biham and Shamir. It studies the propagation of differences in the input of a function (δ_{in}) and their influence on the generated output difference (δ_{out}). In this section, we present the two main types of differential attacks on block ciphers in the classical world: the *differential distinguisher* and the *last-rounds attack*, and then analyze their complexities for quantum adversaries.

4.1 Classical Adversary

Differential attacks exploit the fact that there exists an input difference δ_{in} and an output difference δ_{out} to a cipher E such that

$$h_S := -\log \Pr_x[E(x \oplus \delta_{\text{in}}) = E(x) \oplus \delta_{\text{out}}] < n, \tag{1}$$

i.e., such that we can detect some non-random behaviour of the differences of plaintexts x and $x \oplus \delta_{\text{in}}$. Here, "$\oplus$" represents the bitwise xor of bit strings of equal length. The

value of h_S is generally computed for a random key, and as usual in the literature, we will assume that Eq. (1) approximately holds for the secret key κ^*. Such a relation between δ_{in} and δ_{out} is typically found by studying the internal structure of the primitive in detail. While it seems plausible that a quantum computer could also be useful to find good pairs $(\delta_{\text{in}}, \delta_{\text{out}})$, we will not investigate this problem here, but rather focus on attacks that can be mounted once a good pair satisfying Eq. (1) is given.

4.1.1 Differential Distinguisher

This non-random behaviour can already be used to attack a cryptosystem by distinguishing it from a random function. This distinguisher is based on the fact that, for a random function and a fixed δ_{in}, obtaining the δ_{out} difference in the output would require 2^n trials, where n is the size of the block. On the other hand, for the cipher E, if we collect 2^{h_S} input pairs verifying the input difference δ_{in}, we can expect to obtain one pair of outputs with output difference δ_{out}. The complexity of such a distinguisher exploiting Eq. (1) is 2^{h_S+1} in both data and time, and is negligible in terms of memory:

$$T_{\text{C}}^{\text{s. dist.}} = D_{\text{C}}^{\text{s. dist.}} = 2^{h_S+1}. \tag{2}$$

Here, the subscript C refers to classical and *s. dist.* to "simple distinguisher" by opposition to its truncated version later in the text.

Assuming that such a distinguisher exists for the first R rounds of a cipher, we can transform the attack into a key recovery on more rounds by adding some rounds at the end or beginning of the cipher. This is called a *last-rounds attack*, and allows to attack more rounds than the distinguisher, typically one or two, depending on the cipher.

4.1.2 Last-Rounds Attack

For simplicity and without loss of generality, we consider that the rounds added to the distinguisher are placed at the end. We attack a total of $r = R + r_{\text{out}}$ rounds, where R are the rounds covered by the distinguisher. The main goal of the attack is to reduce the key space that needs to be searched exhaustively from 2^k to some $2^{k'}$ with $k' < k$. For this, we use the fact that we have an advantage for finding an input x such that $E^{(R)}(x) \oplus E^{(R)}(x \oplus \delta_{\text{in}}) = \delta_{\text{out}}$.

For a pair that generates the difference δ_{out} after R rounds, we denote by $\mathcal{D}_{\text{fin}}$ the set of possible differences generated in the output after the final r_{out} rounds, the size of this set by $2^{\Delta_{\text{fin}}} = |\mathcal{D}_{\text{fin}}|$. Let $2^{-h_{\text{out}}}$ denote the probability of generating the difference δ_{out} from a difference in $\mathcal{D}_{\text{fin}}$ when computing r_{out} rounds in the backward direction, and by k_{out} the number of key bits involved in these rounds. The goal of the attack is to construct a list L of candidates for the partial key that contains almost surely the correct value, and that has size strictly less than $2^{k_{\text{out}}}$. For this, one starts with lists L_M and L_K where L_M is a random subset of 2^{h_S} possible messages and L_K contains all possible k_{out}-bit strings. From Eq. (1), the list L_M contains an element x such that $E^{(R)}(x) \oplus E^{(R)}(x \oplus \delta_{\text{in}}) = \delta_{\text{out}}$ with high probability. Let us apply two successive tests to the lists.

The first test keeps only the $x \in L_M$ such that $E(x) \oplus E(x \oplus \delta_{\text{in}}) \in \mathcal{D}_{\text{fin}}$. The probability of satisfying this equation is $2^{\Delta_{\text{fin}}-n}$. This gives a new list L'_M of size $|L'_M| = 2^{h_S+\Delta_{\text{fin}}-n}$. The cost of this first test is 2^{h_S+1}.

The second test considers the set $L'_M \times L_K$ and keeps only the couples (x, κ) such that $E_\kappa^{(R)}(x) + E_\kappa^{(R)}(x + \delta_{in}) = \delta_{\text{out}}$. This is done by computing backward the possible partial keys for a given difference in $\mathcal{D}_{\text{out}}$. Denote $C_{k_{\text{out}}}$ the average cost of generating those keys for a given input pair. Notice that $C_{k_{\text{out}}}$ can be 1 when the number of rounds added is reasonably small[2], and is upper bounded by $2^{k_{\text{out}}}$, that is, $1 \leq C_{k_{\text{out}}} \leq 2^{k_{\text{out}}}$. For a random

[2] For example, using precomputation tables with the values that allow the differential transitions through the S-Boxes.

pair (x, κ), the probability of passing this test is $2^{-h_{\text{out}}}$. The size of the resulting set is therefore expected to be $2^{-h_{\text{out}}} \times |L'_M| \times |L_K| = 2^{h_S+\Delta_{\text{fin}}-n+k_{\text{out}}-h_{\text{out}}}$. The cost of this step is $C_{k_{\text{out}}} 2^{h_S+\Delta_{\text{fin}}-n}$.

The previous step produces a list of candidates for the partial key corresponding to the key bits involved in the last r_{out} rounds and leading to a difference δ_{out} after R rounds. The last step of the attack consists in performing an exhaustive search within all partial keys of this set completed with all possible $k - k_{\text{out}}$ bits. The cost of this step is $2^{h_S+\Delta_{\text{fin}}-n+k-h_{\text{out}}}$.

In practice, the lists do not need to be built and everything can be performed "on the fly". Consequently, memory needs can be made negligible. The total time complexity is:

$$T_{\text{C}}^{\text{s. att.}} = 2^{h_S+1} + 2^{h_S+\Delta_{\text{fin}}-n} \left(C_{k_{\text{out}}} + 2^{k-h_{\text{out}}} \right), \tag{3}$$

while the data complexity of this classical attack is $D_{\text{C}}^{\text{s. att.}} = 2^{h_S+1}$. The attack is more efficient than an exhaustive search if $T_{\text{C}}^{\text{s. att.}} < 2^k$.

4.2 Quantum Adversary

We first give attacks in the Q2 model, using superposition queries.

4.2.1 Differential Distinguisher in the Q2 model

The distinguisher consists in applying a Grover search over the set of messages X, of size 2^n. More precisely, the algorithm makes $2^{h_S/2+1}$ queries to the encryption cipher, trying to find a marked element $x \in M = \{x \in X \,:\, E(x \oplus \delta_{\text{in}}) = E(x) \oplus \delta_{\text{out}}\}$. If it finds any, it outputs "concrete". If it does not, it outputs "random".

With the notations of the previous sections, the fraction of marked elements is $\varepsilon = 2^{-h_S}$ and Grover's algorithm finds a marked element after $\frac{1}{\sqrt{\varepsilon}} = 2^{h_S/2}$ iterations, each one requiring two queries to the encryption cipher. The time and data complexities are:

$$T_{\text{Q2}}^{\text{s. dist.}} = D_{\text{Q2}}^{\text{s. dist.}} = 2^{h_S/2+1}. \tag{4}$$

It remains to prove that in the case of a random function, the probability of finding a marked element is negligible. Assume that the probability of finding a marked element after $2^{h_S/2}$ quantum queries is δ. Then, this can be wrapped into an amplitude amplification procedure (Theorem 2), leading to a bounded error algorithm making $(1/\sqrt{\delta})2^{h_S/2}$ queries. Since Grover's algorithm is optimal, we get that $(1/\sqrt{\delta})2^{h_S/2} \geq 2^{n/2}$, leading to $\delta \leq 2^{h_S-n}$.

4.2.2 Last-Rounds Attack in the Q2 model

An important point of the attack in the Q2 model is that it should avoid creating lists. Instead, the algorithm queries the cryptographic algorithm whenever it needs to sample an element from the list.

The quantum attack can be described as a Grover search, with quantum procedures for the setup and checking phases. The algorithm searches in the set $X = \{x : E(x \oplus \delta_{\text{in}}) \oplus E(x) \in \mathcal{D}_{\text{fin}}\}$ for a message such that $E^{(R)}(x \oplus \delta_{\text{in}}) \oplus E^{(R)}(x) = \delta_{\text{out}}$. This procedure outputs a message and when it is found, it suffices to execute the sequence corresponding to the checking of the Grover search once more: generate partial key candidates and search among them, completed with all possible remaining $k - k_{\text{out}}$ bits. This outputs the correct key and only adds a constant overhead factor to Grover search. Notice that using tailor-made quantum walks, it should be possible to suppress this overhead. Here we use Grover search to keep the attacks as simple as possible.

The setup phase prepares a uniform superposition of the $x \in X$; this costs $\text{S} = 2^{(n-\Delta_{\text{fin}})/2}$ using Grover's algorithm. The checking phase takes a value x and must determine whether $(x, x \oplus \delta_{\text{in}})$ is a good pair; it consists of the following successive steps:

1. Compute all possible partial keys κ_{out} for the k_{out} bits that intervene in the last r_{out} rounds, assuming that $(x, x \oplus \delta_{\text{in}})$ is a good pair $(E^{(R)}(x \oplus \delta_{\text{in}}) \oplus E^{(R)}(x) = \delta_{\text{out}})$;
2. Complete the key by searching exhaustively using a Grover search, checking if the obtained key is the correct one.

The cost of computing all possible partial keys is $C^*_{k_{\text{out}}}$. The number of partial keys is $2^{k_{\text{out}}-h_{\text{out}}}$, then completed by $k - k_{\text{out}}$ remaining bits. The cost of checking through all of them is thus $\mathrm{C} = C^*_{k_{\text{out}}} + 2^{(k-h_{\text{out}})/2}$.

The procedure succeeds whenever a message x is found such that $E^{(R)}(x) \oplus E^{(R)}(x \oplus \delta_{\text{in}}) = \delta_{\text{out}}$. Therefore, the probability of finding a marked element is lower bounded by $\varepsilon \geq 2^{-h_S-\Delta_{\text{fin}}+n}$. This is the conditional probability of getting $E^{(R)}(x \oplus \delta_{\text{in}}) \oplus E^{(R)}(x) = \delta_{\text{out}}$ given that the output difference is in $\mathcal{D}_{\text{fin}}$.

The total cost of the attack in the Q2 model is:

$$T^{\text{s. att.}}_{\text{Q2}} = 2^{h_S/2+1} + 2^{(h_S+\Delta_{\text{fin}}-n)/2}\left(C^*_{k_{out}} + 2^{(k-h_{\text{out}})/2}\right), \quad D^{\text{s. att.}}_{\text{Q2}} = 2^{h_S/2+1}, \tag{5}$$

with a data complexity identical to that of the distinguisher.

4.2.3 Last-Rounds Attack in the Q1 model

We can also have a speed-up for the last-round attack in the Q1 model. In this model, the quantum operations only take place after a classical acquisition of the data. In particular, the data complexity will be the same as for a classical adversary. After the first filtering step of the classical last-round attack, $2^{h_S-n+\Delta_{\text{fin}}}$ couples satisfying $E(x) \oplus E(x \oplus \delta_{\text{in}}) \in \mathcal{D}_{\text{fin}}$ are obtained. The attacker then uses a quantum algorithm to generate the partial keys κ_{out}, and a Grover search among those, completed with all possible remaining $k - k_{\text{out}}$ bits of the key, in order to find the key. This leads to data and time complexities of:

$$D^{\text{s. att.}}_{\text{Q1}} = 2^{h_S+1}, \quad T^{\text{s. att.}}_{\text{Q1}} = 2^{h_S+1} + 2^{(h_S+\Delta_{\text{fin}}-n)/2}\left(C^*_{k_{out}} + 2^{(k-h_{\text{out}})/2}\right), \tag{6}$$

where $C^*_{k_{\text{out}}}$ denotes the average time complexity of generating the partial keys of length k_{out}, on a *quantum* computer.

Let us point out that any classical attack with data complexity smaller than square root of the exhaustive search of the key can be translated into an effective attack also in the Q1 model. This is more likely to happen for larger keys, where the limiting terms are often the second and third terms of $T^{\text{s. att.}}_{\text{Q1}}$ in Eq. (6). See a detailed example in 6.2. The fact that long keys are more likely to "maintain" the validity of the attacks is an interesting result, as longer keys correspond to the recommendations for post-quantum symmetric primitives. In these cases, the Q1 model is particularly meaningful.

4.2.4 Generating partial keys on a quantum computer

We investigate further the average cost $C_{k_{\text{out}}}$ of generating the partial keys compatible with some input x such that $E(x) \oplus E(x \oplus \delta_{\text{in}}) \in \mathcal{D}_{\text{fin}}$. These partial keys correspond to the key bits involved in a transition from $\mathcal{D}_{\text{fin}}$ to δ_{out}.

Using a classical computer and a precomputation table, this usually takes constant time, but can be up to $2^{k_{\text{out}}}$ in the worst case. It turns out that the worst case this can be sped up using a quantum computer.

Let $K = 2^{k_{\text{out}}-h_{\text{out}}}$ be the average number of partial key candidates compatible with some input x, and denote $N = 2^{k_{\text{out}}}$. Finding one partial key can be done using Grover search with $(N/K)^{1/2}$ steps. The cost of finding a second one is $(N/(K-1))^{1/2}$, and so on. This leads to the following upper bound on $C^*_{k_{\text{out}}}$, the quantum version of $C_{k_{\text{out}}}$:

$$C^*_{k_{\text{out}}} \leq \sqrt{N}\sum_{i=1}^{K} i^{-1/2} \leq \quad 2\sqrt{NK}.$$

Replacing with our parameters, this gives $C^*_{k_{\text{out}}} \leq 2^{k_{\text{out}}-h_{\text{out}}/2+1}$.

5 Truncated Differential Cryptanalysis

Truncated differential cryptanalysis was introduced by Knudsen [Knu94] in 94. Instead of fixed input and output differences, it considers sets of differences (like the differences in the output in the last-rounds attack that we have considered in the previous section).

We assume in the following that we are given two sets $\mathcal{D}_{\text{in}}$ and $\mathcal{D}_{\text{out}}$ of input and output differences such that the probability of generating a difference in $\mathcal{D}_{\text{out}}$ from one in $\mathcal{D}_{\text{in}}$ is 2^{-h_T}. We further consider that $\mathcal{D}_{\text{in}}$ and $\mathcal{D}_{\text{out}}$ are vector spaces.

5.1 Classical Adversary

As in the simple differential case, we first present the differential distinguisher based on the non-random property of the differences behaviour, and then discuss the last-rounds attack obtained from the truncated differential distinguishers.

5.1.1 Truncated Differential Distinguisher

Let $2^{\Delta_{\text{in}}}$ and $2^{\Delta_{\text{out}}}$ denote the sizes of the input and output sets of differences, respectively. For simplicity and without loss of generality, we assume to have access to an encryption oracle, and therefore only consider the truncated differential as directed from input to output[3]. We denote by 2^{-h_T} the probability of generating a difference in $\mathcal{D}_{\text{out}}$ from one in $\mathcal{D}_{\text{in}}$. The condition for the distinguisher to work is that $2^{-h_T} > 2^{\Delta_{\text{out}}-n}$. In this analysis, we assume that $2^{-h_T} \gg 2^{\Delta_{\text{out}}-n}$.

The advantage of truncated differentials is that they allow the use of structures, *i.e.*, sets of plaintext values that can be combined into input pairs with a difference in $\mathcal{D}_{\text{in}}$ in many different ways: one can generate $2^{2\Delta_{\text{in}}-1}$ pairs using a single structure of size $2^{\Delta_{\text{in}}}$. This reduces the data complexity compared to simple differential attacks.

Two cases need to be considered. If $\Delta_{\text{in}} \geq (h_T+1)/2$, we build a single structure $\mathcal{S}$ of size $2^{(h_T+1)/2}$ such that for all pairs $(x, y) \in \mathcal{S} \times \mathcal{S}$, $x \oplus y \in \mathcal{D}_{\text{in}}$. This structure generates 2^{h_T} pairs. If $\Delta_{\text{in}} \leq (h_T+1)/2$, we have to consider multiple structures $\mathcal{S}_i$. Each structure contains $2^{\Delta_{\text{in}}}$ elements, and generates $2^{2\Delta_{\text{in}}-1}$ pairs of elements. We consider $2^{h_T-2\Delta_{\text{in}}+1}$ such structures in order to have 2^{h_T} candidate pairs.

In both cases, we have 2^{h_T} candidate pairs. With high probability, one of these pairs shall satisfy $E(x) \oplus E(y) \in \mathcal{D}_{\text{out}}$, something that should not occur for a random function if $2^{-h_T} \gg 2^{\Delta_{\text{out}}-n}$. Therefore detecting a single valid pair gives an efficient distinguisher.

The attack then works by checking if, for a pair generated by the data, the output difference belongs to $\mathcal{D}_{\text{out}}$. Since $\mathcal{D}_{\text{out}}$ is assumed to be a vector space, this can be reduced to trying to find a collision on $n - \Delta_{\text{out}}$ bits of the output. Once the data is generated, looking for a collision is not expensive (*e.g.* using a hash table), which means that time and data complexities coincide:

$$D_{\text{C}}^{\text{tr. dist.}} = \max\{2^{(h_T+1)/2}, 2^{h_T-\Delta_{\text{in}}+1}\}, \quad T_{\text{C}}^{\text{tr. dist.}} = \max\{2^{(h_T+1)/2}, 2^{h_T-\Delta_{\text{in}}+1}\}. \quad (7)$$

5.1.2 Last-Rounds Attack

Last-rounds attacks work similarly as in the case of simple differential cryptanalysis. For simplicity, we assume that r_{out} rounds are added at the end of the truncated differential. The intermediate set of differences is denoted $\mathcal{D}_{\text{out}}$, and its size is $2^{\Delta_{\text{out}}}$. The set $\mathcal{D}_{\text{fin}}$,

[3] In the case where the other direction provides better complexities, we could instead perform queries to a decryption oracle and change the roles of input and output in the attack. We assume that the most interesting direction has been chosen.

of size $2^{\Delta_{\text{fin}}}$ denotes the possible differences for the outputs after the final round. The probability of reaching a difference in $\mathcal{D}_{\text{out}}$ from a difference in $\mathcal{D}_{\text{in}}$ is 2^{-h_T}, and the probability of reaching a difference in $\mathcal{D}_{\text{out}}$ from a difference in $\mathcal{D}_{\text{fin}}$ is $2^{-h_{\text{out}}}$. Applying the same algorithm as in the simple differential case, the data complexity remains the same as for the distinguisher:

$$D_{\text{C}}^{\text{tr. att.}} = \max\{2^{(h_T+1)/2}, 2^{h_T-\Delta_{\text{in}}+1}\}. \tag{8}$$

The time complexity in this case is:

$$T_{\text{C}}^{\text{tr. att.}} = \max\{2^{(h_T+1)/2}, 2^{h_T-\Delta_{\text{in}}+1}\} + 2^{h_T+\Delta_{\text{fin}}-n}\left(C_{k_{\text{out}}} + 2^{k-h_{\text{out}}}\right), \tag{9}$$

where $C_{k_{\text{out}}}$ is the average cost of finding all the partial key candidates corresponding to a pair of data with a difference in $\mathcal{D}_{\text{out}}$. As mentioned earlier, $C_{k_{\text{out}}}$ ranges from 1 to $2^{k_{\text{out}}}$.

5.2 Quantum Adversary

The truncated differential cryptanalysis is similar to the simple differential cryptanalysis, except that $\mathcal{D}_{\text{in}}$ and $\mathcal{D}_{\text{out}}$ are now sets instead of two fixed bit strings.

5.2.1 Truncated Differential Distinguisher

Similarly to simple differential cryptanalysis, the distinguisher can only be more efficient in the Q2 model. This comes from the fact that in both cases, the data complexity is the bottleneck. Since the Q1 model does not provide any advantage over the classical one in data collection, there is no advantage in this model.

We use Ambainis' algorithm for element distinctness, given in Theorem 4, in order to search for collisions inside the structures. If a single structure is involved, the algorithm searches for a pair of messages (x, y) in a set of size $2^{(h_T+1)/2}$, such that $E(x) \oplus E(y) \in \mathcal{D}_{\text{out}}$. Since there is, on average, only one such pair, this can be done using a quantum algorithm with $2^{(h_T+1)/3}$ queries.

If multiple structures are required, the strategy is to search for one structure that contains a pair (x, y) such that $E(x) \oplus E(y) \in \mathcal{D}_{\text{out}}$. This is done with a Grover search on the structure, using Ambainis' algorithm for the checking phase. This returns a structure containing a desired pair, which is sufficient for the distinguisher. The setup cost is constant. The checking step, consisting in searching for a specific pair inside a structure of size $2^{\Delta_{\text{in}}}$, can be done with $\text{C} = 2^{2\Delta_{\text{in}}/3}$ queries. Finally, since there is, with high probability, at least one structure in $2^{h_T-2\Delta_{\text{in}}+1}$ containing a pair such that $E(x) \oplus E(y) \in \mathcal{D}_{\text{out}}$, we get a lower bound on the success probability $\varepsilon \geq 2^{2\Delta_{\text{in}}-h_T-1}$. Using Theorem 1, the total queries complexity is at most $2^{(h_T+1)/2-\Delta_{\text{in}}/3}$.

Combining both results leads to overall data and time complexities given by:

$$D_{\text{Q2}}^{\text{tr. dist.}} = T_{\text{Q2}}^{\text{tr. dist.}} = \max\left\{2^{(h_T+1)/3}, 2^{(h_T+1)/2-\Delta_{\text{in}}/3}\right\}. \tag{10}$$

Similarly to the the quantum simple differential distinguisher, applying the same algorithm to a random function, and stopping it after the same number of queries only provides a correct answer with negligible probability.

5.2.2 Last-Rounds Attack in the Q1 model

As seen in Section 5.1.2, last-round attacks for truncated differential cryptanalysis are very similar to attacks with a simple differential. The attack in the Q1 model will differ from the attack of Section 4.2.3 only in the first step, when querying the encryption function

with the help of structures. We start by generating a list of 2^{h_T} pairs with differences in $\mathcal{D}_{\text{in}}$, which is done with data complexity:

$$D_{\text{Q1}}^{\text{tr. att.}} = \max\{2^{(h_T+1)/2}, 2^{h_T-\Delta_{\text{in}}+1}\}. \tag{11}$$

The second step is to filter the list of elements to keep only the pairs (x, y) such that $E(x) \oplus E(y) \in \mathcal{D}_{\text{fin}}$. Notice that such a filtering can be done at no cost. It suffices to sort the elements according to the values of their image, while constructing the list.

Finally, similarly to the Q1 simple differential attack, a quantum search algorithm is run on the filtered pairs, and the checking procedure consists in generating the partial key candidates completed with $k - k_{\text{out}}$ bits, and searching exhaustively for the key used in the cryptographic oracle. In the Q1 model, the quantum speed-up only occurs in this step.

The average cost of generating the partial keys on a quantum computer is denoted by $C^*_{k_{\text{out}}}$. The average number of partial keys for a given pair of input is $2^{k_{\text{out}}-h_{\text{out}}}$. The fraction ε of marked elements is $\varepsilon = 2^{-h_T-\Delta_{\text{fin}}+n}$, the setup cost is $S = 1$ and the checking cost, a Grover search over the key space, is $\mathrm{C} = C^*_{k_{out}} + 2^{(k-h_{\text{out}})/2}$. This gives a total cost:

$$T_{\text{Q1}}^{\text{tr. att.}} = \max\left\{2^{(h_T+1)/2}, 2^{h_T-\Delta_{\text{in}}+1}\right\} + 2^{(h_T+\Delta_{\text{fin}}-n)/2}\left(C^*_{k_{out}} + 2^{(k-h_{\text{out}})/2}\right). \tag{12}$$

5.2.3 Last-Rounds Attack in the Q2 model

In the Q2 model, we want to avoid building classical lists. Instead, we query the cryptographic oracle each time we need to sample a specific element. This is challenging in the case of truncated differential because the use of structures made of lists is crucial. The idea is to query the elements of the list on the fly.

Assume first that $h_T \leq 2\Delta_{\text{in}} - 1$. Then, it is possible to get 2^{h_T} pairs with differences in $\mathcal{D}_{\text{in}}$ with a single structure, $\mathcal{S}$, of size $2^{(h_T+1)/2}$. The attack runs a Grover search over $X = \{(x, y) \in \mathcal{S} \times \mathcal{S} : E(x) \oplus E(y) \in \mathcal{D}_{\text{fin}}\}$. The checking procedure is the same as for the quantum simple differential attack. For a given a pair of inputs, it generates all possible partial keys, and completes them to try to get the key used by cryptographic oracle. This procedure returns a pair (x, y). The final step is to execute the checking procedure in Grover search once more, suitably modified to return the key given the pair (x, y).

We analyze the setup cost of the attack. To prepare a superposition of the pairs in X, we use a new quantum search algorithm given in Theorem 5. This algorithm searches in a list for a pair of elements with a certain property, considering there exist k such pairs. In our case, the list of elements is $\mathcal{S}$ of size $2^{(h_T+1)/2}$. The total number of elements such that $E(x) \oplus E(y) \in \mathcal{D}_{\text{fin}}$ is therefore $2^{h_T-n+\Delta_{\text{fin}}}$. The algorithm of Theorem 5 prepares a superposition of elements in X in time $\mathrm{S} = 2^{(h_T+1)/3-(h_T-n+\Delta_{\text{fin}})/3} = 2^{(n-\Delta_{\text{fin}}+1)/3}$. The cost of the checking procedure is $\mathrm{C} = C^*_{k_{\text{out}}} + 2^{(k-h_{\text{out}})/2}$, as before. The procedure is successful whenever a pair (x, y) such that $E^{(R)}(x) \oplus E^{(R)}(y) \in \mathcal{D}_{\text{out}}$ is found. Given that the search is among pairs satisfying $x \oplus y \in \mathcal{D}_{\text{in}}$ and $E(x) \oplus E(y) \in \mathcal{D}_{\text{fin}}$, the probability for a pair to be good is $\varepsilon = 2^{-h_T-\Delta_{\text{fin}}+n}$. This gives a total running time:

$$2^{h_T/2-(n-\Delta_{\text{fin}})/6} + 2^{(h_T+\Delta_{\text{fin}}-n)/2}\left(C^*_{k_{\text{out}}} + 2^{(k-h_{\text{out}})/2}\right).$$

Suppose now that multiple structures $\mathcal{S}_i$ of size $2^{\Delta_{\text{in}}}$ are required, where i goes from 1 to $2^{h_T-2\Delta_{\text{in}}+1}$. The search is now over the set $X = \bigcup_i \{(x, y) \in \mathcal{S}_i \times \mathcal{S}_i : E(x) \oplus E(y) \in \mathcal{D}_{\text{fin}}\}$. To get a superposition of the pairs in X, we compose a Grover search over the structure with the algorithm from Theorem 5 inside the structures S_i. This returns a superposition of the pairs in X, together with some additional quantum registers containing the structures the pairs belong to, and the data structure used by our new search algorithm. This additional data does not disturb the Grover search (see Section 3.2). In each structure, the average number of pairs in X is $2^{2\Delta_{\text{in}}-1-n+\Delta_{\text{fin}}}$. The total cost of the setup phase is

therefore $\mathrm{S} = 2^{h_T/3-2\Delta_{\mathrm{in}}/3+2/3+(n-\Delta_{\mathrm{fin}})/3}$. The rest of the attack is similar to the previous case. Putting everything together, the total running time and data complexities of the quantum truncated differential attack in the Q2 model are:

$$T_{\mathrm{Q2}}^{\mathrm{tr.\,att.}} = \max\left\{2^{h_T/2}, 2^{5h_T/6-2\Delta_{\mathrm{in}}/3+2/3}\right\} 2^{-(n-\Delta_{\mathrm{fin}})/6} + 2^{(h_T+\Delta_{\mathrm{fin}}-n)/2}\left(C^*_{k_{\mathrm{out}}} + 2^{(k-h_{\mathrm{out}})/2}\right), \tag{13}$$

$$D_{\mathrm{Q2}}^{\mathrm{tr.\,att.}} = \max\left\{2^{h_T/2}, 2^{h_T-\Delta_{\mathrm{in}}+1}\right\} 2^{-(n-\Delta_{\mathrm{fin}})/6}. \tag{14}$$

6 Applications on existing ciphers

In this section we describe three examples of classical and quantum differential attacks against block ciphers. We have chosen examples of real proposed ciphers where some of the best known attacks are simple variants of differential cryptanalysis. This allows us to illustrate the important counter-intuitive points that we want to highlight, by comparing the best classical attacks and the best quantum attacks. We first consider the block cipher used in the authenticated encryption scheme LAC [ZWW+14], and build for it a classical simple differential distinguisher and a more efficient classical truncated distinguisher. We quantize these attacks, and obtain that the quantum truncated distinguisher performs worse than a generic quantum exhaustive search. In the next application we consider the lightweight block cipher KLEIN [GNL12]. Its 64-bit key version, KLEIN-64, has been recently broken [LN14] by a truncated differential last-rounds attack. When quantizing this attack, we show that it no longer works in the quantum world, and therefore KLEIN-64 is no longer broken. Finally, we consider KLEIN-96 and the best known attack [LN14] against this cipher. We show that its quantum variant still works in the post-quantum world (both in the Q1 and the Q2 models). These applications illustrate what we previously pointed out and believe to be particularly meaningful: block ciphers with longer keys, following the natural recommendation for resisting to generic quantum attacks, are those for which the truncated attacks are more likely to still break the cryptosystem in the postquantum world. Consequently, it is crucial to understand and compute the optimized quantum complexity of the different families of attacks, as we have started doing in this paper.

6.1 Application 1: LAC

We now show an example where a truncated differential attack is more efficient than a simple differential attack using a classical computer, but the opposite is true with a quantum computer.

We consider the reduced version of LBlock [WZ11] used in LAC [ZWW+14]. According to [Leu15], the best known differential for the full 16 rounds has probability $2^{-61.5}$. This yields a classical distinguisher with complexity $2^{62.5}$ and a quantum distinguisher with complexity $2^{31.75}$. The corresponding truncated differential has the following characteristics[4]:

$$n = 64 \qquad \Delta_{\mathrm{in}} = 12 \qquad \Delta_{\mathrm{out}} = 20 \qquad \tilde{h}_T \approx 55.3$$

We note that $\tilde{h}_T > n - \Delta_{\mathrm{out}}$, which is too large to provide a working attack. However, $\tilde{h}_T$ only considers pairs following a given characteristic, and we expect additional pairs to randomly give an output difference in $\mathcal{D}_{\mathrm{out}}$. Therefore, we estimate the probability of the truncated differential as $2^{h_T} = 2^{-44} + 2^{-55.3}$. In order to check this hypothesis, we

[4] We consider the truncated differential with $\mathcal{D}_{\mathrm{in}} =$ 000000000000**0* and $\mathcal{D}_{\mathrm{out}} =$ 0000***00000**00. If the input differential is non-zero on all active bytes, a pair follows the truncated differential when 14 sums of active bytes cancel out, and 3 sums of active bytes don't cancel out. This gives a probability $(15/16)^6 \cdot (1/15)^{14} \approx 2^{-55.3}$.

implemented a reduced version of LAC with 3-bit APN S-Boxes, and verified that a bias can be detected[5]. In every structure, the probability that a pair follows the truncated differential is $2^{23} \cdot 2^{h_T} = 2^{-21} + 2^{-32.3}$, rather than 2^{-21} for a random permutation.

As explained in Section 3 (Theorem 3), this bias can be detected after examining $2 \cdot 2^{-21} \cdot 2^{32.3 \cdot 2} = 2^{44.6}$ structures, *i.e.* $2^{56.6}$ plaintexts in a classical attack (following [BGT11]). In a quantum setting, we use quantum counting [BHT98, Mos98, BHMT02] and examine $4\pi \cdot 2^{-21/2} \cdot 2^{32.3} \approx 2^{25.4}$ structures, for a total cost of $2^{25.4} \cdot 2^{2/3 \cdot 12} = 2^{33.4}$.

To summarize, the best attack in the classical setting is a truncated differential attack (with complexity $2^{60.9}$ rather than $2^{62.5}$ for a simple differential attack), while the best attack in the quantum setting is a simple differential attack (with complexity $2^{31.75}$ rather than $2^{33.4}$ for a truncated differential attack). Moreover, the quantum truncated differential attack is actually less efficient than a generic attack using Grover's algorithm.

6.2 Application 2: KLEIN-64 and KLEIN-96

6.2.1 KLEIN-64

We consider exactly the attack from [LN14]. We omit here the details of the cipher and the truncated differential, but provide the parameters needed to compute the complexity.

When taking into account the attack that provides the best time complexity, we have[6]: $h_T = 69.5$, $\Delta_{\text{in}} = 16$, $\Delta_{\text{fin}} = 32$, $k = 64$, $k_{\text{out}} = 32$, $n = 64$, $C_{k_{\text{out}}} = 2^{20}$ and $h_{\text{out}} = 45$.

In this case, we can recover the time and data complexities from the original result as[7] $D = 2^{54.5}$ and $T = 2^{54.5} + 2^{57.5} + 2^{56.5} = 2^{58.2}$, which is considerably faster than exhaustive search (2^{64}), breaking in consequence the cipher.

In the quantum scenario, the complexity of the generic exhaustive search, which we use to measure the security, is 2^{32}. The cipher is considered broken if we can retrieve the key with smaller complexity. When considering the Q2 or the Q1 case, the two last terms in the time complexity are quadratically accelerated. More precisely, the third is accelerated by square root, the second has a square root in $2^{h_T - n + \Delta_{\text{fin}}}$, which is then multiplied by $C^*_{k_{\text{out}}}$. As shown in Section 4.2.4, $C^*_{k_{\text{out}}}$ is $2^{k_{\text{out}} - h_{\text{out}}/2 + 1} = 2^{11}$ instead of 2^{20}. Consequently, the second term is also completely accelerated by a square root. But this is not the case of the first term, corresponding to data generation. In the Q1 case, it stays the same, being larger than 2^{32} and invalidating the attack. In the Q2 model, the first term becomes $2^{42.6}$, which is also clearly larger than 2^{32}, thus the attack does not work.

We have seen here an example of a primitive broken in the classical world, but remaining secure [8] in the quantum one, for both models.

6.2.2 KLEIN-96

Here we consider the attack of type III given in [LN14], as it is the only one with data complexity lower than 2^{48}, and therefore the only possible candidate for providing also an attack in the Q1 model.

[5] The truncated path for the reduced version has a probability $2^{h_T} = 2^{-33} + 2^{-40.5}$. We ran 32 experiments with 2^{31} structures of 2^9 plaintexts each. With a random function we expect about $2^{31} \cdot 2^9 \cdot (2^9 - 1)/2 \cdot 2^{-33} = 32704$ pairs satisfying the truncated differential, and about 32890 with LAC. The median number of pairs we found is 33050 and it was larger than 32704 is 31 out of 32 experiments. This agrees with our predictions.

[6] For the attacks from [LN14] on KLEIN, h_T is always bigger than $n - \Delta_{\text{in}}$, but the distinguisher from Δ_{in} to Δ_{out} still works exactly as described in Section 5.1.2 because we compare with the probability of producing the truncated differential path and not just the truncated differential.

[7] The slight difference with respect to [LN14] is because here we have not taken into account the relative cost with respect to one encryption, for the sake of simplicity.

[8] We want to point out that notions "not-secure" (i.e. can be attacked in practice) and "broken" (i.e. can be attacked faster than brute-force), are not the same, though they are difficult to dissociate.

The parameters of this classical attack are: $h_T = 78$, $\Delta_{\text{in}} = 32$, $\Delta_{\text{fin}} = 32$, $k_{\text{out}} = 48$, $n = 64$, $C_{k_{\text{out}}} = 2^{30}$ and $h_{\text{out}} = 52$. We compute and obtain the same complexities as the original results in time and data: $D = 2^{47}$ and $T = 2^{47} + 2^{46+30} + 2^{90}$. When quantizing this attack, we have to compare the complexities with $2^{96/2} = 2^{48}$.

In the Q1 model we obtain $2^{47} + 2^{23+23} + 2^{45} = 2^{47.7}$, which is lower that 2^{48}, so the attack still works. The second term comes from $C^*_{k_{\text{out}}} 2^{(h_T - n + \Delta_{\text{out}})/2}$. We can compute $C^*_{k_{\text{out}}}$ as before, obtaining $2^{48-26+1} = 2^{23}$.

In the Q2 model, the first term is reduced to 2^{39} and becomes negligible, with the final complexity at $2^{39} + 2^{46} + 2^{45} = 2^{46.6}$.

7 Linear Cryptanalysis

Linear cryptanalysis was discovered in 1992 by Matsui [MY92, Mat93]. The idea of linear cryptanalysis is to approximate the round function with a linear function, in order to find a linear approximation correlated to the non-linear encryption function E. We describe the linear approximations using linear masks; for instance, an approximation for one round is written as $E^{(1)}(x)[\chi'] \approx x[\chi]$ where χ and χ' are linear masks for the input and output, respectively, and $x[\chi] = \bigoplus_{i:\chi_i=1} x_i$. Here, "$\approx$" means that the probability that the two values are equal is significantly larger than with a random permutation.

The cryptanalyst has to build linear approximations for each round, such that the output mask of a round is equal to the input mask of the next round. The piling-up lemma is then used to evaluate the correlation of the approximation for the full cipher. As for differential cryptanalysis, we assume here that the linear approximation is given and use it with a quantum computer to obtain either a distinguishing attack or a key recovery attack. In this section, we consider linear distinguishers and key recovery attacks following from Matsui's work [Mat93].

7.1 Classical Adversary

7.1.1 Linear distinguisher

In the following, C denotes the ciphertext obtained when encrypting the plaintext P with the key K. We assume that we know a linear approximation with masks (χ_P, χ_C, χ_K) and constant term $\chi_0 \in \{0, 1\}$ satisfying $\Pr\left[C[\chi_C] = P[\chi_P] \oplus K[\chi_K] \oplus \chi_0\right] = (1 + \varepsilon)/2$, with $\varepsilon \gg 2^{-n/2}$; or, omitting the key dependency:

$$\Pr\left[C[\chi_C] = P[\chi_P]\right] = (1 \pm \varepsilon)/2.$$

An attacker can use this to distinguish E from a random permutation. The attack requires $D = A/\varepsilon^2$ known plaintexts P_i and the corresponding ciphertexts C_i, where A is a small constant (*e.g.* $A = 10$). The attacker computes the observed bias $\hat{\varepsilon} = \left|2\#\left\{i : C_i[\chi_C] = P_i[\chi_P]\right\}/D - 1\right|$, and concludes that the data is random if $\hat{\varepsilon} \leq \varepsilon/2$ and that it comes from E otherwise.

If the data is generated by a random permutation, then the expected value of $\hat{\varepsilon}$ is 0, whereas, if it is generated by E, the expected value of $\hat{\varepsilon}$ is ε. We can compute the success probability of the attack assuming that the values of $C_i[\chi_C] \oplus P_i[\chi_P]$ are identically distributed Bernoulli random variables, with parameter $1/2$ or $1/2 \pm \varepsilon$. From Hoeffding's inequality, we get:

$$\Pr\left[\hat{\varepsilon} \geq \varepsilon/2 \middle| \text{random permutation}\right] \leq 2\exp\left(-2\frac{\varepsilon^2}{4^2}D\right) \leq 2\exp\left(-\frac{A}{8}\right),$$

$$\Pr\left[\hat{\varepsilon} \leq \varepsilon/2 \middle| \text{cipher } E\right] \leq \exp\left(-2\frac{\varepsilon^2}{4^2}D\right) \leq \exp\left(-\frac{A}{8}\right);$$

both error terms can also be made arbitrarily small by increasing A.

Overall, the complexity of the linear distinguisher is

$$D_{\mathrm{C}}^{\text{lin. dist.}} = T_{\mathrm{C}}^{\text{lin. dist.}} = 1/\varepsilon^2. \tag{15}$$

As explained in Section 2, we do not take into account the factor A that depends on the success probability, and keep only the asymptotic term in the complexity.

7.1.2 Key-recovery using an r-round approximation (Matsui's Algorithm 1)

The linear distinguisher readily gives one key bit according to the sign of the bias: if $K[\chi_K] = 0$, then we expect $\#\{i : C_i[\chi_C] = P_i[\chi_P] \oplus \chi_0\} > D/2$. The attack can be repeated with different linear approximations in order to recover more key bits. If we have ℓ independent linear approximations $(\chi_P^j, \chi_C^j, \chi_K^j, \chi_0^j)$ with bias at least ε, the total complexity is:

$$D_{\mathrm{C}}^{\mathrm{Mat.1}} = 1/\varepsilon^2, \qquad T_{\mathrm{C}}^{\mathrm{Mat.1}} = \ell/\varepsilon^2 + 2^{k-\ell}. \tag{16}$$

7.1.3 Last-rounds attack (Matsui's Algorithm 2)

Alternatively, linear cryptanalysis can be used in a last-rounds attack that will often be more efficient. Following the notations of the previous sections, we consider a total of $R + r_{\text{out}}$ rounds, with an R-round linear distinguisher $(\chi_P, \chi_{C'})$ with bias ε, and we use partial decryption for the last r_{out} rounds.

We denote by k_{out} the number of key bits necessary to compute $C'[\chi_{C'}]$, where $C' = E^{-r_{\text{out}}}(C)$ from C. The attack proceeds as follows:

1. Initialize a set of $2^{k_{\text{out}}}$ counters $X_{k'}$ to zero, for each key candidate.
2. For each (P, C) pair, and for every partial key guess k', compute C' from C and k', and increment $X_{k'}$ if $P[\chi_P] = C'[\chi_{C'}]$.
3. This gives $X_{k'} = \#\left\{P, C : E_{k'}^{-r_{\text{out}}}(C)[\chi_{C'}] = P[\chi_P]\right\}$.
4. Select the partial key k' with the maximal absolute value of $X_{k'}$.

This gives the following complexity:

$$D_{\mathrm{C}}^{\mathrm{Mat.2}} = 1/\varepsilon^2 \qquad\qquad T_{\mathrm{C}}^{\mathrm{Mat.2}} = 2^{k_{\text{out}}}/\varepsilon^2 + 2^{k-k_{\text{out}}}, \tag{17}$$

where, as before, we neglect constant factors.

We note that this algorithm can be improved using a distillation phase where we count the number of occurrences of partial plaintexts and ciphertexts, and an analysis phase using only these counters rather the full data set. In some specific cases, the analysis phase can be improved by exploiting the Fast Fourier Transform [CSQ07], but we will focus on the simpler case described here.

7.2 Quantum Adversary

7.2.1 Distinguisher in the Q2 model

As in the previous sections, a speed-up for distinguishers is only observed for the Q2 model. The distinguisher is based on the quantum approximate counting algorithm of Theorem 3. As in the classical case, the goal is to distinguish between two Bernoulli distributions with parameter $1/2$ and $1/2 + \varepsilon$, respectively.

Using the quantum approximate counting algorithm, it is sufficient to make $O(1/\varepsilon)$ queries in order to achieve an ε-approximation. The data complexity of the quantum distinguisher is therefore,

$$D_{\mathrm{Q2}}^{\text{lin. dist.}} = T_{\mathrm{Q2}}^{\text{lin. dist.}} = 1/\varepsilon, \tag{18}$$

which constitutes a quadratic speed-up compared to the classical distinguisher.

7.2.2 Key-recovery using an r-round approximation in the Q1 model

Each linear relation allows the attacker to recover a bit of the key using $1/\varepsilon^2$ data, as the classical model. Once ℓ bits of the key have been recovered, one can apply Grover's algorithm to obtain the full key. For ℓ linear relations, the attack complexity is therefore:

$$D_{\text{Q1}}^{\text{Mat.1}} = \ell/\varepsilon^2 \qquad T_{\text{Q1}}^{\text{Mat.1}} = \ell/\varepsilon^2 + 2^{(k-\ell)/2}. \tag{19}$$

7.2.3 Key-recovery using an r-round approximation in the Q2 model

Each linear relation allows the attacker to recover a bit of the key using $1/\varepsilon$ data. If there are ℓ such relations, the attack complexity is:

$$D_{\text{Q2}}^{\text{Mat.1}} = \ell/\varepsilon \qquad T_{\text{Q2}}^{\text{Mat.1}} = \ell/\varepsilon + 2^{(k-\ell)/2}. \tag{20}$$

Note that we do not *a priori* obtain a quadratic improvement for the data complexity compared to the classical model. This is because the same data can be used many times in the classical model, whereas it is unclear whether something similar can be achieved using Grover's algorithm.

7.2.4 Last-rounds attack in the Q1 model

As usual for the Q1 model, one samples the same quantity of data as in the classical model and stores it in a quantum memory. Then the idea is to perform two successive instances of Grover's algorithm: the goal of the first one is to find a partial key of size k_{out} for which a bias ε is detected for the first R rounds: this has complexity $2^{k_{\text{out}}/2}/\varepsilon$ with quantum counting; the second Grover aims at finding the rest of the key and has complexity $2^{(k-k_{\text{out}})/2}$. Overall, the complexity of the attack is

$$D_{\text{Q1}}^{\text{Mat.2}} = 1/\varepsilon^2 \qquad T_{\text{Q1}}^{\text{Mat.2}} = 1/\varepsilon^2 + 2^{k_{\text{out}}/2}/\varepsilon + 2^{(k-k_{\text{out}})/2}. \tag{21}$$

7.2.5 Last-rounds attack in the Q2 model.

The strategy is similar, but the first step of the algorithm, *i.e.* finding the correct partial key, can be improved compared to the Q1 model. One uses a Grover search to obtain the partial key, and the checking step of Grover now consists of performing an approximate counting to detect the bias. Overall, the complexity of the attack is

$$D_{\text{Q2}}^{\text{Mat.2}} = 2^{k_{\text{out}}/2}/\varepsilon \qquad T_{\text{Q2}}^{\text{Mat.2}} = 2^{k_{\text{out}}/2}/\varepsilon + 2^{(k-k_{\text{out}})/2}. \tag{22}$$

8 Discussion

In this section, we first recall all the time complexities obtained through the paper. The data complexities correspond to the first term of each expression for the differential attacks. Next, we discuss how these results affect the post-quantum security of symmetric ciphers with respect to differential and linear attacks. As a remainder, notations are given in Table 1.

Simple Differential Distinguishers:

$$T_{\text{C}}^{\text{s. dist.}} = 2^{h_S+1} \qquad T_{\text{Q2}}^{\text{s. dist.}} = 2^{h_S/2+1}$$

Simple Differential Last-Rounds Attacks:

$$\begin{aligned}
T_{\mathrm{C}}^{\mathrm{s.\,att.}} &= 2^{h_S+1} &&+ 2^{h_S+\Delta_{\mathrm{fin}}-n}\left(C_{k_{\mathrm{out}}} + 2^{k-h_{\mathrm{out}}}\right)\\
T_{\mathrm{Q1}}^{\mathrm{s.\,att.}} &= 2^{h_S+1} &&+ 2^{(h_S+\Delta_{\mathrm{fin}}-n)/2}\left(C^*_{k_{out}} + 2^{(k-h_{\mathrm{out}})/2}\right)\\
T_{\mathrm{Q2}}^{\mathrm{s.\,att.}} &= 2^{h_S/2+1} &&+ 2^{(h_S+\Delta_{\mathrm{fin}}-n)/2}\left(C^*_{k_{out}} + 2^{(k-h_{\mathrm{out}})/2}\right)
\end{aligned}$$

Truncated Differential Distinguishers:

$$T_{\mathrm{C}}^{\mathrm{tr.\,dist.}} = \max\{2^{(h_T+1)/2}, 2^{h_T-\Delta_{\mathrm{in}}+1}\} \quad T_{\mathrm{Q2}}^{\mathrm{tr.\,dist.}} = \max\left\{2^{(h_T+1)/3}, 2^{(h_T+1)/2-\Delta_{\mathrm{in}}/3}\right\}$$

Truncated Differential Last-Rounds Attacks:

$$\begin{aligned}
T_{\mathrm{C}}^{\mathrm{tr.\,att.}} &= \max\left\{2^{(h_T+1)/2}, 2^{h_T-\Delta_{\mathrm{in}}+1}\right\} &&+ 2^{h_T+\Delta_{\mathrm{fin}}-n}\left(C_{k_{\mathrm{out}}} + 2^{k-h_{\mathrm{out}}}\right)\\
T_{\mathrm{Q1}}^{\mathrm{tr.\,att.}} &= \max\left\{2^{(h_T+1)/2}, 2^{h_T-\Delta_{\mathrm{in}}+1}\right\} &&+ 2^{(h_T+\Delta_{\mathrm{fin}}-n)/2}\left(C^*_{k_{out}} + 2^{(k-h_{\mathrm{out}})/2}\right)\\
T_{\mathrm{Q2}}^{\mathrm{tr.\,att.}} &= \max\left\{2^{h_T/2}, 2^{5h_T/6-2\Delta_{\mathrm{in}}/3+2/3}\right\}2^{-(n-\Delta_{\mathrm{fin}})/6} &&+ 2^{(h_T+\Delta_{\mathrm{fin}}-n)/2}\left(C^*_{k_{\mathrm{out}}} + 2^{(k-h_{\mathrm{out}})/2}\right)
\end{aligned}$$

Linear Distinguishers:

$$T_{\mathrm{C}}^{\mathrm{lin.\,dist.}} = 1/\varepsilon^2 \qquad T_{\mathrm{Q2}}^{\mathrm{lin.\,dist.}} = 1/\varepsilon$$

Linear Attacks:

$$\begin{aligned}
T_{\mathrm{C}}^{\mathrm{Mat.1}} &= \ell/\varepsilon^2 + 2^{k-\ell} & T_{\mathrm{C}}^{\mathrm{Mat.2}} &= 2^{k_{\mathrm{out}}}/\varepsilon^2 + 2^{k-k_{\mathrm{out}}}\\
T_{\mathrm{Q1}}^{\mathrm{Mat.1}} &= \ell/\varepsilon^2 + 2^{(k-\ell)/2} & T_{\mathrm{Q1}}^{\mathrm{Mat.2}} &= 1/\varepsilon^2 + 2^{k_{\mathrm{out}}/2}/\varepsilon + 2^{(k-k_{\mathrm{out}})/2}.\\
T_{\mathrm{Q2}}^{\mathrm{Mat.1}} &= \ell/\varepsilon + 2^{(k-\ell)/2} & T_{\mathrm{Q2}}^{\mathrm{Mat.2}} &= 2^{k_{\mathrm{out}}/2}/\varepsilon + 2^{(k-k_{\mathrm{out}})/2}
\end{aligned}$$

The first observation we make is that the cost of a quantum differential or linear attack is at least the square root of the cost of the corresponding classical attack. In particular, if a block cipher is resistant to classical differential and/or linear cryptanalysis (*i.e.* classical attacks cost at least 2^k), it is also resistant to the corresponding quantum cryptanalysis (*i.e.* quantum differential and/or linear attacks cost at least $2^{k/2}$). However, a quadratic speed-up is not always possible with our techniques; in particular truncated attacks might be less accelerated than simple differential ones.

Q1 model *vs* Q2 model. We have studied quantum cryptanalysis with the notion of standard security (Q1 model with only classical encryption queries) and quantum security (Q2 model with quantum superposition queries). As expected, the Q2 model is stronger, and we often have a smaller quantum acceleration in the Q1 model. In particular, the data complexity of attack in the Q1 model is the same as the data complexity of classical attacks. Still, there are important cases where quantum differential or linear cryptanalysis can be more efficient than Grover's search in the Q1 model, which shows that quantum cryptanalysis is also relevant in the more realistic setting with only classical queries.

Quantum differential and linear attacks are more threatening to ciphers with larger key sizes. Though it seems counter-intuitive, the fact is that larger key sizes also mean higher security claims to consider a cipher as secure. In the complexity figure given above, the terms that depend on the key size (the right hand size terms) are likely to be the bottleneck for ciphers with long keys with respect to the internal state size. In all the

attacks studied here, this term is quadratically improved using quantum computation, in both models. Therefore, attacks against those ciphers will get the most benefits from quantum computers. We illustrated this effect in Section 6.2, by studying KLEIN with two different key sizes.

This effect is very strong in the Q1 model because most attacks have a data complexity larger than $2^{n/2}$ (because $h_S > n/2$, $h_T > n/2$, or $\varepsilon < 2^{-n/4}$). If the keysize is equal to n, this makes those attacks less efficient than Grover's search, but they become interesting when k is larger than n. In particular, with $k \geq 2n$, the data complexity is always smaller than $2^{k/2}$.

This observation is particularly relevant because the recommended strategy against quantum adversaries is to use longer keys [ABB+15]. We show that with this strategy, it is likely that classical attacks that break the cryptosystem lead to quantum attacks that also break it, even in the Q1 model where the adversary only makes classical queries to the oracle.

The best attack might change from the classical to the quantum world. Since truncated differential attacks use collision finding in the data analysis step, they do not enjoy a quadratic improvement in the quantum setting. Therefore, as we show in Section 6.1, a truncated differential attack might be the best known attack in the classical world, while the simple differential might become the best in the quantum world. In particular, simply quantizing the best known attack does not ensure obtaining the best possible attack in the post-quantum world, which emphasizes the importance of studying quantum symmetric cryptanalysis.

More strikingly, there are cases where differential attacks are more efficient than brute force in the classical world, but quantum differential attacks are not faster than Grover's algorithm, as we show in the example of Section 6.2.1.

9 Conclusion and open questions

Our work is an important step towards building a quantum symmetric cryptanalysis toolbox. Our results have corroborated our first intuition that symmetric cryptography does not seem ready for the post-quantum world. This not a direct conclusion from the paper, though indirectly the first logical approach for quantum symmetric cryptanalysis would be to quantize the best classical attack, and that would simplify the task. As we know for sure applications where the best attacks might change exist, cryptanalysis must be started anew. The non-intuitive behaviors shown in our examples of applications help to illustrate the importance of understanding how symmetric attacks work in the quantum world, and therefore, of our results. For building trust against quantum adversaries, this work should be extended, and other classical attacks should be investigated. Indeed, we have concluded that quantizing the best known classical differential attacks may not give the best quantum attack. This emphasizes the importance of studying and finding the best quantum attacks, including all known families of cryptanalysis.

We have devised quantum attacks that break classical cryptosystems faster than a quantum exhaustive search. However, the quantum-walk-based techniques used here can only lead to polynomial speed-ups, and the largest gap is quadratic, achieved by Grover's algorithm. Although this is significant, it can not be interpreted as a collapse of cryptography against quantum adversaries similar to public-key cryptography based on the hardness of factoring. However, we already mentioned that attacks based on the quantum Fourier transform, which is at the core of Shor's algorithm for factoring and does not fall in the framework of quantum walks, have been found for symmetric ciphers [KM10, KM12, RS15, KLLNP16].

We end by mentioning a few open questions that we leave for future work. In this work, we have studied quantum versions of differential and linear cryptanalysis. In each of these cases, we were either given a differential characteristics or a linear approximation to begin with, and used quantum algorithms to exploit them to perform a key recovery attack for instance. A natural question is whether quantum computers can also be useful to come up with good differential characteristics or linear approximations in the first place.

So far, we have only scratched the surface of linear cryptanalysis by quantizing the simplest versions of classical attacks, that is excluding more involved constructions using counters or the fast Fourier transform. Of course, since the quantum Fourier transform offers a significant speed-up compared to its classical counterpart, it makes sense to investigate whether it can be used to obtain more efficient quantum linear cryptanalysis.

A major open question in the field of quantum cryptanalysis is certainly the choice of the right model of attack. In this work, we investigated two such models. The Q2 model might appear rather extreme and perhaps even unrealistic since it is unclear why an attacker could access the cipher in superposition. But this model has the advantage of consistency. Also, a cipher secure in this model will remain secure in any setting. On the other hand, the Q1 model appears more realistic, but might be a little bit too simplistic. In particular, it seems important to better understand the interface between the classical register that stores the data that have been obtained by querying the cipher and the quantum register where they must be transferred in order to be further processed by the quantum computer.

References

[ABB+15] Daniel Augot, Lejla Batina, Daniel J Bernstein, Joppe Bos, Johannes Buchmann, Wouter Castryck, Orr Dunkelman, Tim Güneysu, Shay Gueron, Andreas Hülsing, et al. Initial recommendations of long-term secure post-quantum systems. *Available at* http://pqcrypto.eu.org/docs/initial-recommendations.pdf, 2015.

[Amb07] A. Ambainis. Quantum walk algorithm for element distinctness. *SIAM J. Comput.*, 37(1):210–239, 2007.

[ATTU16] Mayuresh Vivekanand Anand, Ehsan Ebrahimi Targhi, Gelo Noel Tabia, and Dominique Unruh. Post-quantum security of the CBC, CFB, OFB, CTR, and XTS modes of operation. In Tsuyoshi Takagi, editor, *Post-Quantum Cryptography - 7th International Workshop, PQCrypto 2016, Fukuoka, Japan, February 24-26, 2016, Proceedings*, volume 9606 of *Lecture Notes in Computer Science*, pages 44–63. Springer, 2016.

[BGT11] Céline Blondeau, Benoît Gérard, and Jean-Pierre Tillich. Accurate estimates of the data complexity and success probability for various cryptanalyses. *Des. Codes Cryptography*, 59(1-3):3–34, 2011.

[BHMT02] Gilles Brassard, Peter Høyer, Michele Mosca, and Alain Tapp. Quantum amplitude amplification and estimation. In *Quantum computation and information (Washington, DC, 2000)*, volume 305 of *Contemp. Math.*, pages 53–74. Amer. Math. Soc., Providence, RI, 2002.

[BHT98] Gilles Brassard, Peter Høyer, and Alain Tapp. Quantum counting. In Kim Guldstrand Larsen, Sven Skyum, and Glynn Winskel, editors, *Automata, Languages and Programming, 25th International Colloquium, ICALP'98, Aalborg, Denmark, July 13-17, 1998, Proceedings*, volume 1443 of *Lecture Notes in Computer Science*, pages 820–831. Springer, 1998.

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

[BZ13a] Dan Boneh and Mark Zhandry. Quantum-secure message authentication codes. In Thomas Johansson and Phong Q. Nguyen, editors, *Advances in Cryptology - EUROCRYPT 2013, 32nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Athens, Greece, May 26-30, 2013. Proceedings*, volume 7881 of *Lecture Notes in Computer Science*, pages 592–608. Springer, 2013.

[BZ13b] Dan Boneh and Mark Zhandry. Secure signatures and chosen ciphertext security in a quantum computing world. In Ran Canetti and Juan A. Garay, editors, *Advances in Cryptology - CRYPTO 2013 - 33rd Annual Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2013. Proceedings, Part II*, volume 8043 of *Lecture Notes in Computer Science*, pages 361–379. Springer, 2013.

[CSQ07] Baudoin Collard, François-Xavier Standaert, and Jean-Jacques Quisquater. Improving the time complexity of matsui's linear cryptanalysis. In Kil-Hyun Nam and Gwangsoo Rhee, editors, *Information Security and Cryptology - ICISC 2007, 10th International Conference, Seoul, Korea, November 29-30, 2007, Proceedings*, volume 4817 of *Lecture Notes in Computer Science*, pages 77–88. Springer, 2007.

[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 *Lecture Notes in Computer Science*, pages 142–161. Springer, 2013.

[GHS15] Tommaso Gagliardoni, Andreas Hülsing, and Christian Schaffner. Semantic security and indistinguishability in the quantum world. *arXiv preprint arXiv:1504.05255*, 2015.

[GNL12] Zheng Gong, Svetla Nikova, and Yee Wei Law. KLEIN: A new family of lightweight block ciphers. In *RFID. Security and Privacy - 7th International Workshop, RFIDSec 2011, Amherst, USA, June 26-28, 2011, Revised Selected Papers*, volume 7055 of *Lecture Notes in Computer Science*, pages 1–18. Springer, 2012.

[Gro96] Lov K. Grover. A fast quantum mechanical algorithm for database search. In Gary L. Miller, editor, *Proceedings of the Twenty-Eighth Annual ACM Symposium on the Theory of Computing, Philadelphia, Pennsylvania, USA, May 22-24, 1996*, pages 212–219. ACM, 1996.

[JKM13] S. Jeffery, R. Kothari, and F. Magniez. Nested quantum walks with quantum data structures. In *Proceedings of 24th AMC-SIAM symposium on discrete algorithms*, 2013.

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

[KLLNP16] Marc Kaplan, Gaëtan Leurent, Anthony Leverrier, and María Naya-Plasencia. Breaking symmetric cryptosystems using quantum period finding. In Matthew Robshaw and Jonathan Katz, editors, *CRYPTO 2016 (to appear)*, Lecture Notes in Computer Science. Springer, 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.

[Knu94] Lars R. Knudsen. Truncated and higher order differentials. In Bart Preneel, editor, *Fast Software Encryption: Second International Workshop. Leuven, Belgium, 14-16 December 1994, Proceedings*, volume 1008 of *Lecture Notes in Computer Science*, pages 196–211. Springer, 1994.

[Leu15] Gaëtan Leurent. Differential forgery attack against LAC. In Orr Dunkelman and Liam Keliher, editors, *Selected Areas in Cryptography - SAC 2015 - 22nd International Conference, Sackville, NB, Canada, August 12-14, 2015, Revised Selected Papers*, volume 9566 of *Lecture Notes in Computer Science*, pages 217–224. Springer, 2015.

[LN14] Virginie Lallemand and María Naya-Plasencia. Cryptanalysis of KLEIN. In Carlos Cid and Christian Rechberger, editors, *Fast Software Encryption - 21st International Workshop, FSE 2014, London, UK, March 3-5, 2014. Revised Selected Papers*, volume 8540 of *Lecture Notes in Computer Science*, pages 451–470. Springer, 2014.

[Mat93] Mitsuru Matsui. Linear cryptanalysis method for DES cipher. In Tor Helleseth, editor, *Advances in Cryptology - EUROCRYPT '93, Workshop on the Theory and Application of of Cryptographic Techniques, Lofthus, Norway, May 23-27, 1993, Proceedings*, volume 765 of *Lecture Notes in Computer Science*, pages 386–397. Springer, 1993.

[Mos98] Michele Mosca. Quantum searching, counting and amplitude amplification by eigenvector analysis. In *MFCS'98 workshop on Randomized Algorithms*, pages 90–100, 1998.

[MY92] Mitsuru Matsui and Atsuhiro Yamagishi. A new method for known plaintext attack of FEAL cipher. In *Advances in Cryptology - EUROCRYPT '92, Workshop on the Theory and Application of of Cryptographic Techniques, Balatonfüred, Hungary, May 24-28, 1992, Proceedings*, pages 81–91, 1992.

[RS15] Martin Roetteler and Rainer Steinwandt. A note on quantum related-key attacks. *Information Processing Letters*, 115(1):40–44, 2015.

[San08] Miklos Santha. Quantum walk based search algorithms. In *Theory and Applications of Models of Computation*, pages 31–46. Springer, 2008.

[Sho97] P. W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM J. Comput.*, 26(5):1484–1509, 1997.

[Sim97] Daniel R Simon. On the power of quantum computation. *SIAM journal on computing*, 26(5):1474–1483, 1997.

[SS16] Thomas Santoli and Christian Schaffner. Using simon's algorithm to attack symmetric-key cryptographic primitives. *arXiv preprint arXiv:1603.07856*, 2016.

[WZ11] Wenling Wu and Lei Zhang. Lblock: A lightweight block cipher. In *Applied Cryptography and Network Security - 9th International Conference, ACNS 2011, Nerja, Spain, June 7-10, 2011. Proceedings*, volume 6715 of *Lecture Notes in Computer Science*, pages 327–344, 2011.

[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. IEEE Computer Society, 2012.

[ZWW+14] Lei Zhang, Wenling Wu, Yanfeng Wang, Shengbao Wu, and Jian Zhang. LAC: A Lightweight Authenticated Encryption Cipher. Submission to CAESAR. Available from: http://competitions.cr.yp.to/round1/lacv1.pdf (v1), March 2014.