Quantum Attacks Without Superposition Queries: The Offline Simon's Algorithm

Xavier Bonnetain[1,3(✉)], Akinori Hosoyamada[2,4], María Naya-Plasencia[1], Yu Sasaki[2], and André Schrottenloher[1]

[1] Inria, Paris, France
{xavier.bonnetain,maria.naya_plasencia,andre.schrottenloher}@inria.fr
[2] NTT Secure Platform Laboratories, Tokyo, Japan
{hosoyamada.akinori,sasaki.yu}@lab.ntt.co.jp
[3] Collège Doctoral, Sorbonne Université, 75005 Paris, France
[4] Nagoya University, Nagoya, Japan

Abstract. In symmetric cryptanalysis, the model of superposition queries has led to surprising results, with many constructions being broken in polynomial time thanks to Simon's period-finding algorithm. But the practical implications of these attacks remain blurry. In contrast, the results obtained so far for a quantum adversary making classical queries only are less impressive.

In this paper, we introduce a new quantum algorithm which uses Simon's subroutines in a novel way. We manage to leverage the algebraic structure of cryptosystems in the context of a quantum attacker limited to classical queries and offline quantum computations. We obtain improved quantum-time/classical-data tradeoffs with respect to the current literature, while using only as much hardware requirements (quantum and classical) as a standard exhaustive search with Grover's algorithm. In particular, we are able to break the Even-Mansour construction in quantum time $\tilde{O}(2^{n/3})$, with $O(2^{n/3})$ classical queries and $O(n^2)$ qubits only. In addition, we improve some previous superposition attacks by reducing the data complexity from exponential to polynomial, with the same time complexity.

Our approach can be seen in two complementary ways: *reusing* superposition queries during the iteration of a search using Grover's algorithm, or alternatively, removing the memory requirement in some quantum attacks based on a collision search, thanks to their algebraic structure.

We provide a list of cryptographic applications, including the Even-Mansour construction, the FX construction, some Sponge authenticated modes of encryption, and many more.

Keywords: Simon's algorithm · Classical queries · Symmetric cryptography · Quantum cryptanalysis · Even-Mansour construction · FX construction

S. D. Galbraith and S. Moriai (Eds.): ASIACRYPT 2019, LNCS 11921, pp. 552–583, 2019.
https://doi.org/10.1007/978-3-030-34578-5_20

1 Introduction

Ever since Shor [39] introduced his celebrated quantum polynomial-time algorithm for solving factorization and Discrete Logarithms, both problems believed to be classically intractable, post-quantum cryptography has become a subject of wide interest. Indeed, the security of classical cryptosystems relies on computational assumptions, which until recently, were made with respect to classical adversaries; if quantum adversaries are to be taken into account, the landscape of security is bound to change dramatically.

While it is difficult to assert the precise power of quantum computers, which are yet to come, it is still possible to study quantum algorithms for cryptographic problems, and to estimate the computational cost of solving these problems for a quantum adversary. The ongoing project by NIST [35] for post-quantum *asymmetric* schemes aims to replace the current mostly used ones by new standards.

In *symmetric* cryptography, the impact of quantum computing seems, at first sight, much more limited. This is because the security of most of symmetric-key schemes is not predicated on structured problems. Symmetric-key schemes are required to be computed extremely efficiently, and designers must avoid such computationally expensive operations. Grover's quantum search algorithm [22], another cornerstone of quantum computing, speeds up by a quadratic factor exhaustive search procedures. This has led to the common saying that "doubling the key sizes" should ensure a similar level of post-quantum security.

However, the actual post-quantum security of symmetric-key schemes requires more delicate treatment. Recovering the secret key via exhaustive search is only one of all the possible approaches. The report of the National Academy of Sciences on the advent of quantum computing [34] also states that "it is possible that there is some currently unknown clever quantum attack" that would perform much better than Grover's algorithm. Indeed, cryptographers are making significant progress on quantum attackers with *superposition queries*, which break many symmetric-key schemes in polynomial time.

Quantum Generic Attacks in Q1 and Q2 Models. Quantum attacks can be mainly classified into two types [20,23,25], Q1 model and Q2 model, assuming different abilities for the attacker. In the Q1 model, attackers have an access to a quantum computer to perform any offline computation, while they are only allowed to make online queries in a classical manner. In the Q2 model, besides the offline quantum computation, attackers are allowed to make superposition queries to a quantum cryptographic oracle. Here, we briefly review previous results in these models to introduce the context of our results.

The Q2 model is particularly interesting as it yields some attacks with a very low cost. Kuwakado and Morii [29,30] showed that the Even-Mansour cipher and the three-round Feistel networks, classically proven secure if their underlying building blocks are ideal, were broken in polynomial time. This exponential speedup, the first concerning symmetric cryptography, was obtained thanks to Simon's algorithm [40] for recovering a Boolean hidden shift. Later on, more results have been obtained in this setting, with more generic constructions

broken [24,31], and an exponential acceleration of *slide attacks*, which target ciphers with a self-similar structure. Versions of these attacks [6] for constructions with modular additions use Kuperberg's algorithm [27], allowing a better than quadratic speed-up. All these attacks, however, run in the model of *superposition queries*, which models a quantum adversary having some inherently quantum access to the primitives attacked. As such, they do not give any improvement when the adversary only has classical access.

Stated differently, the attacks in the Q1 model are particularly relevant due to their impact on current data communication technology. However, the quantum algorithms that have been exploited for building attacks in the Q1 model are very limited and have not allowed more than a quadratic speed-up. The most used algorithm is the simple quantum exhaustive search with Grover's algorithm. A possible direction is the collision finding algorithm that is often said to achieve "$2^{n/3}$ complexity" versus $2^{n/2}$ classically. However, even in this direction, there are several debatable points; basic quantum algorithms for finding collisions have massive quantum hardware requirements [9]. There is a quantum-hardware-friendly variant [12], but then the time complexity becomes suboptimal.

In summary, attacks using Simon's algorithm could achieve a very low complexity but could only be applied in the Q2 model, a very strong model. In contrast, attacks in the Q1 model are practically more relevant, but for now the obtained speed-ups were not surprising.

Another model to consider when designing quantum attacks is whether the attacker has or not a big amount of quantum memory available. Small quantum computers seem like the most plausible scenario, and therefore attacks needing a polynomial amount of qubits are more practically relevant. Therefore, the most *realistic* scenario is Q1 with small quantum memory.

Our Main Contribution. The breakthrough we present in this paper is the first application of Simon's algorithm [40] in the Q1 model, which requires significantly less than $\mathcal{O}\left(2^{n/2}\right)$ classical queries and offline quantum computations, only with $\mathsf{poly}(n)$ qubits, and no qRAM access (where n is the size of the secret). Namely, we remove the superposition queries in previous attacks. The new idea can be applied to a long list of ciphers and modes of operation. Let us illustrate the impact of our attacks by focusing on two applications:

The first application is the key recovery on the Even-Mansour construction, which is one of the simplest attacks using Simon's algorithm. Besides the polynomial time attacks in the Q2 model, Kuwakado and Morii also developed an attack in the Q1 model with $\mathcal{O}\left(2^{n/3}\right)$ classical queries, quantum computations, qubits, and classical memory [30]. The extension of this Q1 attack by Hosoyamada and Sasaki [23] recovers the key with $\mathcal{O}\left(2^{3n/7}\right)$ classical queries, $\mathcal{O}\left(2^{3n/7}\right)$ quantum computations, polynomially many qubits and $\mathcal{O}\left(2^{n/7}\right)$ classical memory (to balance classical queries and quantum computations). Our attack in the Q1 model only uses polynomially many qubits, yet only requires $\mathcal{O}\left(2^{n/3}\right)$ classical queries, $\mathcal{O}\left(n^3 2^{n/3}\right)$ quantum computations and $\mathsf{poly}(n)$ classical memory.

The second application is the key recovery on the FX-construction $\mathrm{FX}_{k,k_{in},k_{out}}$, which computes a ciphertext c from a plaintext p by $c \leftarrow E_k(p \oplus k_{in}) \oplus k_{out}$, where E is a block cipher, k is an m-bit key and k_{in}, k_{out} are two n-bit keys. Leander and May proposed an attack in the Q2 model with $\mathcal{O}\left(n2^{m/2}\right)$ superposition queries, $\mathcal{O}\left(n^3 2^{m/2}\right)$ quantum computations, $\mathsf{poly}(n)$ qubits and $\mathsf{poly}(n)$ classical memory [31].[1] They combined Simon's algorithm and Grover's algorithm in a clever way, while it became inevitable to make queries in an adaptive manner. For the Q1 model, the meet-in-the-middle attack [23] can recover the key with $\mathcal{O}\left(2^{3(m+n)/7}\right)$ complexities. Our results can improve the previous attacks in two directions. One is to reduce the amount of superposition queries in the Q2 model to the polynomial order and convert the adaptive attack to a non-adaptive one. The other is to completely remove the superposition queries. The comparison of previous quantum attacks and our attacks on Even-Mansour and the FX construction is shown in Table 1. Other interesting complexity trade-offs are possible, as shown in detail in Sects. 4 and 5.

Table 1. Previous and new quantum attacks on Even-Mansour and FX, assuming that $m = \mathcal{O}(n)$.

Target	Model	Queries	Time	Q-memory	C-memory	Reference
EM	Q2	$\mathcal{O}(n)$	$\mathcal{O}\left(n^3\right)$	$\mathcal{O}(n)$	$\mathcal{O}\left(n^2\right)$	[30]
	Q1	$\mathcal{O}\left(2^{n/3}\right)$	$\mathcal{O}\left(2^{n/3}\right)$	$\mathcal{O}\left(2^{n/3}\right)$	$\mathcal{O}\left(2^{n/3}\right)$	[30]
	Q1	$\mathcal{O}\left(2^{3n/7}\right)$	$\mathcal{O}\left(2^{3n/7}\right)$	$\mathcal{O}(n)$	$\mathcal{O}\left(2^{n/7}\right)$	[23]
	Q1	$\mathcal{O}\left(2^{n/3}\right)$	$\mathcal{O}\left(n^3 2^{n/3}\right)$	$\mathcal{O}\left(n^2\right)$	$\mathcal{O}(n)$	Section 5
FX	Q2	$\mathcal{O}\left(n2^{m/2}\right)$	$\mathcal{O}\left(n^3 2^{m/2}\right)$	$\mathcal{O}\left(n^2\right)$	0	[31]
	Q2	$\mathcal{O}(n)$	$\mathcal{O}\left(n^3 2^{m/2}\right)$	$\mathcal{O}\left(n^2\right)$	$\mathcal{O}(n)$	Section 4
	Q1	$\mathcal{O}\left(2^{3(m+n)/7}\right)$	$\mathcal{O}\left(2^{3(m+n)/7}\right)$	$\mathcal{O}(n)$	$\mathcal{O}\left(2^{(m+n)/7}\right)$	[23]
	Q1	$\mathcal{O}\left(2^{(m+n)/3}\right)$	$\mathcal{O}\left(n^3 2^{(m+n)/3}\right)$	$\mathcal{O}\left(n^2\right)$	$\mathcal{O}(n)$	Section 5

Our New Observation. Here we describe our new algorithm used in the Q1 model with the Even-Mansour construction as an example. Recall that the encryption E_{k_1,k_2} of the Even-Mansour construction is defined as $E_{k_1,k_2}(x) = P(x \oplus k_1) \oplus k_2$, where P is a public permutation and $k_1, k_2 \in \{0,1\}^n$ are the secret keys. Roughly speaking, our attack guesses $(2n/3)$-bit of k_1 (denoted by $k_1^{(2)}$ in Fig. 1) by using the Grover search, and checks if the guess is correct by applying Simon's algorithm to the remaining $(n/3)$-bit of k_1 (denoted by $k_1^{(1)}$ in Fig. 1). If we were in the Q2 model, we could recover k_1 using the technique by Leander and May [31] in time $\tilde{\mathcal{O}}(2^{n/3})$. However, their technique is not applicable in the Q1 setting since quantum queries are required.

Our core observation that realizes the above idea in the Q1 model is that, we can judge whether a function $f \oplus g$ has a period (i.e., we can apply Simon's algorithm) without any quantum query to g, if we have the quantum state

[1] Here we are assuming that m is in $\mathcal{O}(n)$, which is the case for usual block ciphers.

$|\psi_g\rangle := (\sum_x |x\rangle|g(x)\rangle)^{\otimes cn}$ (c is a small constant): If we have the quantum state $|\psi_g\rangle$, then we can make the quantum state $|\psi_{f\oplus g}\rangle := (\sum_x |x\rangle|(f \oplus g)(x)\rangle)^{\otimes cn}$ by making $\mathcal{O}(n)$ quantum queries to f. Once we obtain $|\psi_{f\oplus g}\rangle$, by applying the Hadamard operation $H^{\otimes n}$ to each $|x\rangle$ register, we obtain the quantum state

$$\left(\sum_{x_1,u_1} (-1)^{u_1\cdot x_1}|u_1\rangle|(f\oplus g)(x_1)\rangle\right)\otimes\cdots\otimes\left(\sum_{x_{cn},u_{cn}} (-1)^{u_{cn}\cdot x_{cn}}|u_{cn}\rangle|(f\oplus g)(x_{cn})\rangle\right)$$

Then, roughly speaking, $\dim(\mathrm{Span}(u_1,\ldots,u_{cn})) < n$ always holds if $f \oplus g$ has a secret period s, while $\dim(\mathrm{Span}(u_1,\ldots,u_{cn})) = n$ holds with a high probability if $f \oplus g$ does not have any period. Since the dimension of the vector space can be computed in time $\mathcal{O}\left(n^3\right)$, we can judge if $f \oplus g$ has a period in time $\mathcal{O}\left(n^3\right)$. Note that we can reconstruct the quantum data $|\psi_g\rangle$ after judging whether $(f \oplus g)$ has a period (with some errors) by appropriately performing uncomputations, which help us use these procedures as a subroutine without measurement in other quantum algorithms.

For the Even-Mansour construction, we set $g : \{0,1\}^{n/3} \to \{0,1\}^n$ by $g(x) := E_{k_1,k_2}(x\|0^{2n/3})$. Then we can make the quantum state $|\psi_g\rangle$ by classically querying x to g for *all* $x \in \{0,1\}^{n/3}$, which requires $2^{n/3}$ classical queries. After obtaining the state $|\psi_g\rangle$, we guess $k_1^{(2)}$. Suppose that here our guess is $k' \in \{0,1\}^{2n/3}$. We define $f_{k'} : \{0,1\}^{n/3} \to \{0,1\}^n$ by $f_{k'}(x) := P(x\|k')$. Then, roughly speaking, our guess is correct if and only if the function $f_{k'} \oplus g$ has a period $k_1^{(1)}$. Thus we can judge whether the guess is correct without quantum queries to g, by using our technique described above. Since $k_1^{(2)}$ can be guessed in time $\tilde{\mathcal{O}}(2^{n/3})$ by using the Grover search, we can recover the keys by making $\mathcal{O}(2^{n/3})$ classical queries and $\tilde{\mathcal{O}}(2^{n/3})$ offline quantum computations.

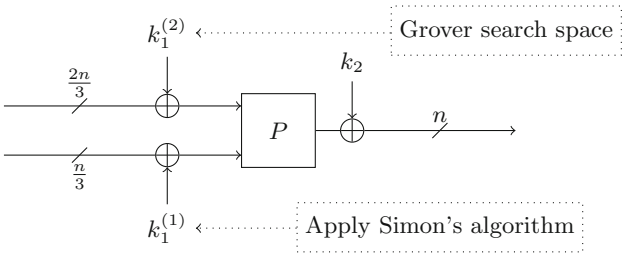


Fig. 1. Idea of our Q1 attack on the Even-Mansour construction.

We will show how we can similarly attack the FX construction in the Q1 model, by guessing additional key bits (see Fig. 2).

Moreover, our attack idea in the Q1 model can also be used to reduce the number of quantum queries of attacks in the Q2 model. The Leander and May's attack on the FX construction in the Q2 model [31] guesses the m-bit key k of the FX construction $\mathrm{FX}_{k,k_{in},k_{out}}$ and checks whether the guess is correct by

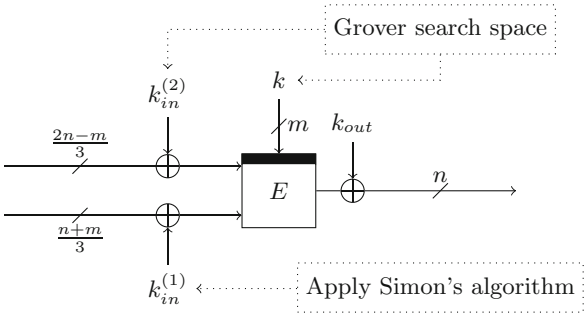


Fig. 2. Idea of our Q1 attack on the FX construction.

using Simon's algorithm, which requires $\mathcal{O}(2^{m/2})$ online quantum queries and $\tilde{\mathcal{O}}(2^{m/2})$ offline quantum computations. Roughly speaking, the guess k' for the key k is correct if and only if $(f_{k'} \oplus g)(x)$ has the secret period k_{in}, where $f_{k'}(x) = E_{k'}(x)$ and $g(x) = \mathrm{FX}_{k,k_{in},k_{out}}(x)$. In the Q2 model, we can make the quantum state $|\psi_g\rangle = (\sum_x |x\rangle|g(x)\rangle)^{\otimes cn}$ by making $\mathcal{O}(n)$ quantum queries to g. Thus, by our new attack idea described above, we can break the FX construction with $\mathcal{O}(n)$ online quantum queries and $\tilde{\mathcal{O}}(2^{m/2})$ offline quantum computations, which exponentially improves the attack by Leander and May from the viewpoint of *quantum query* complexity.

This exponential improvement on the quantum query complexity is due to the separation of offline queries and online computations: In the previous attack on the FX construction in the Q2 model by Leander and May, we have to do online queries and offline computations alternately in each iteration of the Grover search. Thus the number of online quantum queries becomes exponential in the previous attack. On the other hand, in our new attack, the online queries (i.e., the procedures to make the quantum state $|\psi_g\rangle$) are completely separated from offline computations. This enables us to decrease the number of quantum queries exponentially, while we still need exponentially many offline computations.

Paper Organization. Section 2 gives preliminaries. Section 3 describes our main algorithms. Section 4 shows applications of our algorithms in the Q2 model. Section 5 shows applications of our algorithms in the Q1 model. Section 6 discusses further applications of our algorithm. Section 7 concludes the paper.

2 Preliminaries

In this section, we introduce some quantum computing notions and review Simon's and Grover's algorithms. We refer to [36] for a broader presentation.

2.1 The Quantum Circuit Model

It has become standard in the cryptographic literature to write quantum algorithms in the circuit model, which is universal for quantum computing. We only consider the logical level of quantum circuits, with logical qubits, not their implementation level (which requires physical qubits, quantum error-correction, *etc*). Although it is difficult to estimate the cost of a physical implementation which does not yet exist, we can compare security levels as quantum operation counts in this model. For example, Grover search of the secret key for AES-128 is known to require approximately 2^{64} quantum evaluations of the cipher, and 2^{84} quantum operations [21].

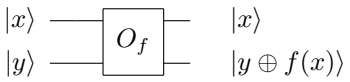
Qubits and Operations. A quantum circuit represents a sequence of quantum operations, denoted as *quantum gates*, applied to a set of *qubits*. An individual qubit is a quantum object whose state is an element of a two-dimensional Hilbert space, with basis $|0\rangle, |1\rangle$ (analogs of the classical logical 0 and 1). Hence, the state is described as a linear combination of $|0\rangle, |1\rangle$ with complex coefficients (a *superposition*). We add to this a normalization condition: $\alpha|0\rangle + \beta|1\rangle$ is such that $|\alpha|^2 + |\beta|^2 = 1$. When it is clear from context, we dismiss common normalization factors.

When n qubits are given, the computational basis has 2^n vectors, which are all n-bit strings. The qubits start in a state $|0\rangle$, for example a fixed spin or polarization. The sequence of quantum gates that is applied modifies the superposition, thanks to constructive and destructive interferences. In the end, we measure the system, and obtain some n-bit vector in the computational basis, which we expect to hold a meaningful result.

All computations are (linear) unitary operators of the Hilbert space, and as such, are reversible (this holds for the individual gates, but also for the whole circuit). In general, any classical computation can be made reversible (and so, implemented as a quantum circuit) provided that one uses sufficiently many *ancilla qubits* (which start in the state $|0\rangle$ and are brought back to $|0\rangle$ after the computation). Generally, on input $|x\rangle$, we can perform some computation, copy the result to an output register using CNOT gates, and uncompute (perform backwards the same operations) to restore the initial state of the ancilla qubits. Uncomputing a unitary U corresponds to applying its adjoint operator U^*.

By the principle of *deferred measurements*, any measure that occurs inside the quantum circuit can be deferred to the end of the computation.

Quantum Oracles. Many quantum algorithms require an oracle access. The difference they make with classical algorithms with this respect is that classical oracles (*e.g.* cryptographic oracles such as a cipher with unknown key) are queried "classically", with a single value, while quantum oracles are unitary operators. We consider oracle calls of the type:



which XOR their output value to an output register (ensuring reversibility). If we consider that $|y\rangle$ starts in the state $|0\rangle$, then $f(x)$ is simply written here. If the function f can be accessed through O_f, we say it has superposition oracle access.

Quantum RAM. Additionally to the use of "plain" quantum circuits with universal quantum computation, many algorithms require quantum random-access, or being able to access at runtime a *superposition* of memory cells. This is a strong requirement, since this requires an extensive quantum hardware (the qRAM) and a huge architecture that is harder to build than a quantum circuit with a limited number of qubits. Shor's algorithm, Simon's algorithm, Grover's algorithm do not require qRAM, if their oracle calls do not either, contrary to, *e.g.*, the algorithm for quantum collision search of [9], whose optimal speedup can be realized only by using massive qRAM.

Our algorithm has no such requirement, which puts it on the same level of practicality as Grover's algorithm for attacking symmetric primitives.

2.2 Simon's Algorithm

Simon's algorithm [40] gives an exponential speedup on the following problem.

Problem 1 (Simon's problem). Suppose given access to a function $f : \{0,1\}^n \to \{0,1\}^n$ that is either injective, or such that there exists $s \in \{0,1\}^n$ with:

$$\forall x, f(x) = f(y) \iff y = x \text{ or } y = x \oplus s,$$

then find α.

In other words, the function f has a hidden Boolean period. It is also easy to extend this algorithm to a hidden Boolean shift, when we want to decide whether two functions f and g are such that $g(x) = f(x \oplus s)$ for all x. In practice, f can fall in any set X provided that it can be represented efficiently, but in our examples, we will consider functions producing bit strings.

Solving this problem with classical oracle access to f requires $\Omega\left(2^{n/2}\right)$ queries, as we need to find a collision of f (or none, if there is no hidden period). Simon [40] gives an algorithm which only requires $\mathcal{O}(n)$ superposition queries. We fix $c \geq 1$ a small constant to ensure a good success probability and repeat cn times Algorithm 1.

We obtain either:

- a list of cn random values of y;
- a list of cn random values of y in the hyperplane $y \cdot s = 0$.

It becomes now easy to test whether s exists or not. If it doesn't, the system of equations obtained has full rank. If it does exist, we can find it by solving the system. Judging whether there exists such an s and actually finding it (if it exists) can be done in time $\mathcal{O}\left(n^3\right)$ by Gaussian elimination.

Algorithm 1. Quantum subroutine of Simon's algorithm.

1: Start in the all-zero state $|0\rangle|0\rangle$ where the first register contains n qubits and the second represents elements of X.
2: Apply Hadamard gates to obtain:

$$\sum_{x\in\{0,1\}^n} |x\rangle|0\rangle$$

3: Query O_f to obtain:

$$\sum_{x\in\{0,1\}^n} |x\rangle|f(x)\rangle = \sum_{a\in X} \left(\sum_{x\in\{0,1\}^n | f(x)=a} |x\rangle \right) |a\rangle$$

4: Measure a (alternatively, we can defer this measurement), get a random value $a \in X$ and:

$$\sum_{x\in\{0,1\}^n | f(x)=a} |x\rangle$$

5: Apply Hadamard gates:

$$\sum_{y\in\{0,1\}^n} \left(\sum_{x\in\{0,1\}^n | f(x)=a} (-1)^{x\cdot y} \right) |y\rangle$$

6: Now measure the y register. There are two cases.
- Either f hides no period s, in which case we get a random y.
- Either f hides a period s, in which case the amplitude of $|y\rangle$ is:

$$\sum_{x\in\{0,1\}^n | f(x)=a} (-1)^{x\cdot y} = (-1)^{x_0\cdot y} + (-1)^{(x_0\oplus s)\cdot y}$$

which is zero if $y \cdot s = 1$ and non-zero otherwise.
- In that case, measuring gives a random y such that $y \cdot s = 0$.

Simon's Algorithm in Cryptography. This algorithm has been used in many attacks on modes of operation and constructions where recovering a secret requires to find a hidden shift between two functions having bit-string inputs. Generally, the functions to which Simon's algorithm is applied are not injective, and random collisions can occur. But a quick analysis (as done *e.g.* in [24]) shows that even in this case, a mild increase of the constant c will increase the success probability to a sufficient level. To be precise, the following proposition holds.

Proposition 1 (Theorem 2 in [24]). *Suppose that $f : \{0,1\}^n \to X$ has a period $s \neq 0^n$, i.e., $f(x \oplus s) = f(x)$ for all $x \in \{0,1\}^n$, and satisfies*

$$\max_{t\neq\{s,0^n\}} \Pr_x [f(x \oplus t) = f(x)] \leq \frac{1}{2}. \tag{1}$$

When we apply Simon's algorithm to f, it returns s with a probability at least $1 - 2^n \cdot (3/4)^{cn}$.

2.3 Grover's Algorithm

Grover's algorithm [22] allows a quadratic speedup on classical exhaustive search. Precisely, it solves the following problem:

Problem 2 (Grover's problem). Consider a set X (the "search space") whose elements are represented on $\lceil \log_2(|X|) \rceil$ qubits, such that the uniform superposition $\sum_{x \in X} |x\rangle$ is computable in $\mathcal{O}(1)$ time. Given oracle access to a function $f : X \to \{0,1\}$ (the "test"), find $x \in X$ such that $f(x) = 1$.

Classically, if there are 2^t preimages of 1, we expect one to be found in time (and oracle accesses to f) $\mathcal{O}(|X|/2^t)$. Quantumly, Grover's algorithm finds one in time (and oracle accesses to O_f) $\widetilde{\mathcal{O}}\left(\sqrt{|X|/2^t}\right)$. In particular, if there is one preimage of 1, the running time is $\widetilde{\mathcal{O}}\left(\sqrt{|X|}\right)$. If the superposition oracle for f uses a ancilla qubits, then Grover's algorithm requires $a + \lceil \log_2(|X|) \rceil$ qubits only.

Grover's algorithm works first by producing the superposition $\sum_{x \in |X|} |x\rangle$. It applies $\widetilde{\mathcal{O}}\left(\sqrt{|X|/2^t}\right)$ times an operator which, by querying O_f "moves" some amplitude towards the preimages of 1.

3 Simon's Algorithm with Asymmetric Queries

In this section, we introduce a problem that can be seen as a general combination of Simon's and Grover's problems, and that will be solved by an according combination of algorithmic ideas. The problem has many cryptographic applications, and it will be at the core of our improved Q2 and Q1 time-memory-data tradeoffs.

Problem 3 (Asymmetric Search of a Period). Let $F : \{0,1\}^m \times \{0,1\}^n \to \{0,1\}^\ell$ and $g : \{0,1\}^n \to \{0,1\}^\ell$ be two functions. We consider F as a family of functions indexed by $\{0,1\}^m$ and write $F(i,\cdot) = f_i(\cdot)$. Assume that we are given quantum oracle access to F, and classical or quantum oracle access to g. (In the Q1 setting, g will be a classical oracle. In the Q2 setting, g will be a quantum oracle.)

Assume that there exists exactly one $i \in \{0,1\}^m$ such that $f_i \oplus g$ has a hidden period, *i.e.*: $\forall x \in \{0,1\}^n, f_{i_0}(x) \oplus g(x) = f_{i_0}(x \oplus s) \oplus g(x \oplus s)$ for some s. Furthermore, assume that:

$$\max_{\substack{i \in \{0,1\}^m \setminus \{i_0\} \\ t \in \{0,1\}^n \setminus \{0^n\}}} \Pr_{x \leftarrow \{0,1\}^n} [(f_i \oplus g)(x \oplus t) = (f_i \oplus g)(x)] \leq \frac{1}{2} \tag{2}$$

Then find i_0 and s.

In our cryptographic applications, g will be a keyed function such that adversaries have to make online queries to evaluate it, while F will be a function such that adversaries can evaluate it offline. For example, the problem of recovering

keys of the FX construction $\mathrm{FX}_{k,k_{in},k_{out}}(x) = E_k(x \oplus k_{in}) \oplus k_{out}$ can be regarded as a simple cryptographic instantiation of Problem 3: Set $g(x) := \mathrm{FX}_{k,k_{in},k_{out}}(x)$ and $F(i,x) := E_i(x)$. Then, roughly speaking, the function $f_i \oplus g$ has a period k_{in} if $k = i$, whereas it does not have any period if $i \neq k$ and Condition (2) holds. Thus we can know whether $i = k$ by checking whether $f_i \oplus g$ has a period.

Justification of Condition (2). We added Condition (2) in Problem 3 because the problem would be much harder to solve if we do not suppose any condition on f_i. Such assumptions are standard in the litterature of quantum attacks using Simon's algorithm (see for example [24, Sections 2.2 and 4] or [4, Section 3]). This is reasonable for cryptographic applications, as a block cipher is expected to behave like a random permutation, which makes the functions we construct in our applications behave like random functions. This assumption is made in [24, 31], and such functions satisfy Condition (2) with an overwhelming probability. Moreover, as remarked in [24], a cryptographic construction that fails to satisfy Condition (2) would exhibit some poor differential properties which could be used for cryptanalysis.

3.1 Existing Techniques to Solve the Problem

Here we explain existing algorithms to solve Problem 3 in both the Q1 model and the Q2 model, with the algorithms to recover keys of the FX construction as an example. Note that we consider the situation in which exponentially many qubits are *not* available.

The Model Q1. In the Q1 model, when we are allowed to make only classical queries to $g := \mathrm{FX}_{k,k_{in},k_{out}}$, there exists a Q1 algorithm to attack the FX construction that uses a kind of meet-in-the-middle technique [23]. However, it does not make use of the exponential speed-up of Simon's algorithm, and its time complexity and query complexity is $\mathcal{O}\left(2^{3(n+m)/7}\right)$ (for $m \leq 4n/3$).

The Model Q2. Problem 3 can be solved with $\mathcal{O}\left(n2^{m/2}\right)$ superposition queries to $F(i,x) = E_i(x)$ and $g(x) = \mathrm{FX}_{k,k_{in},k_{out}}(x)$, and in time $\mathcal{O}\left(n^3 2^{m/2}\right)$, using the Grover-meet-Simon algorithm of [31]. Indeed, we make a Grover search on index $i \in \{0,1\}^m$. When testing whether a guess i for the key k is correct, we perform $\mathcal{O}(n)$ queries to F and $\mathcal{O}(n)$ queries to g, to check whether $f_i \oplus g$ has a hidden period, hence whether the guess i is correct. Moreover, since superposition access to F and g is allowed, we can test i in superposition as well.

3.2 An Algorithm for Asymmetric Search of a Shift

Here we describe our new algorithms to solve Problem 3. We begin with explaining two observations on the Grover-meets-Simon algorithm in the Q2 model described in Sect. 3.1, and how to improve it. Then we describe how to use the idea to make a good algorithm to solve Problem 3 in the Q1 model.

Two Observations. Our first observation is that, when doing the Grover search over i for Problem 3, each time a new i is tested, a new function f_i is queried. But, in contrast, the function g is always the same. We would like to take this asymmetry into account, namely, to make less queries to g since it does not change. This in turn has many advantages: queries to g can become more costly than queries to f_i.

Our second observation is that, for each $i \in I$, once we have a superposition $|\psi_g\rangle = \bigotimes^{cn} \left(\sum_{x \in \{0,1\}^n} |x\rangle |g(x)\rangle \right)$ and given a quantum oracle access to f_i, we can obtain the information if $f_i \oplus g$ has a period or not without making queries to g.

From $|\psi_g\rangle$, we can make the state $|\psi_{f_i \oplus g}\rangle = \bigotimes^{cn} \left(\sum_{x \in \{0,1\}^n} |x\rangle |f_i(x) \oplus g(x)\rangle \right)$ by making queries to f_i. By applying usual Simon's procedures on $|\psi_{f_i \oplus g}\rangle$, we can judge if $f_i \oplus g$ has a period. Moreover, by appropriately performing uncomputations, we can recover $|\psi_g\rangle$ (with some errors) and reuse it in other procedures.

With these observations in mind, below we give an intuitive description of our algorithm Alg-PolyQ2 to solve Problem 3 in the Q2 model (we name our algorithm Alg-PolyQ2 because it will be applied to make Q2 attacks with *polynomially* many online queries in later sections). The main ideas of Alg-PolyQ2 are separating an online phase and offline computations, and iteratively reusing the quantum data $|\psi_g\rangle$ obtained by the online phase.

Algorithm Alg-PolyQ2**(informal)**

1. Online phase: Make cn quantum queries to g to prepare $|\psi_g\rangle$.
2. Offline computations: Run the Grover search over $i \in \{0,1\}^m$. For each fixed i, run a testing procedure test such that: (a) test checks if i is a good element (i.e., $f_i \oplus g$ has a period) by using $|\psi_g\rangle$ and making queries to f_i, and (b) after checking if i is good, appropriately performs uncomputations to recover the quantum data $|\psi_g\rangle$.

A formal description of Alg-PolyQ2 is given in Algorithm 2. We fix a constant $c \geq 1$, to be set later depending on the probability of error wanted.

We show how to implement the testing procedure test in Algorithm 3 without any new query to g, using only exactly $2cn$ superposition queries to F. To write this procedure clearly, we consider a single function f in input, but remark that it works as well if f is a superposition of f_i (as will be the case when test is called as the oracle of a Grover search).

In practice, Algorithm 3 works up to some error (see Remark 1), which is amplified at each iteration of Algorithm 2. The complexity and success probability (including the errors) of Alg-PolyQ2 can be analyzed as below.

Proposition 2. *Suppose that m is in $\mathcal{O}(n)$. Let c be a sufficiently large constant.*[2] *Consider the setting of Problem 3: let $i_0 \in \{0,1\}^m$ be the good element such that $g \oplus f_{i_0}$ is periodic and assume that (2) holds. Then* Alg-PolyQ2

[2] See Proposition 5 for a concrete estimate.

Algorithm 2. Alg-PolyQ2.

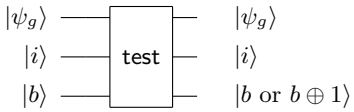
1: Start in the all-zero state.
2: Using cn queries to g, create the state:

$$|\psi_g\rangle = \bigotimes^{cn} \left(\sum_{x \in \{0,1\}^n} |x\rangle|g(x)\rangle \right)$$

The circuit now contains $|\psi_g\rangle$, the "g-database", and additional registers on which we can perform Grover search. Notice that $|\psi_g\rangle$ contains cn independent (and disentangled) registers.
3: Create the uniform superposition over indices $i \in \{0,1\}^m$:

$$|\psi_g\rangle \otimes \sum_{i \in \{0,1\}^m} |i\rangle$$

4: Apply Grover iterations. The testing oracle is a unitary operator test that takes in input a register for $|i\rangle$ and the "g-database", and tests in superposition whether $f_i \oplus g$ has a hidden period. If this is the case, it returns $|b \oplus 1\rangle$ on input $|b\rangle$. Otherwise it returns $|b\rangle$. (Algorithm 3 gives the details for test in the case that i is fixed.)



The most important feature of test is that it does not change the g-database (up to some errors). The registers holding $|\psi_g\rangle$ are disentangled before and after the application of test.
5: After $\mathcal{O}\left(2^{m/2}\right)$ Grover iterations, measure the index i.
6: If the hidden shift is also wanted, apply a single instance of Simon's algorithm (or re-use the database and perform a slightly extended computation of test to retrieve the result).

finds i_0 with a probability in $\Theta(1)$ by making $\mathcal{O}(n)$ quantum queries to g and $\mathcal{O}\left(n2^{m/2}\right)$ quantum queries to F.[3] *The offline computation (the procedures excluding the ones to prepare the state $|\psi_g\rangle$) of* Alg-PolyQ2 *is done in time $\mathcal{O}\left((n^3 + nT_F)2^{m/2}\right)$, where T_F is the time required to evaluate F once.*

See Section A in the full version of the paper [5] for a proof.

Remark 1. Intuitively, the error produced in each iteration of Algorithm 3 is bounded by the maximum, on i, of: $p^{(i)} := \Pr\left[\dim(\mathrm{Span}(u_1, \ldots, u_{cn})) < n\right]$, when $u_1, \ldots, u_{cn}$ are produced with Simon's algorithm, *i.e.* the probability that

[3] In later applications, F will be instantiated with unkeyed primitives, and quantum queries to F are emulated with offline computations of primitives such as block ciphers.

Algorithm 3. The procedure test that checks if a function $f \oplus g$ has a period against the g-database, without any new query to g.

1: We start with the g-database:

$$|\psi_g\rangle = \bigotimes^{cn} \left(\sum_{x \in \{0,1\}^n} |x\rangle |g(x)\rangle \right)$$

2: Using cn superposition queries to f, build the state:

$$|\psi_{f \oplus g}\rangle = \bigotimes^{cn} \left(\sum_{x \in \{0,1\}^n} |x\rangle |g(x) \oplus f(x)\rangle \right)$$

We will now perform, in a reversible way, the exact computations of Simon's algorithm to find if $g \oplus f$ has a hidden period or not (in which case f and g have a hidden shift).

3: Apply $\left(H^{\otimes n} \otimes I_m\right)^{cn} \otimes I_1$ to $|\psi_{f \oplus g}\rangle \otimes |b\rangle$, to obtain

$$\left(\sum_{u_1, x_1 \in \{0,1\}^n} (-1)^{u_1 \cdot x_1} |u_1\rangle |(f \oplus g)(x_1)\rangle \right) \otimes \cdots \tag{3}$$
$$\cdots \otimes \left(\sum_{u_{cn}, x_{cn} \in \{0,1\}^n} (-1)^{u_{cn} \cdot x_{cn}} |u_{cn}\rangle |(f \oplus g)(x_{cn})\rangle \right) \otimes |b\rangle.$$

4: Compute $d := \dim(\mathrm{Span}(u_1, \ldots, u_{cn}))$, set $r := 0$ if $d = n$ and $r := 1$ if $d < n$, and add r to b. Then uncompute d and r, and obtain

$$\sum_{\substack{u_1, \ldots, u_{cn} \\ x_1, \ldots, x_{cn}}} (-1)^{u_1 \cdot x_1} |u_1\rangle |(f \oplus g)(x_1)\rangle \otimes \cdots \tag{4}$$
$$\cdots \otimes (-1)^{u_{cn} \cdot x_{cn}} |u_{cn}\rangle |(f \oplus g)(x_{cn})\rangle \otimes |b \oplus r\rangle.$$

Note that r in (4) depends on $u_1, \ldots, u_{cn}$ and now the last register may be entangled with the registers of $u_1, \ldots, u_{cn}$.

5: Uncompute $\left(H^{\otimes n} \otimes I_m\right)^{cn} \otimes I_1$.

6: Using cn new superposition queries to f, revert $|\psi_{f \oplus g}\rangle$ to $|\psi_g\rangle$.

There are two cases:

- If $f \oplus g$ has a hidden period, then $r = 1$ always holds. Hence, in the output register, we always write 1.
- If $f \oplus g$ does not have a hidden period, then with high probability, $r = 0$. Hence, in the output register, we write 0.

Simon's algorithm returns the incorrect answer "$f_i \oplus g$ is periodic" even though $f_i \oplus g$ is not periodic. From condition (2), we can show that $p^{(i)} \leq 2^{(n+1)/2}((1 + \frac{1}{2})/2)^{cn/2}$ holds (see Lemma 1 in the full version of the paper [5]).

Remark 2. Alg-PolyQ2 finds the index i such that $f_i \oplus g$ has a period, but does not return the actual period of $f_i \oplus g$. However, we can find the actual period of $f_i \oplus g$ (after finding i with Alg-PolyQ2) by applying Simon's algorithm to $f_i \oplus g$.

Summary. With Alg-PolyQ2, we realize an "asymmetric" variant of Simon's algorithm, in which we store a "compressed" database for a single function g, which is not modified (up to some errors) while we test whether another function f has a hidden shift with g, or not. An immediate application of this algorithm will be to achieve an exponential improvement of the query complexity of some Q2 attacks on symmetric schemes. Indeed, in the context where Simon's algorithm and Grover's algorithm are combined, it may be possible to perform the queries to the secret-key cryptographic oracle only once, and so, to lower the query complexity to $\mathcal{O}(n)$.

3.3 Asymmetric Search with Q1 Queries

In Alg-PolyQ2, (online) queries to g and (offline) queries to F are separated, and only cn superposition queries to g are made. Hence another tradeoff is at our reach, which was not possible when g was queried in each Grover iteration: removing superposition queries to g completely.

Algorithm 4. Producing the g-database $|\psi_g\rangle$.

Input: Classical query access to g
Output: The g-database:

$$|\psi_g\rangle = \bigotimes^{cn} \left(\sum_{x \in \{0,1\}^n} |x\rangle|g(x)\rangle \right)$$

1: Start with the all-zero state

$$\bigotimes^{cn} |0\rangle|0\rangle$$

2: Apply Hadamard gates:

$$\bigotimes^{cn} \sum_{x \in \{0,1\}^n} |x\rangle|0\rangle$$

3: For each $y \in \{0,1\}^n$, query (classically) $g(y)$, then apply a unitary which writes $g(y)$ in the second register if the first contains the value y.

This requires now to query *the whole codebook* for g to prepare the quantum state $|\psi_g\rangle$. Once $|\psi_g\rangle$ is built, the second offline phase runs in exactly the same way. Building $|\psi_g\rangle$ costs roughly 2^n time (and classical queries), while going through the search space for f takes $2^{m/2}$ iterations (and quantum queries to F). We call our new algorithm in the Q1 model Alg-ExpQ1 because it will be applied to make Q1 attacks with *exponentially* many online queries in later sections. The optimal point arrives when $m = 2n$.

Below we give an intuitive description of our algorithm Alg-ExpQ1 to solve Problem 3 in the Q1 model. As described above, the difference between Alg-ExpQ1 and Alg-PolyQ2 is the online phase to prepare $|\psi_g\rangle$.

Algorithm Alg-ExpQ1**(informal)**

1. Online phase: Make 2^n classical queries to g and prepare the state $|\psi_g\rangle$.
2. Offline computations: Run the Grover search over $i \in \{0,1\}^m$. For each fixed i, run a testing procedure test such that: (a) test checks if i is a good element (i.e., $f_i \oplus g$ has a period) by using $|\psi_g\rangle$ and making queries f_i, and (b) after checking if i is good, appropriately perform uncomputations to recover the quantum data $|\psi_g\rangle$.

A formal description of Alg-ExpQ1 is the same as that of Alg-PolyQ2 (Algorithm 2) except that we make 2^n classical queries to g to prepare the quantum state $|\psi_g\rangle$. See Algorithm 4 for formal description of the online phase.

The complexity and success probability (including the errors) of Alg-ExpQ1 can be analyzed as below.

Proposition 3. *Suppose that m is in $\mathcal{O}(n)$. Let c be a sufficiently large constant.*[4] *Consider the setting of Problem 3: let $i_0 \in \{0,1\}^m$ be the good element such that $g \oplus f_{i_0}$ is periodic and assume that (2) holds. Then* Alg-ExpQ1 *finds i_0 with a probability in $\Theta(1)$ by making $\mathcal{O}(2^n)$ classical queries to g and $\mathcal{O}\left(n2^{m/2}\right)$ quantum queries to F.*[5] *The offline computation (the procedures excluding the ones to prepare the state $|\psi_g\rangle$) of* Alg-ExpQ1 *is done in time $\mathcal{O}\left((n^3 + nT_F)2^{m/2}\right)$, where T_F is the time required to evaluate F once.*

A proof is given in Section A in the full version of the paper [5].

Finding Actual Periods. The above algorithm Alg-ExpQ1 returns the index i_0 such that $f_{i_0} \oplus g$ has a period, but does not return the actual period. Therefore, if we want to find the actual period of $f_{i_0} \oplus g$ after finding i_0, we have to apply Simon's algorithm to $f_{i_0} \oplus g$ again. Now we can make only classical queries to g, though, we can use the same idea with Alg-ExpQ1 to make an algorithm SimQ1 that finds the period of $f_{i_0} \oplus g$. Again, let c be a positive integer constant.

Algorithm SimQ1

1. Make 2^n classical queries to g and prepare the quantum state $|\psi_g\rangle$.
2. Make cn quantum queries to f_{i_0} to obtain the quantum state $\left|\psi_{f_{i_0} \oplus g}\right\rangle = \bigotimes^{cn} \left(\sum_x |x\rangle |f_{i_0}(x) \oplus g(x)\rangle\right)$.
3. Apply $H^{\otimes n}$ to each $|x\rangle$ register to obtain the state

$$\bigotimes^{cn} \left(\sum_{x,u} (-1)^{x \cdot u} |u\rangle |f_{i_0}(x) \oplus g(x)\rangle \right) .$$

[4] See Proposition 5 for a concrete estimate.

[5] Again, in later applications, F will be instantiated with unkeyed primitives, and quantum queries to F are emulated with offline computations of primitives such as block ciphers.

4. Measure all $|u\rangle$ registers to obtain cn vectors $u_1, \ldots, u_{cn}$.
5. Compute the dimension d of the vector space V spanned by $u_1, \ldots, u_{cn}$. If $d \neq n-1$, return $\perp$. If $d = n-1$, compute the vector $v \neq 0^n \in \{0,1\}^n$ that is orthogonal to V.

Obviously the probability that the above algorithm SimQ1 returns the period of $f_{i_0} \oplus g$ is the same as the probability that the original Simon's algorithm returns the period, under the condition that cn quantum queries can be made to the function $f_{i_0} \oplus g$. Thus, from Proposition 1, the following proposition holds.

Proposition 4. *Suppose that $f_{i_0} \oplus g$ has a period $s \neq 0^n$ and satisfies*

$$\max_{t \neq \{s, 0^n\}} \Pr_x \left[(f_{i_0} \oplus g)(x \oplus t) = (f_{i_0} \oplus g)(x)\right] \leq \frac{1}{2}. \tag{5}$$

Then SimQ1 *returns s with a probability at least $1 - 2^n \cdot (3/4)^{cn}$ by making $\mathcal{O}(2^n)$ classical queries to g and cn quantum queries to f_{i_0}. The offline computation of* SimQ1 *(the procedures excluding the ones to prepare the state $|\psi_g\rangle$) runs in time $\mathcal{O}\left(n^3 + nT_f\right)$, where T_f is the time required to evaluate f_{i_0} once.*

Proposition 5 (Concrete cost estimates). *In practice, for Propositions 2 and 3, $c \simeq m/\left(n \log_2(4/3)\right)$ is sufficient.*

Proof. We need to have $4\lfloor \pi / \left(4 \arcsin\left(2^{-m/2}\right)\right) \rfloor 2^{(n+1)/2} (3/4)^{cn/2} < 1/2$.

In practice, $\arcsin(x) \simeq x$ and the rounding has a negligible impact. Hence, we need that $m/2 + (n+1)/2 + \log_2(\pi) + \log_2(3/4)cn/2 < -1$.

This reduces to $c > \log_2(4/3)^{-1} \left(m + 3 + 2\log_2(\pi)\right)/n \simeq m/\left(n \log_2(4/3)n\right)$.

Remark 3. If $m = n$, this means $c \simeq 2.5$, and if $m = 2n$, $c \simeq 5$.

4 Q2 Attacks on Symmetric Schemes with Reduced Query Complexity

This section shows that our new algorithm Alg-PolyQ2 can be used to construct Q2 attacks on various symmetric schemes. By using Alg-PolyQ2 we can exponentially reduce the number of quantum queries to the keyed oracle compared to previous Q2 attacks, with the same time cost.

In each application, we consider that one evaluation of each primitive (e.g., a block cipher) can be done in time $\mathcal{O}(1)$, for simplicity. For our practical estimates, we use the cost of the primitive as our unit, and consider that it is greater than the cost of solving the linear equation system. In addition, we assume that key lengths of n-bit block ciphers are in $\mathcal{O}(n)$, which is the case for usual block ciphers.

4.1 An Attack on the FX Construction

Here we show a Q2 attack on the FX construction. As described in Sect. 3, the FX construction builds an n-bit block cipher $\mathrm{FX}_{k,k_{in},k_{out}}$ with $(2n+m)$-bit keys ($k_{in}, k_{out} \in \{0,1\}^n$ and $k \in \{0,1\}^m$) from another n-bit block cipher E_k with m-bit keys as

$$\mathrm{FX}_{k,k_{in},k_{out}}(x) := E_k(x \oplus k_{in}) \oplus k_{out}. \tag{6}$$

This construction is used to obtain a block cipher with long ($(2n+m)$-bit) keys from another block cipher with short (m-bit) keys. Roughly speaking, in the classical setting, the construction is proven to be secure up to $\mathcal{O}\left(2^{(n+m)/2}\right)$ queries and computations if the underlying block cipher is secure [26].

Concrete block ciphers such as DESX, proposed by Rivest in 1984 and analyzed in [26], PRINCE [8], and PRIDE [1] are designed based on the FX construction. To estimate security of these block ciphers against quantum computers, it is important to study quantum attacks on the FX construction.

As briefly explained in Sect. 3, the previous Q2 attack by Leander and May [31] breaks the FX construction by making $\mathcal{O}\left(n2^{m/2}\right)$ quantum queries, and its time complexity is $\mathcal{O}\left(n^3 2^{m/2}\right)$.

Application of Our Algorithm Alg-PolyQ2. Below we show that, by applying our algorithm Alg-PolyQ2, we can recover keys of the FX construction with only $\mathcal{O}(n)$ quantum queries. The time complexity of our attack remains $\mathcal{O}\left(n^3 2^{m/2}\right)$, which is the same as Leander and May's.

Attack Idea. As explained in Sect. 3, the problem of recovering the keys k and k_{in} of the FX construction $F_{k,k_{in},k_{out}}$ can be reduced to Problem 3: Define $F : \{0,1\}^m \times \{0,1\}^n \to \{0,1\}^n$ and $g : \{0,1\}^n \to \{0,1\}^n$ by

$$\begin{aligned} F(i,x) &= E_i(x) \oplus E_i(x \oplus 1) \\ g(x) &= \mathrm{FX}_{k,k_{in},k_{out}}(x) \oplus \mathrm{FX}_{k,k_{in},k_{out}}(x \oplus 1). \end{aligned}$$

Then

$$f_k(x) \oplus g(x) = f_k(x \oplus k_{in}) \oplus g(x \oplus k_{in}) \tag{7}$$

holds, i.e., $f_k \oplus g(x)$ has a period k_{in} (note that $f_k(x) = F(k,x)$). If E is a secure block cipher and E_i is a random permutation for each i, intuitively, $f_i \oplus g$ does not have any period for $i \neq k$. Thus the problem of recovering k and k_{in} is reduced to Problem 3 and we can apply our algorithm Alg-PolyQ2. Formally, the attack procedure is as follows.

Attack Description

1. Run Alg-PolyQ2 for the above F and g to recover k.
2. Apply Simon's algorithm to $f_k \oplus g$ to recover k_{in}.
3. Compute $k_{out} = E_k(0^n) \oplus \mathrm{FX}_{k,k_{in},k_{out}}(0^n)$.

Next we give a complexity analysis of the above attack.

Analysis. We assume that $m = \mathcal{O}(n)$, which is the case for usual block ciphers. If E is a secure block cipher and E_i is a random permutation for each $i \in \{0,1\}^m$, we can assume that $f_k \oplus g = E_k \oplus E_k(\cdot \oplus 1) \oplus \mathrm{FX}_{k,k_{in},k_{out}} \oplus \mathrm{FX}_{k,k_{in},k_{out}}(\cdot \oplus 1)$ is far from periodic for all $i \neq k$, and that assumption (2) in Problem 3 holds.

Hence, by Proposition 2, Alg-PolyQ2 recovers k with a high probability by making $\mathcal{O}(n)$ quantum queries to g and $\mathcal{O}\left(n2^{m/2}\right)$ quantum queries to F, which implies that k is recovered only with $\mathcal{O}(n)$ quantum queries made to $\mathrm{FX}_{k,k_{in},k_{out}}$, and in time $\mathcal{O}\left(n^3 2^{m/2}\right)$. (Note that one evaluation of g (resp., F) can be done by $\mathcal{O}(1)$ evaluations of $\mathrm{FX}_{k,k_{in},k_{out}}$ (resp., E)).

From Proposition 1, the second step can be done with $\mathcal{O}(n)$ quantum queries in time $\mathcal{O}\left(n^3\right)$. It is obvious that the third step can be done efficiently.

In summary, our attack recovers the keys of the FX construction with a high probability by making $\mathcal{O}(n)$ quantum queries to the (keyed) online oracle, and it runs in time $\mathcal{O}\left(n^3 2^{m/2}\right)$.

Applications to DESX, PRINCE and PRIDE. DESX [26] has a 64-bit state, two 64-bit whitening key and one 56-bit inner key. From Propositions 2 and 5, we can estimate that our attack needs roughly 135 quantum queries and 2^{29} quantum computations of the cipher circuit.

PRINCE [8], and PRIDE [1] are two ciphers using the FX construction with a 64-bit state, a 64-bit inner key and two 64-bit whitening keys. Hence, from Propositions 2 and 5, we can estimate that our attack needs roughly 155 quantum queries and 2^{33} quantum computations of the cipher circuit.

5 Q1 Attacks on Symmetric Schemes

This section shows that our new algorithm Alg-ExpQ1 can be used to construct Q1 attacks on various symmetric schemes, with a tradeoff between online classical queries, denoted below by D, and offline quantum computations, denoted below by T.

All the algorithms that we consider run with a single processor, but they can use quantum or classical memories, whose amount is respectively denoted by Q (number of qubits) and M. Again, we consider that one evaluation of each primitive (*e.g.* a block cipher) can be done in time $\mathcal{O}(1)$, for simplicity, and we assume that key lengths of n-bit block ciphers are in $\mathcal{O}(n)$.

5.1 Tradeoffs for the Even-Mansour Construction

The Even-Mansour construction [19] is a simple construction to make an n-bit block cipher E_{k_1,k_2} from an n-bit public permutation P and two n-bit keys k_1, k_2 (see Fig. 3). The encryption E_{k_1,k_2} is defined as $E_{k_1,k_2}(x) := P(x \oplus k_1) \oplus k_2$, and the decryption is defined accordingly.

In the classical setting, roughly speaking, the Even-Mansour construction is proven secure up to $\mathcal{O}\left(2^{n/2}\right)$ online queries and offline computations [19]. In fact there exists a classical attack with tradeoff $TD = 2^n$, which balances at $T = D = 2^{n/2}$ [15].

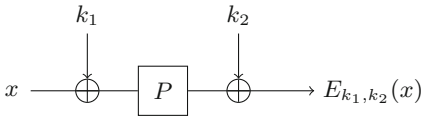


Fig. 3. The Even-Mansour construction.

Previous Q1 Attacks on the Even-Mansour Construction. Kuwakado and Morii gave a Q1 attack that recovers keys of the Even-Mansour construction with $\mathcal{O}\left(2^{n/3}\right)$ classical queries and qubits, and $\mathcal{O}\left(2^{n/3}\right)$ offline quantum computations [30]. Their attack is based on a claw-finding algorithm by Brassard *et al.* [9], and gives the tradeoff $T^2D = 2^n$, with additional $Q = D$ qubits. The balanced point $2^{n/3}$ is significantly smaller than the classical balanced point $2^{n/2}$. However, if we want to recover keys with this attack in time $T \ll 2^{n/2}$, we need an exponential amount of qubits.

Main Previous Attacks with Polynomial Qubits. The best classical attacks allow a trade-off of $D \cdot T = 2^n$ (see [18] for other trade-offs involving memory). With Grover we could recover the keys with a complexity of $2^{n/2}$ and 2 plaintexts-ciphertext pairs, (P_1, C_1) and (P_2, C_2), by performing an exhaustive search over the value of k_1 that would verify $P(P_1 \oplus k_1) \oplus P(P_2 \oplus k_1) = C_1 \oplus C_2$. In [23], Hosoyamada and Sasaki also gave a tradeoff $D \cdot T^6 = 2^{3n}$ for $D \leq 2^{3n/7}$ under the condition that only polynomially many qubits are available, by using the multi-target preimage search by Chailloux et al. [12]. D and T are balanced at $D = T = 2^{3n/7}$, which is smaller than the classical balanced point $2^{n/2}$. The attack uses only polynomially many qubits, but requires $M = D^{1/3} = 2^{n/7}$ classical memory. At the balanced point, this still represents an exponentially large storage. Note that this is the only previous work that recover keys in time $T \ll 2^{n/2}$ with polynomially many qubits.

Table 2. Tradeoffs for Q1 attacks on the Even-Mansour construction. In this table we omit to write order notations, and ignore polynomial factors in the first and last rows.

Reference	Classical attack	Grover	[23]	[30]	[Ours]
Tradeoff of D and T	$D \cdot T = 2^n$	$T = 2^{n/2}$, $D =$ constant	$D \cdot T^6 = 2^{3n}$ ($D \leq 2^{3n/7}$)	$D = 2^{n/3}$, $T = 2^{n/3}$	$D \cdot T^2 = 2^n$
Num. of qubits	–	$\mathsf{poly}(n)$	$\mathsf{poly}(n)$	$2^{n/3}$	$\mathsf{poly}(n)$
Classical memory	D	$\mathsf{poly}(n)$	$D^{1/3}$	$\mathsf{poly}(n)$	$\mathsf{poly}(n)$
Balanced point of D and T	$2^{n/2}$	–	$2^{3n/7}$	–	$2^{n/3}$

Application of Alg-ExpQ1. We explain how to use our algorithm Alg-ExpQ1 to attack the Even-Mansour construction. The tradeoff that we obtain is $T^2 \cdot D = 2^n$, the same as the attack by Kuwakado and Morii above. It balances at $T = D = 2^{n/3}$, but we use only $\mathsf{poly}(n)$ qubits and $\mathsf{poly}(n)$ classical memory. See Table 2 for comparison of attack complexities under the condition that $\mathsf{poly}(n)$ many qubits are available.

Attack Idea. The core observation of Kuwakado and Morii's polynomial-time attack in the Q2 model [30] is that the n-bit secret key k_1 is the period of the function $E_{k_1,k_2}(x) \oplus P(x)$, and thus Simon's algorithm can be applied if quantum queries to E_{k_1,k_2} are allowed. The key to this exponential speed up (compared to the classical attack) is to exploit the algebraic structure of E_{k_1,k_2} (the hidden period of the function) with Simon's algorithm.

On the other hand, the previous Q1 (classical query) attacks described above use only generic multi-target preimage search algorithms that do not exploit any algebraic structures. Hence being able to exploit the algebraic structure in the Q1 model should give us some advantage.

Our algorithm Alg-ExpQ1 realizes this idea. It first makes classical online queries to emulate the quantum queries required by Simon's algorithm (the g-database above) and then runs a combination of Simon's and Grover's algorithms offline (Grover search is used to find the additional m-bit secret information). A naive way to attack in the Q1 model would be to immediately combine Kuwakado and Morii's Q2 attack with Alg-ExpQ1. However, we would have to query the whole classical codebook to emulate quantum queries, which is too costly (and there is no Grover search step).

Our new attack is as follows: We divide the n-bit key k_1 in $k_1^{(1)}$ of u bits and $k_1^{(2)}$ of $n-u$ bits and apply Simon's algorithm to recover $k_1^{(1)}$, while we guess $k_1^{(2)}$ by the Grover search (see Fig. 4). Then, roughly speaking, Alg-ExpQ1 recovers the key by making $D = 2^u$ classical queries and $T = 2^{(n-u)/2}$ offline Grover search iterations (note that the offline computation cost for Simon's algorithm is $\mathsf{poly}(n)$ and we ignore polynomial factors here for simplicity), which yields the tradeoff $D \cdot T^2 = 2^n$, only with $\mathsf{poly}(n)$ qubits and $\mathsf{poly}(n)$ classical space.

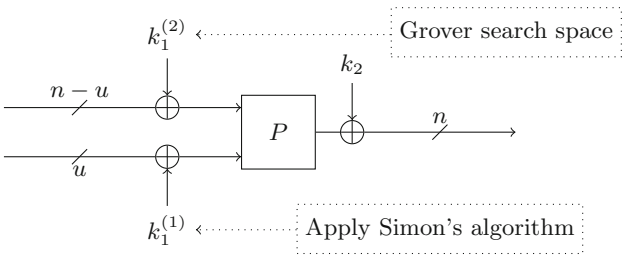


Fig. 4. Idea of our Q1 attack on the Even-Mansour construction.

Attack Description. Here we give the description of our Q1 attack. Let u be an integer such that $0 \leq u \leq n$. Define $F : \{0,1\}^{n-u} \times \{0,1\}^u \to \{0,1\}^n$ by

$$F(i,x) = P(x\|i), \tag{8}$$

and define $g : \{0,1\}^u \to \{0,1\}^n$ by $g(x) = E_{k_1,k_2}(x\|0^{n-u})$.

Note that $F(k_1^{(2)},x) \oplus g(x)$ has the period $k_1^{(1)}$ since $F(k_1^{(2)},x) \oplus g(x) = P(x\|k_1^{(2)}) \oplus P((x \oplus k_1^{(1)})\|k_1^{(2)}) \oplus k_2$. Our attack is described as the following procedure:

1. Run Alg-ExpQ1 for the above F and g to recover $k_1^{(2)}$.
2. Recover $k_1^{(1)}$ by applying SimQ1 to $f_{k_1^{(2)}}$ and g.
3. Compute $k_2 = E_{k_1,k_2}(0^n) \oplus P(k_1)$.

Analysis. Below we assume that u is not too small, *e.g.*, $u \geq n/\log_2 n$. This assumption is not an essential restriction since, if u is too small, then the complexity of the first step becomes almost the same as the Grover search on k_1, which is not of interest.

If P is a random permutation, we can assume that $f_i \oplus g = P(\cdot||i) \oplus E_{k_1,k_2}(\cdot||0^{n-u})$ is far from periodic for all $i \neq k_1^{(2)}$, and that assumption (2) in Problem 3 holds.

Hence, by Proposition 3, Alg-ExpQ1 in Step 1 recovers $k_1^{(2)}$ with a high probability by making $\mathcal{O}(2^u)$ classical queries to g and the offline computation of Alg-ExpQ1 runs in time $\mathcal{O}\left(n^3 2^{(n-u)/2}\right)$. Here, notice that one evaluation of g (resp. F) can be done in $\mathcal{O}(1)$ evaluations of E_{k_1,k_2} (resp. P). In addition, from Proposition 4, SimQ1 in Step 2 recovers $k_1^{(1)}$ with a high probability by making $\mathcal{O}(2^u)$ classical queries to g and the offline computation of Alg-ExpQ1 runs in time $\mathcal{O}\left(n^3\right)$. Step 3 requires $\mathcal{O}(1)$ queries to E_{k_1,k_2} and $\mathcal{O}(1)$ offline computations.

In summary, our attack recovers keys of the Even-Mansour construction with a high probability by making $D = \mathcal{O}(2^u)$ classical queries to E_{k_1,k_2} and doing $T = \mathcal{O}\left(n^3 2^{(n-u)/2}\right)$ offline computations, which balances at $T = D = \tilde{\mathcal{O}}(2^{n/3})$. By construction of Alg-ExpQ1 and SimQ1, our attack uses poly(n) qubits and poly(n) classical memory.

Applications to Concrete Instances. The Even-Mansour construction is a commonly used cryptographic construction. The masks used in Even-Mansour are often derived from a smaller key, which can make a direct key-recovery using Grover's algorithm more efficient. This is for example the case in the CAESAR candidate Minalpher [38]. In general, we need to have a secret key of at least two thirds of the state size for our attack to beat the exhaustive search.

The Farfalle construction [2] degenerates to an Even-Mansour construction if the input message is only 1 block long (Fig. 5). Instances of this construction use variable states and key sizes. The Kravatte instance [2] has a state size of 1600 bits, and a key size between 256 and 320 bits, which leads to an attack at

a whopping cost of 2^{533} data and time, while the direct key exhaustive seach would cost at most 2^{160}. Xoofff [16] has a state size of 384 bits and a key size between 192 and 384 bits. Our attack needs 2^{128} data, which is exactly the data limit of Xoofff. Hence, it is relevant if the key size is greater than 256.

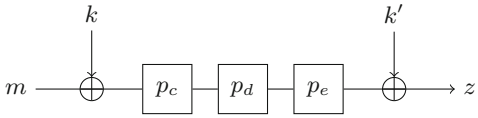


Fig. 5. One-block Farfalle.

5.2 Tradeoffs for the FX Construction

The FX construction [26] $\mathrm{FX}_{k,k_{in},k_{out}}$, computes a ciphertext c from a plaintext p by $c \leftarrow E_k(p \oplus k_{in}) \oplus k_{out}$, where E is a block cipher, k is an m-bit key and k_{in}, k_{out} are two n-bit keys. In the classical setting, there exists a classical attack with tradeoff $TD = 2^{n+m}$, which balances at $T = D = 2^{(n+m)/2}$ (see, for example, [17] for more details and memory trade-offs).

Previous Q1 Attacks on the FX Construction. Applying Grover as we did before on Even-Mansour on the keys k_{in} and k, we can recover the keys with only two pairs of plaintext-ciphertext and a time complexity of $2^{(n+m)/2}$, while only needing a polynomial number of qubits.

In [23], Hosoyamada and Sasaki proposed a tradeoff $D \cdot T^6 = 2^{3(n+m)}$ for $D \leq \min\{2^n, 2^{3(n+m)/7}\}$ with a polynomial amount of qubits, by using the multi-target preimage search by Chailloux et al. [12]. The balance occurs at $D = T = 2^{3(n+m)/7}$ (if $m \leq 4n/3$), which is smaller than the classical balanced point $2^{(n+m)/2}$. The attack requires $M = D^{1/3}$ classical memory, thus the attack still requires exponentially large space at the balanced point. This was the only Q1 attack with time $T \ll 2^{(n+m)/2}$ and a polynomial amount of qubits.

Application of Alg-ExpQ1. We explain how to apply our algorithm Alg-ExpQ1 to the FX construction. Our new tradoff is $T^2 \cdot D = 2^{n+m}$ for $D \leq 2^n$, which balances at $T = D = 2^{(n+m)/3}$ (for $m \leq 2n$), using only $\mathsf{poly}(n)$ qubits and $\mathsf{poly}(n)$ classical memory. See Table 3 for comparison of attack complexities under the condition that only $\mathsf{poly}(n)$ qubits are available.

Attack Idea. Recall that, in the Q1 attack on the Even-Mansour construction in Sect. 5.1, we divided the first key k_1 to two parts $k_1^{(1)}$ and $k_1^{(2)}$ and applied Simon's algorithm to $k_1^{(1)}$ while we performed Grover search on $k_1^{(2)}$.

In a similar manner, for the FX construction $\mathrm{FX}_{k,k_{in},k_{out}}$ we divide the n-bit key k_{in} in $k_{in}^{(1)}$ of u bits and $k_{in}^{(2)}$ of $(n-u)$ bits. We apply Simon's algorithm to recover $k_{in}^{(1)}$ while we perform Grover search on k in addition to $k_{in}^{(2)}$ (see Fig. 6).

Table 3. Tradeoffs for Q1 attacks on the FX construction. In this table we omit to write order notations, and ignore polynomial factors in the first and last rows.

Reference	Classical attack	Grover	[23]	[Ours]
Tradeoff of D and T	$D \cdot T = 2^{n+m}$ ($D \leq 2^n$)	$T = 2^{(n+m)/2}$ $D =$ constant	$D \cdot T^6 = 2^{3(n+m)}$ ($D \leq \min\{2^n, 2^{3n/7}\}$)	$D \cdot T^2 = 2^{n+m}$ ($D \leq 2^n$)
Num. of qubits	–	$\mathsf{poly}(n)$	$\mathsf{poly}(n)$	$\mathsf{poly}(n)$
Class. memory	D	$\mathsf{poly}(n)$	$D^{1/3}$	$\mathsf{poly}(n)$
Balanced point of D and T	$2^{(n+m)/2}$ ($m \leq n$)	–	$2^{3(n+m)/7}$ ($m \leq 4n/3$)	$2^{(n+m)/3}$ ($m \leq 2n$)

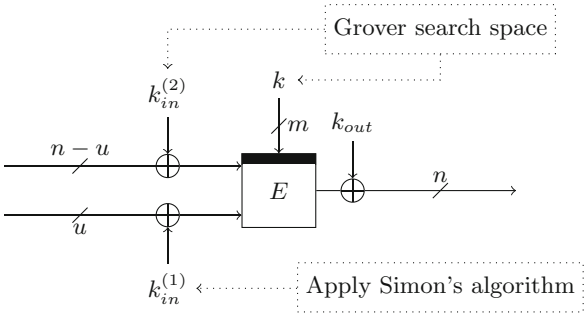


Fig. 6. Idea of our Q1 attack on the FX construction.

Then, roughly speaking, by applying Alg-ExpQ1 we can recover the key by making $D = 2^u$ classical queries and $T = 2^{(n-u)/2}$ offline Grover iterations (note that the offline computation cost for the Simon's algorithm is $\mathsf{poly}(n)$ and we ignore polynomial factors here for simplicity), which yields the tradeoff $D \cdot T^2 = 2^{(n+m)}$ for $D \leq 2^n$, with only $\mathsf{poly}(n)$ qubits and $\mathsf{poly}(n)$ classical memories.

Attack Description. Here we give the description of our Q1 attack. Let u be an integer such that $0 \leq u \leq n$. Define $F : \{0,1\}^{m+(n-u)} \times \{0,1\}^u \to \{0,1\}^n$ by

$$F(i\|j, x) = E_i(x\|j)(i \in \{0,1\}^m, j \in \{0,1\}^{n-u}), \tag{9}$$

and define $g : \{0,1\}^u \to \{0,1\}^n$ by $g(x) = \mathrm{FX}_{k,k_{in},k_{out}}(x\|0^{n-u})$.

Note that $F(k\|k_{in}^{(2)}, x) \oplus g(x)$ has the period $k_{in}^{(1)}$ since $F(k\|k_{in}^{(2)}, x) \oplus g(x) = E_k(x\|k_{in}^{(2)}) \oplus E_k((x \oplus k_{in}^{(1)})\|k_{in}^{(2)}) \oplus k_{out}$. Our attack procedure runs as follows:

1. Run Alg-ExpQ1 for the above F and g to recover k and $k_{in}^{(2)}$.
2. Recover $k_{in}^{(1)}$ by applying SimQ1 to $f_{k\|k_{in}^{(2)}}$ and g.
3. Compute $k_{out} = \mathrm{FX}_{k,k_{in},k_{out}}(0^n) \oplus E_k(k_{in})$.

Analysis. We assume that $m = \mathcal{O}(n)$, which is the case for usual block ciphers. In the same way as in the analysis for the attack on the Even-Mansour construction in Sect. 5.1, if E is a (pseudo) random permutation family and u is not too small (*e.g.* $u \geq n/\log_2 n$), we observe that the assumption (2), rephrased as:

$$\max_{t\in\{0,1\}^n\setminus\{0^n\}} \Pr_{x\in\{0,1\}^n} \Big[E_i(x||j) \oplus E_i(x\oplus t||j) \oplus E_k\left(x \oplus k_{in}^{(1)}||k_{in}^{(2)}\right) \oplus E_k\left(x\oplus t\oplus k_{in}^{(1)}||k_{in}^{(2)}\right) = 0\Big] \leq 1/2 \tag{10}$$

holds for all $(i,j) \neq (k, k_1^{(2)})$ with overwhelming probability.

This again implies that the claims of Propositions 3 and 4 hold for Alg-ExpQ1 in Step 1 and SimQ1 in Step 2, respectively.

Thus our attack recovers keys of the FX construction with a high probability by making $D = \mathcal{O}(2^u)$ classical queries to $\mathrm{FX}_{k,k_{in},k_{out}}$ and doing $T = \mathcal{O}\left(n^3 2^{(m+n-u)/2}\right)$ offline computations for $D \leq 2^n$, which balances at $T = D = \tilde{\mathcal{O}}(2^{(n+m)/3})$ if $m \leq 2n$. Our attack uses only $\mathsf{poly}(n)$ qubits and $\mathsf{poly}(n)$ classical memory by construction of Alg-ExpQ1 and SimQ1.

Application to Concrete Instances. DESX [26] has a 64-bit state, two 64-bit whitening key and one 56-bit inner key. From Propositions 2 and 5, we can estimate that our attack needs roughly 2^{42} classical queries and 2^{40} quantum computations of the cipher circuit.

PRINCE [8], and PRIDE [1] are two ciphers using the FX construction with a 64-bit state, a 64-bit inner key and two 64-bit whitening keys. Hence, from Propositions 2 and 5, we can estimate that our attack needs roughly 2^{45} quantum queries and 2^{43} quantum computations of the cipher circuit.

We can also see some encryption modes as an instance of the FX construction. This is for example the case of the XTS mode [32], popular for disk encryption. It is generally used with AES-256 and two whitening keys that depend on the block number and another 256-bit key. Hence, with the full codebook of one block, we can obtain the first key and the value of the whitening keys of the corresponding block. Once the first key is known, the second can easily be brute-forced from a few known plaintext-ciphertext couples in other blocks.

Adiantum [14] is another mode for disk encryption that uses a variant of the FX construction with AES-256 and Chacha. There is however one slight difference: the FX masking keys are added with a modular addition instead of a xor. The FX construction is still vulnerable [6], but we will need to use Kuperberg's algorithm [28] instead of Simon's algorthm. As before, with the full codebook on one block, we can recover the AES and Chacha keys in a time slightly larger than 2^{256}.

5.3 Other Applications

Chaskey. The lightweight MAC Chaskey [33] is very close to an Even-Mansour construction (see Fig. 7). Since the last message block (m_2 in Fig. 7) is XORed

to the key K_1, we can immediately apply our Q1 attack and recover K_1 and the value of the state before the xoring of the last message block. As π is a permutation easy to invert, this allows to retrieve K. The Chaskey round function applies on 128 bits. It contains 16 rounds with 4 modular additions on 32 bits, 4 XORs on 32 bits and some rotations. With a data limit of 2^{48}, as advocated in the specification, our attack would require roughly $2^{(128-48)/2} \times 2^{19} = 2^{59}$ quantum gates, where the dominant cost is solving the 80-dimensional linear system inside each iteration of Grover's algorithm.

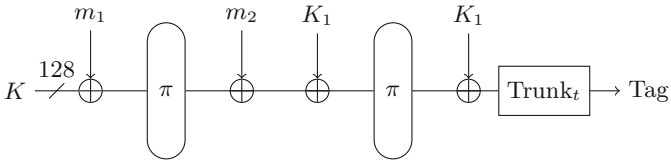


Fig. 7. Two-block Chaskey example.

Sponges. Our attack can be used on sponges if there is an input injected on a fixed state. In general, it has two drawbacks: the nonce has to be fixed, and the cost of the attack is at least $2^{c/2}$ with c the capacity of the sponge, which is often the classical security parameter. However, there are some cases where our attack is of interest.

In particular, our attack needs a set of values that contains an affine space. If a nonce was injected the same way the messages are, then we only need to know the encryptions of identical messages, with a set of nonces that fills an affine space. Nonce-respecting adversaries are generally allowed to choose the nonce, but here, the mere assumption that the nonce is incremented for each message (which is the standard way nonces are processed in practice) is sufficient: A set of 2^k consecutive values contains an affine space of $(\mathbb{Z}/(2))^{k-1}$.

This is the case in the Beetle mode of lightweight authenticated encryption [13], whose initialization phase is described as $(K_1 \oplus N)\|K_2 \mapsto f((K_1 \oplus N)\|K_2)$, where $K_1, N \in \{0,1\}^r$, $K_2 \in \{0,1\}^c$, and f is a $(r+c)$-bit permutation (Fig. 8).

Here, the nonce is directly added to the key K_1, but as the key has the same length as the state, the attack would still work if the nonce was added after the first permutation. In Beetle[Light+], the rate is $r = 64$ bits and the capacity $c = 80$ bits. The rate is sufficiently large to embed 48 varying bits for the nonce; in that case, by making 2^{48} classical queries and 2^{48} Grover iterations, we can recover the secret $K_1||K_2$. In Beetle[Secure+], $r = c = 128$ bits. We can recover $K_1||K_2$ with 2^{85} messages and Grover iterations.

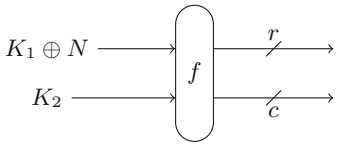


Fig. 8. Beetle state initialization.

6 Discussion

In this section, we discuss on the application of our attack idea to related-key attacks, to some slide attacks, and to an extension of Problem 3. See also Section B in the full version of the paper [5] for discussions on adaptive attacks and non-adaptive attacks.

6.1 Related Keys

Consider a block cipher E_k with a key and block size of n bits. In the related-key setting, as introduced in [41], we are not only allowed to make chosen plaintext or ciphertext queries to a secret-key oracle hiding k, but also to query $E_{k\oplus\ell}(m)$ for any n-bit difference ℓ and message m. Classically, this is a very powerful model, but it becomes especially meaningful when the block cipher is used inside a mode of operation (*e.g.* a hash function) in which key differences can be controlled by the attacker. It is shown in [41] that a secret key recovery in this model can be performed in $2^{n/2}$ operations, as it amounts to find a collision between some query $E_{k\oplus\ell}(m)$ and some offline computation $E_{\ell'}(m)$ (we can use more than a single plaintext m to ensure an overwhelming success probability).

Rötteler and Steinwandt [37] noticed that, if a quantum adversary has *superposition* access to the oracle that maps ℓ to $E_{k\oplus\ell}(m)$, it can mount a key-recovery in polynomial time using Simon's algorithm. Indeed, one can define a function:

$$f(x) = E_{k\oplus x}(m) \oplus E_x(m)$$

which has k as hidden period, apply Simon's algorithm and recover k. This attack works for any block cipher, even ideal. In contrast, in the Q2 quantum attacker model, we know that some constructions are broken, but it does not seem to be the case for all of them.

With our algorithm Alg-ExpQ1, we are able to translate this related-key superposition attack into an attack where the related-key oracle is queried only classically, but the attacker has quantum computing power. We write $k = k_1||k_2$ where k_1 has $n/3$ bits and k_2 has $2n/3$ bits. We query $E_{(k_1||k_2)\oplus(\ell_1||0)}(m)$ for a fixed m and all $n/3$-bit differences ℓ_1. Then we perform a Grover search on k_2. The classical security level in presence of a related-key oracle of this form, which is $2^{n/2}$, is reduced quantumly to $2^{n/3}$. This shows that the transition to a quantum setting has an impact on the related-key security even if the oracle remains classical.

As a consequence, we could complete the security claims of the 16-round version of the block cipher SATURNIN [10], a submission to the ongoing NIST lightweight cryptography competition[6]. The authors of SATURNIN gave security claims against quantum attackers meeting the best generic attacks. No claims were given regarding the Q_1 model for related-key attacks. Our result gives the best generic quantum related-key attack on ideal block ciphers without superposition queries, and sets the level of security that should be expected from a block cipher in this setting: the key can be recovered in quantum time $\widetilde{\mathcal{O}}\left(2^{n/3}\right)$ for a block cipher of n bits (and using $2^{n/3}$ classical related-key queries). The corresponding security level for SATURNIN_{16}, which has blocks of 256 bits, lies at $2^{256/3} = 2^{85}$: we can say that in the Q_1 related-key setting, SATURNIN_{16} should have no attack with time complexity lower than 2^{85}.

6.2 Slide Attacks

Quantum slide attacks are a very efficient quantum counterpart of the classical slide attacks [3]. They have been introduced in [24], with a polynomial-time attack on 1-round self-similar ciphers. In many cases, our algorithm does not improve these attacks, because they are already too efficient and do not rely on a partial exhaustive search. Still, some of them use a partial exhaustive search. This is the case of the slide attack against 2 round self-similar ciphers of [31] and the slide attacks against whitened Feistels of [7].

For example, we can see a 2 round self-similar cipher as an example of iterated FX cipher, as in Fig. 9. Define functions p_i, F_i, and g as

$$p_i((b,x)) = \begin{cases} (0, E_i(x)) & \text{if } b = 0 \\ (1, x) & \text{if } b = 1 \end{cases}, \quad F_i((b,x),y) = \begin{cases} y \oplus x & \text{if } b = 0 \\ E_i(y) \oplus x & \text{if } b = 1 \end{cases},$$

and $g((b,x)) = \text{iFX}(x)$. We have the property that $\text{iFX}(E_{k_2}(x \oplus k_1)) \oplus (x \oplus k_1) = E_{k_2}(\text{iFX}(x)) \oplus x$. Hence, we have the hidden period $(1, k_1)$ in the function $f_{k_2}((b,x)) = F_{k_2}\left((b,x), g(p_{k_2}(b,x))\right)$. To apply our attack, we need to compute $\sum_{x,b} |x\rangle|b\rangle|f_i((b,x))\rangle$ from the state $\sum_x |x\rangle|\text{iFX}(x)\rangle$. We first need to add one qubit to obtain $\sum_x |x\rangle(|0\rangle + |1\rangle)|\text{iFX}(x)\rangle$. Then, conditioned on the second register to be 0, we transform x into $E_i^{-1}(x)$. Next, conditioned on the second register to be 1, we transform $\text{iFX}(x)$ into $E_i(\text{iFX}(x))$. Finally, we add the first register to the third. Hence, we can apply our attack, and retrieve k_1 and k_2 using $\mathcal{O}\left(|k_1|\right)$ queries and $\mathcal{O}\left(|k_1|^3 2^{|k_2|/2}\right)$ time, assuming $|k_1| = \Omega(|k_2|)$.

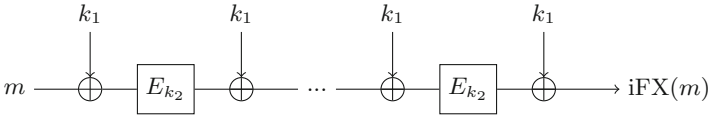


Fig. 9. Iterated-FX cipher.

[6] https://csrc.nist.gov/Projects/Lightweight-Cryptography.

The above problem of recovering keys can be generalized as the following problem, which can be solved by the same strategy as above.

Problem 4 (Constructing and Finding a Hidden Period). Let $g : \{0,1\}^n \to \{0,1\}^\ell$ be a function, $i \in I$, $p_i : \{0,1\}^n \to \{0,1\}^n$ be a permutation and $F_i : \{0,1\}^n \times \{0,1\}^\ell \to \{0,1\}^\ell$ be a function such that $F_i(x,\cdot)$ is a permutation. Assume that there exists $i_0 \in I$ such that $f_{i_0}(x) = F_{i_0}(x, g(p_{i_0}(x)))$ has a period, *i.e.*: $\forall x \in \{0,1\}^n, f_{i_0}(x) = f_{i_0}(x \oplus s)$ for some s. Assume that we are given quantum oracle access to F_i and p_i and classical or quantum oracle access to g. (In the Q1 setting, g will be a classical oracle. In the Q2 setting, g will be a quantum oracle.) Then find i_0 and s.

This problem assumes that g is a keyed function, and that we can reversibly transform $(x, g(x))$ into a couple $(y, f_i(y))$, with f_i a periodic function if $i = i_0$. We can see this transformation as a generalization of the CCZ equivalence [11], where the function mapping the graph of g and the graph of f_i do not need to be an affine function. There may also be more than one solution (in which case we just want to find one), or there may be none, just as Grover's algorithm can handle cases with many expected solutions, or distinguish whether there is a solution or not. Note that Problem 3 is a special case of the above problem, in the case where p_i is the identity, and F_i is only the xoring of g and another function.

7 Conclusion

In this paper, we have introduced a new quantum algorithm, in which we make use of Simon's algorithm in an *offline* way. The idea of making $\mathsf{poly}(n)$ superposition queries to the oracle (with, as input, a uniform superposition), storing them as some compressed database on n^2 qubits, and reusing them during the iterations of a Grover search, yielded surprising results. This idea, initially targeting the query complexity of some Q2 attacks on cryptographic schemes, enabled us to find new quantum-time/classical-data tradeoffs. Our result has three consequences, each of which answers a long-standing question in post-quantum cryptography.

Simon's Algorithm Can Be Used in an Offline Setting. We provided the first example of use of Simon's algorithm (or more precisely, its core idea) in an offline setting, without quantum oracle queries.

Improving More Than the Time Complexity. Consider the example of our attack on the Even-Mansour construction in quantum time $\widetilde{\mathcal{O}}\left(2^{n/3}\right)$ and classical queries $\mathcal{O}\left(2^{n/3}\right)$. With the same number of queries, the classical attack requires $\mathcal{O}\left(2^{2n/3}\right)$ time *and* $\mathcal{O}\left(2^{n/3}\right)$ classical memory to store the queries. In our attack, we do not need this storage. To the best of our knowledge, this is the first time that a quantum Q1 attack provides a quadratic speedup while the needs of hardware are also reduced.

Q2 Attacks Make a Difference. Schemes which do not have an attack in the superposition model cannot be attacked by our algorithm. We showed that their algebraic structure, which makes the superposition attack possible, indeed made a practical difference when it came to Q1 attacks. We believe that this question needs further investigation.

Acknowledgements. The authors thank Léo Perrin for proofreading this article and Elena Kirshanova for helpful remarks. This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement nº 714294 - acronym QUASYModo).

References

1. Albrecht, M.R., Driessen, B., Kavun, E.B., Leander, G., Paar, C., Yalçın, T.: Block ciphers – focus on the linear layer (feat. PRIDE). In: Garay, J.A., Gennaro, R. (eds.) CRYPTO 2014. LNCS, vol. 8616, pp. 57–76. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-662-44371-2_4
2. Bertoni, G., Daemen, J., Hoffert, S., Peeters, M., Assche, G.V., Keer, R.V.: Farfalle: parallel permutation-based cryptography. IACR Trans. Symmetric Cryptol. **2017**(4), 1–38 (2017). https://tosc.iacr.org/index.php/ToSC/article/view/801
3. Biryukov, A., Wagner, D.: Slide attacks. In: Knudsen, L. (ed.) FSE 1999. LNCS, vol. 1636, pp. 245–259. Springer, Heidelberg (1999). https://doi.org/10.1007/3-540-48519-8_18
4. Bonnetain, X.: Quantum key-recovery on full AEZ. In: Adams, C., Camenisch, J. (eds.) SAC 2017. LNCS, vol. 10719, pp. 394–406. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-72565-9_20
5. Bonnetain, X., Hosoyamada, A., Naya-Plasencia, M., Sasaki, Y., Schrottenloher, A.: Quantum attacks without superposition queries: the offline simon algorithm. IACR Cryptology ePrint Archive 2019, 614 (2019). https://eprint.iacr.org/2019/614
6. Bonnetain, X., Naya-Plasencia, M.: Hidden shift quantum cryptanalysis and implications. In: Peyrin, T., Galbraith, S. (eds.) ASIACRYPT 2018. LNCS, vol. 11272, pp. 560–592. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-03326-2_19
7. Bonnetain, X., Naya-Plasencia, M., Schrottenloher, A.: On quantum slide attacks. In: Selected Areas in Cryptography - SAC 2019. Lecture Notes in Computer Science, Springer (2020)
8. Borghoff, J., et al.: PRINCE – a low-latency block cipher for pervasive computing applications. In: Wang, X., Sako, K. (eds.) ASIACRYPT 2012. LNCS, vol. 7658, pp. 208–225. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-34961-4_14
9. Brassard, G., HØyer, P., Tapp, A.: Quantum cryptanalysis of hash and claw-free functions. In: Lucchesi, C.L., Moura, A.V. (eds.) LATIN 1998. LNCS, vol. 1380, pp. 163–169. Springer, Heidelberg (1998). https://doi.org/10.1007/BFb0054319
10. Canteaut, A., et al.: Saturnin: a suite of lightweight symmetric algorithms for post-quantum security (2019). https://project.inria.fr/saturnin/files/2019/05/SATURNIN-spec.pdf
11. Carlet, C., Charpin, P., Zinoviev, V.: Codes, bent functions and permutations suitable for DES-like cryptosystems. Designs Codes Crypt. **15**(2), 125–156 (1998)

12. Chailloux, A., Naya-Plasencia, M., Schrottenloher, A.: An efficient quantum collision search algorithm and implications on symmetric cryptography. In: Takagi, T., Peyrin, T. (eds.) ASIACRYPT 2017. LNCS, vol. 10625, pp. 211–240. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-70697-9_8
13. Chakraborti, A., Datta, N., Nandi, M., Yasuda, K.: Beetle family of lightweight and secure authenticated encryption ciphers. IACR Trans. Crypt. Hardw. Embed. Syst. **2018**(2), 218–241 (2018). https://doi.org/10.13154/tches.v2018.i2.218-241
14. Crowley, P., Biggers, E.: Adiantum: length-preserving encryption for entry-level processors. IACR Trans. Symmetric Cryptol. **2018**(4), 39–61 (2018). https://doi.org/10.13154/tosc.v2018.i4.39-61
15. Daemen, J.: Limitations of the Even-Mansour construction. In: Imai, H., Rivest, R.L., Matsumoto, T. (eds.) ASIACRYPT 1991. LNCS, vol. 739, pp. 495–498. Springer, Heidelberg (1993). https://doi.org/10.1007/3-540-57332-1_46
16. Daemen, J., Hoffert, S., Assche, G.V., Keer, R.V.: The design of xoodoo and xoofff. IACR Trans. Symmetric Cryptol. **2018**(4), 1–38 (2018). https://doi.org/10.13154/tosc.v2018.i4.1-38
17. Dinur, I.: Cryptanalytic time-memory-data tradeoffs for FX-constructions with applications to PRINCE and PRIDE. In: Oswald, E., Fischlin, M. (eds.) EUROCRYPT 2015. LNCS, vol. 9056, pp. 231–253. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-46800-5_10
18. Dinur, I., Dunkelman, O., Keller, N., Shamir, A.: Cryptanalysis of Iterated Even-Mansour schemes with two keys. In: Sarkar, P., Iwata, T. (eds.) ASIACRYPT 2014. LNCS, vol. 8873, pp. 439–457. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-662-45611-8_23
19. Even, S., Mansour, Y.: A construction of a cipher from a single pseudorandom permutation. J. Cryptol. **10**(3), 151–162 (1997). https://doi.org/10.1007/s001459900025
20. Gagliardoni, T.: Quantum Security of Cryptographic Primitives. Ph.D. thesis, Darmstadt University of Technology, Germany (2017). http://tuprints.ulb.tu-darmstadt.de/6019/
21. Grassl, M., Langenberg, B., Roetteler, M., Steinwandt, R.: Applying Grover's algorithm to AES: quantum resource estimates. In: Takagi, T. (ed.) PQCrypto 2016. LNCS, vol. 9606, pp. 29–43. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-29360-8_3
22. Grover, L.K.: A fast quantum mechanical algorithm for database search. In: Miller, G.L. (ed.) Proceedings of the Twenty-Eighth Annual ACM Symposium on the Theory of Computing, Philadelphia, Pennsylvania, USA, 22–24 May 1996, pp. 212–219. ACM (1996). http://doi.acm.org/10.1145/237814.237866
23. Hosoyamada, A., Sasaki, Y.: Cryptanalysis against symmetric-key schemes with online classical queries and offline quantum computations. In: Smart, N.P. (ed.) CT-RSA 2018. LNCS, vol. 10808, pp. 198–218. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-76953-0_11
24. Kaplan, M., Leurent, G., Leverrier, A., Naya-Plasencia, M.: Breaking symmetric cryptosystems using quantum period finding. In: Robshaw, M., Katz, J. (eds.) CRYPTO 2016. LNCS, vol. 9815, pp. 207–237. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-53008-5_8
25. Kaplan, M., Leurent, G., Leverrier, A., Naya-Plasencia, M.: Quantum differential and linear cryptanalysis. IACR Trans. Symmetric Cryptol. **2016**(1), 71–94 (2016). http://tosc.iacr.org/index.php/ToSC/article/view/536

26. Kilian, J., Rogaway, P.: How to protect DES against exhaustive key search. In: Koblitz, N. (ed.) CRYPTO 1996. LNCS, vol. 1109, pp. 252–267. Springer, Heidelberg (1996). https://doi.org/10.1007/3-540-68697-5_20
27. Kuperberg, G.: A subexponential-time quantum algorithm for the dihedral hidden subgroup problem. SIAM J. Comput. **35**(1), 170–188 (2005). https://doi.org/10.1137/S0097539703436345
28. Kuperberg, G.: Another subexponential-time quantum algorithm for the dihedral hidden subgroup problem. In: TQC 2013, LIPIcs, vol. 22, pp. 20–34. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik (2013)
29. Kuwakado, H., Morii, M.: Quantum distinguisher between the 3-round feistel cipher and the random permutation. In: IEEE International Symposium on Information Theory, ISIT 2010, Proceedings, pp. 2682–2685. IEEE (2010)
30. Kuwakado, H., Morii, M.: Security on the quantum-type Even-Mansour cipher. In: Proceedings of the International Symposium on Information Theory and Its Applications, ISITA 2012, pp. 312–316. IEEE (2012)
31. Leander, G., May, A.: Grover meets Simon – quantumly attacking the FX-construction. In: Takagi, T., Peyrin, T. (eds.) ASIACRYPT 2017. LNCS, vol. 10625, pp. 161–178. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-70697-9_6
32. Martin, L.: XTS: a mode of AES for encrypting hard disks. IEEE Secur. Privacy **8**(3), 68–69 (2010). https://doi.org/10.1109/MSP.2010.111
33. Mouha, N., Mennink, B., Van Herrewege, A., Watanabe, D., Preneel, B., Verbauwhede, I.: Chaskey: an efficient MAC algorithm for 32-bit microcontrollers. In: Joux, A., Youssef, A. (eds.) SAC 2014. LNCS, vol. 8781, pp. 306–323. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-13051-4_19
34. National Academies of Sciences, Engineering, and Medicine: Quantum Computing: Progress and Prospects. The National Academies Press, Washington, DC (2018). https://www.nap.edu/catalog/25196/quantum-computing-progress-and-prospects
35. National Institute of Standards and Technlology: Submission requirements and evaluation criteria for the post-quantum cryptography standardization process (2016). https://csrc.nist.gov/CSRC/media/Projects/Post-Quantum-Cryptography/documents/call-for-proposals-final-dec-2016.pdf
36. Nielsen, M.A., Chuang, I.: Quantum Computation and Quantum Information. AAPT (2002)
37. Rötteler, M., Steinwandt, R.: A note on quantum related-key attacks. Inf. Process. Lett. **115**(1), 40–44 (2015). https://doi.org/10.1016/j.ipl.2014.08.009
38. Sasaki, Y., et al.: Minalpher v1.1. CAESAR competition (2015). https://competitions.cr.yp.to/round2/minalpherv11.pdf
39. Shor, P.W.: Algorithms for quantum computation: discrete logarithms and factoring. In: 35th Annual Symposium on Foundations of Computer Science, pp. 124–134. IEEE Computer Society (1994)
40. Simon, D.R.: On the power of quantum computation. In: 35th Annual Symposium on Foundations of Computer Science, pp. 116–123 (1994)
41. Winternitz, R.S., Hellman, M.E.: Chosen-key attacks on a block cipher. Cryptologia **11**(1), 16–20 (1987). https://doi.org/10.1080/0161-118791861749