ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Quantumania: Three Quantum Attacks on AES-OTR's Confidentiality and a Quantum Key-Recovery Attack on OPP

Bachelor Thesis

Melanie Jauch

March 26, 2023

Advisors: Prof. Dr. Kenny Paterson, Varun Maram

Applied Cryptography Group
Institute of Information Security
Department of Computer Science, ETH Zürich

Abstract

The impact of quantum computers on symmetric-key cryptography has long been considered to be low due to Grover's algorithm, which only improves attacks on primitives such as block ciphers by a quadratic speed up. However, in recent works, the quantum security of certain block cipher modes of operation has been broken in polynomial time in a setting where the adversary has access to a quantum encryption oracle, i.e., is allowed to make encryption queries in a quantum superposition of states. In particular, in this model works by Maram *et al.* (ToSC 2022) and Chang *et al.* (Symmetry 2022) have shown how to break unforgeability ((E/U)UF-qCMA security) and confidentiality (IND-qCPA security) of OCB, and unforgeability of the AES-OTR authenticated encryption (AE) modes respectively.

However, the confidentiality of AES-OTR and the quantum security of other OCB-like modes of operation, such as OPP, have not been addressed. This is where this thesis steps in and presents the first quantum key recovery attack on OPP and for the first time breaks the IND-qCPA security of AES-OTR in three different settings: Since AES-OTR comes with two different ways for authenticating associated data (AD), we present an attack for each of them. To complete the analysis, we restrict the mode and consider it as a pure AE scheme (i.e., it does not use AD) and break IND-qCPA security once again.

Contents

Chapter 1

Introduction

With the rise of the development of quantum computers, the security of cryptographic systems faces new attack threats, including a speed up of existing attacks. Opposed to using binary digits (bits) as classical computers do, quantum computers are based on calculations with quantum bits (qubits), that can exist in many different states at the same time. As a consequence, quantum computers exceed the abilities of a classical computer and thus commonly used public-key cryptosystems, that rely on the hardness of computing factorization and discrete logarithms, would suffer from devastating attacks that are based on Shor's algorithm [Sho99].

The threat of quantum computers is being addressed by a new research area called *post-quantum cryptography*. There it is assumed, that the adversary has access to a quantum computer. This setting has gained a lot of attention during the last years and the desire of using quantum-resistant cryptographic algorithms arose. This desire has been approached by a six year competition of NIST in an effort for standardizing such algorithms. A first conclusion has been made and four algorithms for public-key encryption, key establishment and digital signatures have been selected for standardisation [ACD+22]. Regardless, the competition did not reach its final conclusion yet and a lot of analysis is to be made.

Coming to the symmetric-key cryptography setting, the impact of quantum computers has been assumed to be significantly less severe. Due to Grover's algorithm [Gro96], it was believed that attacks on symmetric-key primitives such as block ciphers would only improve by a quadratic speed up. Naturally one could assume, that it is enough to just simply double the key size of the affected primitives in order to restore the same level of security as before. In spite of that, it has soon been realized that it is not sufficient to just consider the quantum security of standalone primitives such as block ciphers. Since block ciphers are only able to encrypt plaintexts one block at a time, they are usually used in the form of so called *modes of operation* that

allow the user to encrypt plaintext of arbitrary lengths. These modes are designed and extended upon the security of its underlying block cipher and can achieve several desired security goals such as confidentiality, integrity and authenticity depending on requirement. As modes of operation are further developing the use of block ciphers, they also pose new possibilities for designing innovative attacks. Following this, it is important to analyze the quantum security of modes of operation as well to ensure that security persists.

It is to be noted that there exists a difference between post-quantum and quantum security: opposed to the post-quantum setting, where the adversary has quantum access only to public oracles such as hash functions, the quantum security setting goes one step further and assumes that the adversary also has quantum access to secret keyed encryption or decryption oracles. This is referred to as making quantum encryption queries (resp. quantum decryption queries, in case of decryption oracles), which essentially means that an adversary can ask for the encryption of quantum states that are quantum superpositions of messages and receives superpositions of ciphertexts as an answer. However, the practicality of this model is still currently debated (e.g. in [KLLN16, BBC$^+$21, ABKM22, JST21, BHN$^+$19]) and to be determined. The quantum superposition model still led to interesting observations and devastating attacks on modes of operation such as CBC, CFB, CTR and XTS as described in [ATTU16] as well as CBC-MAC, and GCM as described in [KLLN16, BBC$^+$21]. These attacks break the security guarantees of confidentiality (IND-CPA security) and unforgeability ((E/U)UF-CMA security). To do so, the classical definitions of IND-CPA security and (E/U)UF-CMA security have been extended to a setting where the adversary is allowed to make quantum encryption queries. These notions are therefore referred to as *IND-qCPA* and *(E/U)UF-qCMA* security.

More recently, the IND-qCPA security of the OCB modes has been analyzed in [MMPR22]. As the OCB modes, namely OCB1 [RBBK01], OCB2(f)[1] [Rog04] and OCB3 [KR11] are widely influential and well-studied authenticated encryption modes, they extended upon the already existing analysis on quantum forgery attacks in [KLLN16, BBC$^+$21, BLNS21], that essentially break EUF-qCMA security of the OCB modes using Simon's [Sim97] algorithm and Deutsch's algorithm [Deu85], in order to study the modes' IND-qCPA security as well. Simon's algorithm is a quantum period finding algorithm which is a frequently used tool when it comes to analyzing block cipher modes in a quantum setting. On the other hand, Deutsch's algorithm is able to solve a certain black box problem with a single query, whereas a classical computer would need two. The authors of [MMPR22] came to the

[1] OCB2 has been broken classically by [IIMP19] and OCB2f is a patched version proposed by the authors.

conclusion that despite all of the OCB modes being classically IND-CPA secure, these modes are (apart from OCB2(f) with "empty" associated data) in fact not IND-qCPA secure by applying the above mentioned quantum algorithms. More surprisingly they were able to proove that OCB2 is IND-qCPA secure when it is used in the restricted setting of empty associated data.

Another mode of operation is the *Offset Two-Round* (OTR) mode instantiated with *AES* as its underlying block cipher (AES-OTR) [Min16]. AES-OTR was a third-round candidate of the CAESAR competition [CAE19] that aimed to devise a portfolio of secure authenticated encryption schemes that are suitable for widespread application. It is a nonce-based authenticated encryption with associated data (AEAD) scheme that offers two different ways for associated data processing. Interestingly, this mode is closely related to the OCB modes as it shares similar structures. In the classical setting it has been shown to offer provable security guarantees [Min16] under the assumption that AES is a pseudorandom function. Naturally, the question arises if this security can be confirmed in a quantum setting. However, recent work in [LCP22] proposed an approach to attack the EUF-qCMA security that is also based on the quantum superposition model and uses Simon's algorithm, indicating that AES-OTR is not quantum secure. Having said that, confidentiality of AES-OTR has not been addresses at all. This leads us to pose the question:

Is the AES-OTR mode IND-qCPA secure?

Of course AES-OTR being vulnerable to forgery attacks rules out the mode being used as a secure AE scheme in a quantum setting. Nevertheless, analyzing IND-qCPA security is still a question of interest, as it gives further insight to the mode. In an attempt to fix the mode it then either requires minimal changes to fix the issues regarding authenticity, if the mode already guarantees confidentiality, or it needs a more drastic update to salvage the mode as a whole when there are attacks on confidentiality.

There exist even more block cipher modes that are closely related to the OCB modes including the *Offset Public Permutation* (OPP) mode. OPP essentially tries to generalize OCB3 by replacing the underlying block cipher by an efficiently invertible public permutation and a different form of masking. It ensures fast encryption and full parallelization. Therefore, it is of integral interest to study the quantum security of this mode as well. However, none of the aforementioned works analyze OPP as a mode at all, which motivates us to ask the following question:

Is the OPP mode quantum secure?

1.1 Our Contributions

In this thesis, we make contributions to answer the above formulated questions concerning the quantum security of both AES-OTR and OPP modes. We do this by presenting attacks for both modes and thus make significant progress in filling the gap of analyzing the quantum security of important OCB-like modes of operation in the quantum superposition model. We point out that our work is the first to look into the quantum security of OPP and the first attempt at breaking the confidentiality of AES-OTR as these points of interest have not been addressed in any prior work to the best of our knowledge. Our results are listed below:

AES-OTR is NOT IND-qCPA secure. In Chapter 3 we present the first IND-qCPA attacks against AES-OTR. We first consider the setting of a weak adversary where the nonces used by the challenger to answer encryption queries in the IND-qCPA security game are generated uniformly at random and the adversary has no influence in choosing them. We adapted this setting from the analysis on OCB in [MMPR22]. We consider AES-OTR with both parallel and serial processing of associated data and exploit exactly the way AES-OTR authenticates AD.

On a high level, the attack breaking IND-qCPA security for parallel AD processing uses Simon's algorithm as well as Deutsch's algorithm to gain raw block cipher access. Having raw block cipher access, we are able to compute the encryption of any input message with respect to the underlying block cipher without knowing the secret key. Using this we are able to formulate our IND-qCPA attack. This setting is similar to the attack strategies in [MMPR22] that break confidentiality of the OCB modes. However, we are able to pinpoint an issue that arises in their attack on OCB2 with AD that has not been addressed properly. This also concerns AES-OTR. To be precise, we have to take into account whether the tag produced by AES-OTR encryption is truncated or not. Our work treats both cases accordingly and upon contacting the authors of [MMPR22] regarding this issue, they confirmed that they implicitly assumed to work with untruncated tags. If the tags are truncated, we have to distinguish two cases: if only a constant amount of bits are truncated out, then the same attack still applies when we perform an additional step of guessing the missing bits. If it is not constant however, this approach does not work as it is inefficient.

We highlight, that our attack breaking IND-qCPA security of AES-OTR for serial processing of AD is different from the ones in [MMPR22] and our attack where AD is processed in parallel. Here, we immediately gain raw block cipher access applying only Simon's algorithm and do *not* need Deutsch's algorithm to achieve this. Our attack is therefore more powerful, as it needs less steps to break the same security notion. To the best of our knowledge, *no* prior application of Simon's algorithm in the quantum crypt-

analysis literature was able to achieve this strong notion of raw block cipher access with respect to the considered cryptosystem.

To complete the analysis of AES-OTR, we consider a setting where the scheme is used as a pure AE scheme, i.e. the AD is always assumed to be empty. For our attack to succeed, we have to assume a setting in which the adversary is allowed to choose the classical nonces adaptively and hand them to the challenger. The challenger then has to respond to encryption queries in the IND-qCPA security game using these nonces. This setting is the same as considered in [MMPR22, Section 4.4] with respect to their attack against OCB2 as a "pure" AE scheme. What makes our attack a non-trivial extension to the above attack on OCB2, is that for AES-OTR there is an additional formatting function applied to the nonce before it is AES-encrypted. We thus have to perform an additional step that increases the overall complexity of our attack.

Refined Attacks on EUF-qCMA Security of AES-OTR. As a side contribution of this thesis, we refine the attacks on EUF-qCMA Security of AES-OTR in [LCP22] when associated data is processed in both parallel and serial. These attacks use the same properties of the associated data and utilize the same methods as in our IND-qCPA attacks and therefore follow in a straight forward way.

A Quantum Key-Recovery Attack on OPP. We present the first quantum key-recovery attack on OPP in Chapter 4. This mode is therefore completely broken in the quantum superposition model. Our attack is conducted in the weakest setting similar to our attacks on AES-OTR, where the nonces are chosen uniformly at random by the challenger. In contrast to AES-OTR being based on a block cipher that may only be inverted knowing the key, OPP is built upon an efficiently invertible public permutation. We are able to show that this property actually is the key issue of OPP and allows us to formulate the key recovery attack. On a high level, we are able to recover the permutation of a value that incorporates the key by using only a *single* quantum encryption query in an application of Simon's algorithm. Hence, we gain knowledge of the key by simply inverting the permutation. Even if OPP is an AEAD scheme, our attack is focused solely on the encryption part, meaning that OPP is used as a pure AE scheme. This is in contrast to our attacks on AES-OTR in the same adverserial setting, where we exploit the way AD is processed.

Applicability to related modes. When choosing the modes of operation to analyze in this thesis we also came across OCB-hc and OTR-hc (*h*alf-*c*hecksum) [IM19]. There, the half-checksum method is introduced, which sets the checksum to be the xor of the $n/2$ most significant bit of the plaintext blocks instead of the checksum being the xor of all plaintext blocks. Further, they use masking which is dependent on the nonce and the tag is

truncated by $n/2$ bits, which makes it only half of the original tag size. The modes therefore share a lot of similar structures with the OCB and AES-OTR modes and it is a valid hypothesis that variations or combinations of attacks in [KLLN16, BBC$^+$21, BLNS21, MMPR22] breaking unforgeability and confidentiality of the OCB modes, as well as attacks presented in this thesis may be applicable in a similar fashion.

Chapter 2

Preliminaries

2.1 Notation

Denote by $\{0,1\}^*$ the set of all finite-length bit strings and $\{0,1\}^{8*}$ the set of all finite-length byte strings. We let the parameters n, k, τ, $\kappa \geq 0$ define the block length, the size of the key, tag, and nonce respectively. For $b \in \mathbb{N}$ we let $[b] := \{1, ..., b\}$. Given $x, y \in \{0,1\}^*$, the concatenation of x and y is denoted as $x||y$. We let the length of x in bits be denoted as $|x|$ and we define $|x|_b := \max\{1, \lceil X/b \rceil\}$. We use the symbols $\oplus, \ll, \gg, \lll, \ggg$ to denote bit-wise XOR, left-shift, right-shift, left-rotation and right-rotation, respectively.

Further, we define the following padding function that, for a given input X, extends it to a desired length m

$$\text{pad}_m^0 : \{0,1\}^{\leq m} \to \{0,1\}^m, X \mapsto X||0^{m-|X|}$$

and for $0 \leq |X| < m$, we write $\underline{X} = X||10^{m-|X|-1}$ as the 10^* padding.

By $\text{msb}_l(x)$ we mean the sequence of first l left-most bits of the bit sting x and for any non-negative integer q, let $\text{bin}(q, m)$ denote the standard m-bit encoding of q.

We want to highlight the difference in notation for the encryption functions used for AES-OTR in Chapter 3 and for OPP in Chapter 4. By OTR-$\mathcal{E}_{K,\cdot}(\cdot)$ we indicate the encryption algorithm of AES-OTR with an underlying block cipher (AES to be precise) encryption function E_K with key K. On the other hand we use OPP-$\mathcal{E}(K, \cdot)$ to denote the encryption algorithm of OPP with key K that uses a public permutation instead of a block cipher.

In the context of AES-OTR we use notations such as $2X, 3X$ or $7X$ for an n-bit string X. Following [Min16], we here interpret X as a coefficient vector of the polynomial in $\text{GF}(2^n)$. So by $2X$ we essentially mean multiplying

the generator of the field $\mathrm{GF}(2^n)$, which is the polynomial x, and X over $\mathrm{GF}(2^n)$. This process is referred to as *doubling*. Similarly, $2^i X$ denotes i-times doubling X and we denote $3X = X \oplus 2X$ as well as $7X = 2^2X \oplus 2X \oplus X$. Field multiplication over $\mathrm{GF}(2^n)$ for $n = 128$ can be implemented as

$$2X = \begin{cases} X \ll 1 & \text{if } \mathrm{msb}_1 X = 0. \\ (X \ll 1) \oplus 0^{120}10000111 & \text{if } \mathrm{msb}_1 X = 1. \end{cases}$$

We omit the details here and refer to [Min16] for further details.

2.2 Quantum Algorithms

In this section we present two algorithms we will use in Chapter 3 and 4 to formulate our attacks. More specifically, we use Simon's and Deutsch's algorithm, which are quantum algorithms, that enable us to recover some values that are important in order for our attacks to succeed. We adapt the specification of Simon's algorithm from [KLLN16] and Deutsch's algorithm from [MMPR22].

2.2.1 Simon's Algorithm

Simon's algorithm is a quantum algorithm that is able to solve the following problem referred to as *Simon's problem*. This algorithm is the key element of most of our attacks.

Definition 2.1 *(Simon's Problem) Given quantum access to a Boolean function $f : \{0,1\}^n \to \{0,1\}^n$ (called Simon's function) for which it holds: $\exists s \in \{0,1\}^n$: $\forall x, y \in \{0,1\}^n$*

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

the goal is to find the period s of f.

This problem of course can be solved in a classical setting by searching for collisions as well. It is known that the time needed to solve this problem is $\Theta(2^{n/2})$, when we are given classical access to the function f. We assume here, that access to an input function is made by querying it. (A classical query oracle is given by a function $x \mapsto f(x)$.) However, when we are able to query the function f quantum-mechanically, and we are thus allowed to make queries of arbitrary quantum superpositions of the form $|x\rangle|0\rangle \mapsto |x\rangle|f(x)\rangle$, Simon's algorithm can solve this problem with query complexity $\mathcal{O}(n)$.

On a high level, Simon's algorithm is able to recover a random vector $y \in \{0,1\}^n$ in a *single* quantum query to f that is orthogonal to the period s, i.e. $y \cdot s = 0$. This subroutine is repeated $\mathcal{O}(n)$ times such that one obtains $n-1$ independent vectors where each is orthogonal to s with high probability.

Therefore s can be recovered by solving the corresponding system of linear equations using basic linear algebra using $\mathcal{O}(n)$ quantum queries to the function f. For more detail on the subroutine, we refer to [KLLN16].

We want to highlight that Simon's algorithm is able to recover a vector that is orthogonal to s in a *single* quantum query to f. This is important for our attacks in Sections 3.5 and 4.2. There, Simon's function does not only take a single input $x \in \{0,1\}^n$, but n inputs $x_i \in \{0,1\}^n$ for $i \in [n]$. If now f is periodic with period s_i in each of the inputs x_i, f has period $(s_1, ..., s_n)$. The vector $y = (y_1, ..., y_n) \in \{0,1\}^{n^2}$, $y_i \in \{0,1\}^n$, recovered in one application of the subroutine in Simon's algorithm, is now orthogonal to *each* of the periods s_i, i.e. $y_i \cdot s_i = 0 \; \forall i \in [n]$.

Unwanted Collisions. It is possible, that apart from the period s of f that exists by construction of f, there might me numerous more collisions than those of the form $f(x) = f(x \oplus s)$. We refer to those as *"unwanted periods"*. Note, that if this number is too large, then Simon's algorithm might not be able to always recover the right vectors y and the system of linear equations might not be of full rank after $\mathcal{O}(n)$ queries. However, [KLLN16] showed that even if f admits some unwanted periods of the form $f(x) = f(x \oplus t)$ for some $t \notin \{0, s\}$, Simon's algorithm is still able to recover s as long as

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

As we will show, this condition is always satisfied for our attacks.

2.2.2 Deutsch's Algorithm

Deutsch's algorithm is a tool to solve the following problem.

Definition 2.2 *Given quantum access to a Boolean function $f : \{0,1\} \to \{0,1\}$, the goal is to decide whether f is constant, i.e. $f(0) = f(1)$, or f is balanced, i.e. $f(0) \neq f(1)$.*

The algorithm can solve this problem with a *single* quantum query to f with success probability 1. To be precise, Deutsch's algorithm is able to compute

$$f(0) \oplus f(1) = \begin{cases} 0 & \text{if } f \text{ is constant.} \\ 1 & \text{if } f \text{ is balanced.} \end{cases}$$

using a single quantum query to f. Note, that any algorithm with classical access to f would need two queries to decide this problem with the same success probability.

2.3 Definitions

In this section we present quantum security definitions used in the following chapters as well as the definitions for some cclassical primitives we will encounter. We adapt the definitions as presented in [MMPR22].

Our analysis on AES-OTR will mainly focus on breaking IND-qCPA security defined below, as [LCP22] already claims to break existential unforgeability. However, we will refine their attacks in Subsection 3.6 by extending on our IND-qCPA attacks.

Note, that we are able to formulate a key recovery attack for OPP. As key recovery is usually quite severe, we will discuss the implications this has on all of the security notions below.

We begin by defining a *nonce-based authenticated encryption with associated data (AEAD) scheme* as both AES-OTR and OPP are one.

Definition 2.3 *(Nonce-based AEAD scheme) A nonce-based AEAD scheme is a tuple of probabilistic polynomial-time algorithms* $\Pi = (Enc, Dec)$ *with a key space* $\mathcal{K} = \{0,1\}^k$ *such that*

Enc(K, N, A, M)**:** *This algorithm takes as input a key K, a nonce N, associated data (AD) A and a message M. It produces as an output a ciphertext C and a tag T. We write* $(C, T) \leftarrow Enc_K(N, A, M)$.

Dec(K, N, A, C, T)**:** *This algorithm takes as input a key K, a nonce N, AD A, a ciphertext C and a tag T and outputs a message M or* $\perp$*. We write* $Dec_K(N, A, C, T)$ *to denote this output.*

The AEAD scheme needs to satisfy correctness, which is given, if for any N, A *and* M *we have*

$$Pr[Dec_K(N, A, Enc_K(N, A, M)) = M] \geq 1 - \epsilon,$$

$\forall K \in \mathcal{K}$ *and* $\epsilon > 0$ *is negligible.*

Below, we define three quantum security notions for an AEAD scheme.

Definition 2.4 *(IND-qCPA with random nonces) A nonce-based AEAD scheme* $\Pi = (Enc, Dec)$ *is indistinguishable under quantum chosen-plaintext attack (IND-qCPA secure) with random nonces, if there is no efficient quantum adversary* $\mathcal{A}$ *that is able to win the following security game, except with probability at most* $\frac{1}{2} + \epsilon$ *where* $\epsilon > 0$ *is negligible.*

Key generation: *A random key $K \leftarrow \mathcal{K}$ and a random bit $b \leftarrow \{0,1\}$ are chosen by the challenger.*

Queries: *In any order the adversary $\mathcal{A}$ is allowed to make two types of queries:*

- **Encryption queries:** *The challenger first randomly chooses a nonce $N \leftarrow \{0,1\}^\kappa$ and forwards it to $\mathcal{A}$. The adversary now can choose a message-AD pair (M, A), possibly in superposition, and the challenger encrypts (N, A, M) with the classical nonce N and returns the output (C, T) to $\mathcal{A}$.*
- **Challenge query:** *The challenger picks a random nonce $N \leftarrow \{0,1\}^\kappa$ once more and gives it to the adversary. Afterwards, $\mathcal{A}$ chooses two same sized classical message-AD pairs $(M_0, A), (M_1, A)$ and forwards them to the challenger which in turn encrypts (N, A, M_b) with the previously chosen classical nonce N. The output (C^*, T^*) is again given to $\mathcal{A}$.*

Guess: *The adversary outputs a bit b' and wins if $b = b'$.*

Let p be the probability that $\mathcal{A}$ wins the above game. Then its IND-qCPA advantage with respect to the AEAD scheme Π is given by $Adv_\Pi^{IND\text{-}qCPA}(\mathcal{A}) = \left|p - \frac{1}{2}\right|$. So Π is said to be IND-qCPA secure under randomly chosen nonces if $Adv_\Pi^{IND\text{-}qCPA}(\mathcal{A})$ of any polynomial-time quantum adversary $\mathcal{A}$ is negligible.

Definition 2.5 *(EUF-qCMA with random nonces) A nonce-based AEAD scheme $\Pi = (Enc, Dec)$ is existentially unforgeable under quantum chosen message attack (EUF-qCMA secure) under randomly chosen nonces, if there is no efficient quantum adversary $\mathcal{A}$ that is able to win the following security game, except with negligible probability.*

Key Generation: *A random key $K \leftarrow \mathcal{K}$.*

Queries: *The adversary $\mathcal{A}$ is allowed to make encryption queries of the following form:*

- **Encryption queries:** *The challenger first randomly chooses a nonce $N \leftarrow \{0,1\}^\kappa$ and gives it to $\mathcal{A}$. Now, the adversary chooses a message-AD pair (M, A), possibly in superposition, and the challenger encrypts (N, A, M) with the previously chosen classical nonce N and returns the output (C, T) to $\mathcal{A}$.*

Forgeries: *Let q be the number of encryption queries the adversary made. Now $\mathcal{A}$ produces $q+1$ classical tuples (N, A, C, T) with any nonces N of its own choice. The adversary wins, if for each tuple it holds $Dec_K(N, A, C, T) \neq \perp$.*

Definition 2.6 *(UUF-qCMA with random nonces) A nonce-based AEAD scheme* $\Pi = (Enc, Dec)$ *is universally unforgeable under quantum chosen message attack (UUF-qCMA secure) under randomly chosen nonces, if there is no efficient quantum adversary* $\mathcal{A}$ *that is able to win the following security game, except with negligible probability.*

Key Generation: *A random key* $K \leftarrow \mathcal{K}$.

Challenge: *First, the challenger randomly picks a nonce* $N^* \leftarrow \{0,1\}^\kappa$ *and a message-AD pair* (M^*, A^*) *from its corresponding spaces.* $\mathcal{A}$ *then gets forwarded the tuple* (N^*, A^*, M^*) *from the challenger.*

Queries: *The adversary* $\mathcal{A}$ *is allowed to make encryption queries of the following form:*

- **Encryption queries:** *The challenger randomly chooses a nonce* $N \leftarrow \{0,1\}^\kappa$ *and gives it to* $\mathcal{A}$. *Now, the adversary chooses a message-AD pair* (M, A), *possibly in superposition, and the challenger encrypts* (N, A, M) *with the previously chosen classical nonce* N *and returns the output* (C, T) *to* $\mathcal{A}$.

Forgeries: $\mathcal{A}$ *outputs a classical tuple* (C^*, T^*) *after making a polynomial number of encryption queries. The adversary wins if* $Dec_K(N^*, A^*, C^*, T^*) = M^*$.

It is important to note, that regarding the definition of IND-qCPA security we are only allowed to make quantum queries to the encryption oracle for encryption queries. The challenge queries need to be classical values. However, we are allowed to make these two types of queries in any order and are allowed to output our guess bit b' at any point.

The difference between EUF-qCMA security and UUF-qCMA security essentially is that for the former the adversary has to output one forgery more than the amount of encryption queries it made and in the latter it has to produce a specific forgery corresponding to a given set of nonce, AD and message.

Chapter 3

Quantum Cracks in the AES-OTR Encryption Armor

The *AES-OTR* block cipher mode emerged from the *Offset Two-Round* (OTR) mode [Min14] as a part of the CAESAR competition [CAE19] and is based on the AES block cipher as proposed in [Min16]. It is a nonce based authenticated encryption with associated data (AEAD) scheme and provides two methods for associated data processing. AES-OTR has a provable security in the classical setting under the assumption that AES is a pseudorandom function as argued in [Min16].

However, in this chapter we will show that we can exploit the way AD is processed in both cases, namely in parallel and serial, to break IND-qCPA security. In this case, we assume a setting where the adversary has quantum access to an encryption oracle and the nonces the challenger uses to answer encryption queries are picked uniformly at random. We even go one step further and break IND-qCPA security of AES-OTR considered as a pure AE scheme, i.e., with empty AD. To do so, we consider a stronger adversarial setting in which the adversary is allowed to pick the classical nonces adaptively. We will be extending upon techniques as utilized in [MMPR22].

3.1 Specifications of AES-OTR

We begin by describing the AES-OTR mode by following the specifications as proposed in [Min16] for the third round of the CAESAR competition. Let n, k, τ, κ as labeled in Chapter 2, where $k \in \{128, 192, 256\}$, $\tau \in \{32, 40, ..., 128\}$ and $\kappa \in \{8, 16, ..., 120\}$ are of a fixed length. Since AES-OTR uses AES as its underlying block cipher, $n = 128$ is fixed as well and we assume E_K to denote the AES encryption function with key K. Also, the lengths of both a plaintext M and associated data A are required to fulfill $|M|, |A| \in \{0,1\}^{8*}$ such that $|M|_8, |A|_8 \leq 2^{64}$. We note that [Min16] provides

sets of recommended parameters which imply that for both instantiations of AES-OTR either with AES-128 or AES-256 a 16-byte tag should be used. This recommendation becomes relevant for our attack in Section 3.3. For further details on the parameters we refer to [Min16].

Below, we provide a simplified description of AES-OTR in algorithmic and pictorial description for both variants of processing AD, namely in parallel (on the left) and in serial (on the right). To indicate how the AD is processed, we use p for parallel and s for serial processing and write OTR-$\mathcal{E}_{K,p}(N, A, M)$ or OTR-$\mathcal{E}_{K,s}(N, A, M)$ respectively. We omit the description of the decryption algorithm, as decryption is not relevant for our attacks. We again refer to [Min16] for the details. To be more precise, Algorithm 3 corresponds to the encryption part and Algorithms 5 and 6 describe the authentication part of the AEAD scheme described in Algorithm 1 and 2 for parallel and serial AD processing respectively. Note, that the encryption core of AES-OTR with parallel (normal box) and serial (dashed box) AD processing only differ in the way U is defined. Algorithm 4 outlines how the nonce N is formatted before being incorporated into the mask U, used to encrypt the plaintext. This formatting will play an important role for our attack in Section 3.5.

We provide a brief explanation of the encryption Algorithm 3. The plaintext M is first decomposed in blocks of n bits $M_1||...||M_m \leftarrow M$ with a possibly partial last block M_m. Two consecutive plaintext blocks M_{2i-1}, M_{2i} for $i \leq \lceil m/2 \rceil - 1$ are encrypted by a two-round Feistel permutation with masks as

$$C_{2i-1} = E_K(2^{i-1}U \oplus M_{2i-1}) \oplus M_{2i}$$
$$C_{2i} = E_K(2^{i-1}3U \oplus C_{2i-1}) \oplus M_{2i-1}$$

with U being an encrypted nonce that depends on the way AD is processed (see Algorithm 3, Lines 2&3). The last plaintext block(s) is/are treated differently depending on m. If m is even, a variation of the above encryption is applied and otherwise a variation of CTR mode is applied. In particular, for a single block message AES-OTR only encrypts it by xor-ing it with some value depending on U. We will exploit this property for our IND-qCPA attacks. Following this, the unauthenticated ciphertext is obtained.

The tag is computed by AES-encrypting the checksum of even plaintext blocks (including a part of the last block) xor-ed with some mask yielding a value TE. In the case of parallel AD processing, the tag is the truncation of the xor of TE and TA, where TA is obtained by Algorithm 5. For AD in serial, TE is already dependent on TA obtained in Algorithm 6 and only needs to be truncated to the tag length. When the AD is empty then TA is set to be 0^n in both serial and parallel AD processing versions of AES-OTR.

We discuss the way associated data is processed to obtain the value TA in Algorithms 5 and 6 in more detail in Sections 3.2.1 and 3.2.2 respectively.

Algorithm 1 OTR-$\mathcal{E}_{K,p}(N, A, M)$

$(C, TE) \leftarrow$ EF-P$_{K,\tau}(N, M)$
if $A \neq \varepsilon$ **then**
    $TA \leftarrow$ AF-P$_K(A)$
else $TA \leftarrow 0^n$
$T \leftarrow \text{msb}_\tau(TE \oplus TA)$
return (C, T)

Algorithm 2 OTR-$\mathcal{E}_{K,s}(N, A, M)$

if $A \neq \varepsilon$ **then**
    $TA \leftarrow$ AF-S$_K(A)$
else $TA \leftarrow 0^n$
$(C, TE) \leftarrow$ EF-S$_{K,\tau}(N, M, TA)$
$T \leftarrow \text{msb}_\tau(TE)$
return (C, T)

Algorithm 3 EF-P$_{K,\tau}(N, M)$, EF-S$_{K,\tau}(N, M, TA)$

$\Sigma \leftarrow 0^n$
$U \leftarrow E_K(\text{Format}(\tau, N))$
$U \leftarrow 2\big(E_K(\text{Format}(\tau, N)) \oplus TA\big)$
$L \leftarrow U$, $L^\# \leftarrow 3U$
$M_1||...||M_m \leftarrow M$ s.t. $|M_i| = n$
for $i \in \{1, ..., \lceil m/2 \rceil - 1\}$ **do**
    $C_{2i-1} \leftarrow E_K(L \oplus M_{2i-1}) \oplus M_{2i}$
    $C_{2i} = E_K(L^\# \oplus C_{2i-1}) \oplus M_{2i-1}$
    $\Sigma \leftarrow \Sigma \oplus M_{2i}$
    $L \leftarrow L \oplus L^\#$, $L^\# \leftarrow 2L^\#$ ▷ $L = 2^i U$, $L^\# = 2^i 3U$
if m is even **then**
    $Z \leftarrow E_K(L \oplus M_{m-1})$
    $C_m \leftarrow \text{msb}_{|M_m|}(Z) \oplus M_m$
    $C_{m-1} \leftarrow E_K(L^\# \oplus \underline{C_m}) \oplus M_{m-1}$
    $\Sigma \leftarrow \Sigma \oplus Z \oplus \underline{C_m}$
    $L^* \leftarrow L^\#$
else if m is odd **then**
    $C_m \leftarrow \text{msb}_{|M_m|}(E_K(L)) \oplus M_m$
    $\Sigma \leftarrow \Sigma \oplus \underline{M_m}$
    $L^* \leftarrow L$
if $|M_m| \neq n$ **then** $TE \leftarrow E_K(3^2 L^* \oplus \Sigma)$
else $TE \leftarrow E_K(7L^* \oplus \Sigma)$
$C \leftarrow C_1||...||C_m$
return (C, TE)

Algorithm 4 Format(τ, N)

return $\text{bin}(\tau \bmod n, 7)||0^{n-8-\kappa}||1||N$

Algorithm 5 AF-P$_K(A)$

$\Xi \leftarrow 0^n$
$Q \leftarrow E_K(0^n)$
$A_1||...||A_a \leftarrow A$, s.t. $|A_i| = n$
for $i \in \{1, ..., a-1\}$ **do**
    $\Xi \leftarrow \Xi \oplus E_K(Q \oplus A_i)$
    $Q \leftarrow 2Q$
$\Xi \leftarrow \Xi \oplus \underline{A_a}$
if $|A_a| \neq n$ **then**
    $TA \leftarrow E_K(3Q \oplus \Xi)$
else $TA \leftarrow E_K(3^2Q \oplus \Xi)$
return TA

Algorithm 6 AF-S$_K(A)$

$\Xi \leftarrow 0^n$
$Q \leftarrow E_K(0^n)$
$A_1||...||A_a \leftarrow A$, s.t. $|A_i| = n$
for $i \in \{1, ..., a-1\}$ **do**
    $\Xi \leftarrow E_K(A_i \oplus \Xi)$
$\Xi \leftarrow \Xi \oplus \underline{A_a}$
if $|A_a| \neq n$ **then**
    $TA \leftarrow E_K(2Q \oplus \Xi)$
else
    $TA \leftarrow E_K(4Q \oplus \Xi)$
return TA

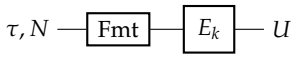

Figure 3.1: Computation of the U value in the case of parallel AD processing

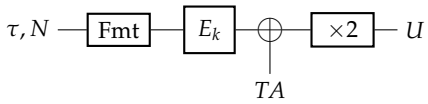

Figure 3.2: Computation of the U value in the case of serial AD processing

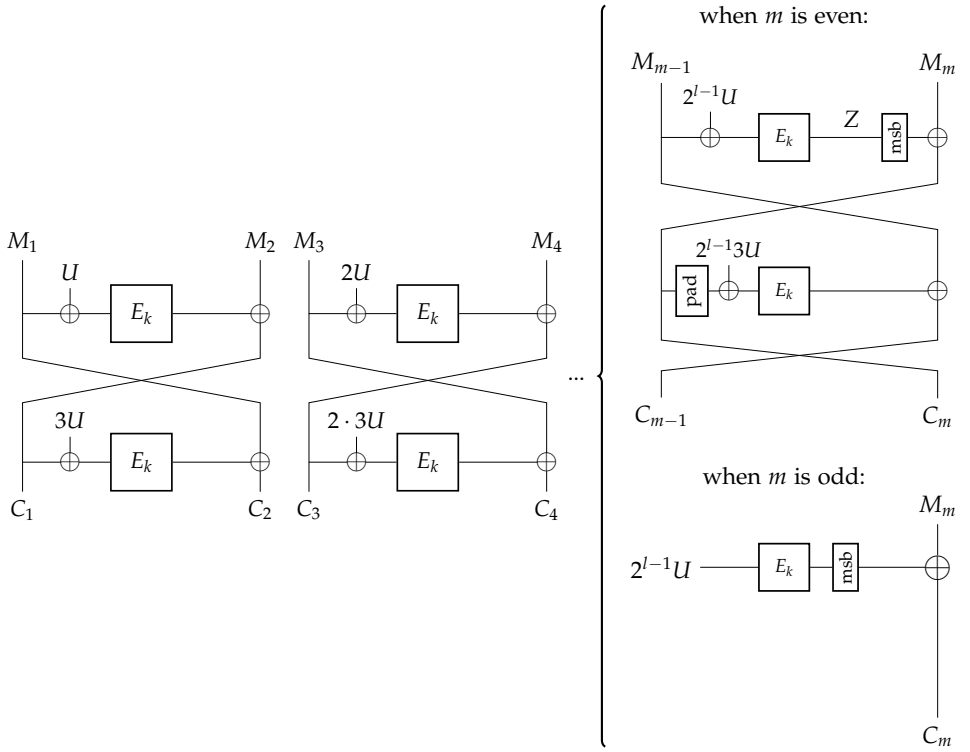

Figure 3.3: Encryption core of AES-OTR. Note, that this is the same for parallel and serial AD processing. Only the value U is different. Also, we have $l = \lceil m/2 \rceil$.

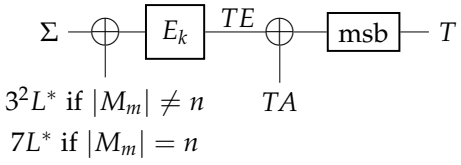

Figure 3.4: Computation of the tag T in the case of parallel AD processing.

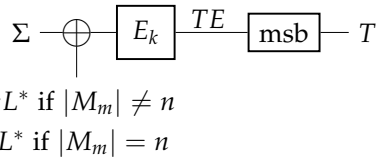

Figure 3.5: Computation of the tag T in the case of serial AD processing.

Figure 3.6: Here, $\Sigma = M_2 \oplus M_4 \oplus ... \oplus M_{m-2} \oplus Z \oplus \underline{C_m}$ and $L^* = 2^{l-1}3U$ when m is even and $\Sigma = M_2 \oplus M_4 \oplus ... \oplus M_{m-1} \oplus \underline{M_m}$ and $L^* = 2^{l-1}U$ when m is odd. In both cases $l = \lceil m/2 \rceil$. Note, that M_m and C_m possibly are only partial block and thus are padded with the 10^* padding.

In the pictorial description of how the value TA in Algorithms 5 and 6 for parallel and serial AD processing is computed, the value Q is used as part of the masking and is defined as $Q = E_K(0^n)$. Note, that Q is a constant value and is independent of the nonce N. This is the key observation we use in our attacks in Sections 3.3 and 3.4 that exploit the way AD is being processed.

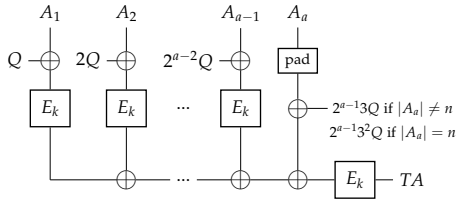

Figure 3.7: Computation of the value TA of the authentication core of AES-OTR with parallel AD processing with $Q = E_K(0^n)$.

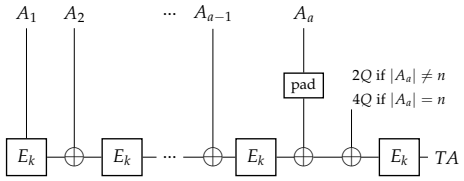

Figure 3.8: Computation of the value TA of the authentication core of AES-OTR with serial AD processing with $Q = E_K(0^n)$.

3.2 Prior Quantum Attacks on AES-OTR

Prior work in [LCP22] already proposed an approach to attack the unforgeability of AES-OTR which uses quantum access to the encryption oracle and applies Simon's algorithm. The way associated data is processed and its impact on the tag T (more specifically on the intermediate variable TA) is exploited. We will use the same properties of TA to formulate our attacks in order to break confidentiality. We here give a short description on said properties and refer to [LCP22] for specific details.

On a high level, the processing of associated data, whether in parallel or serial, enables the derivation of collisions for the intermediate variable TA, which is utilized to compute the tag T. For convenience, we assume that the last block of AD is always of full size to avoid having to treat it differently in the analysis as padding would be applied otherwise. Additionally, for

the sake of convenience we introduce a subscript to the value TA in order to keep track to which set of AD it belongs. E.g. for a set of associated data A we set $TA_A = \text{AF-P}_K(A)$.

3.2.1 Finding Collisions when Associated Data Processed in Parallel

We first consider the case when associated data is processed in parallel as described in Algorithm 5 with output TA. Given a set of associated data $A = A_1||A_2||...||A_a$, where $A_i \in \{0,1\}^n \ \forall i \in [a]$, we define a set of intermediate variables $X_A[i]$ for $i \in [a-1]$ as

$$X_A[i] = E_K(A_i \oplus 2^{i-1}Q)$$

such that

$$TA_A = E_K(X_A[1] \oplus ... \oplus X_A[a-1] \oplus A_a \oplus 2^{a-1}3^2Q).$$

Note, that even though the value Q is constantly being updated in Algorithm 5, we here consider Q to be a fixed value and set it to $Q = E_K(0^n)$.

In Figure 3.9 below, the values corresponding to $X[i]$ are illustrated.

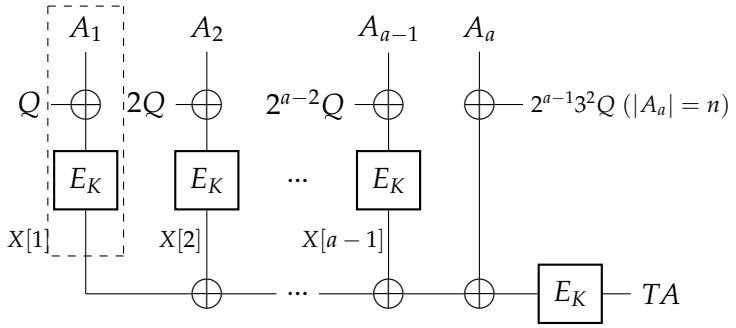

Figure 3.9: Illustration of the newly defined variable $X[i]$ used to construct collision for parallel processing of AD. In contrast to Figure 3.7, there is no padding on the last AD block since it is considered to be of size n.

For the given associated data A we can construct a second set of associated data $B = B_1||...||B_a$, $A \neq B$ which yields the same value TA after being processed in parallel as follows: Define

$$\begin{aligned} B_1 &= A_2 \oplus 3Q \\ B_2 &= A_1 \oplus 3Q \\ B_i &= A_i \end{aligned}$$

for $i \in \{3, ..., a\}$. Note, that $3Q = Q \oplus 2Q$ by definition (see Section 2.1). This implies

$$\begin{aligned} X_B[1] &= E_K(B_1 \oplus Q) = E_K(A_2 \oplus 3Q \oplus Q) = E_K(A_2 \oplus 2Q) \\ &= X_A[2] \\ X_B[2] &= E_K(B_2 \oplus 2Q) = E_K(A_1 \oplus 3Q \oplus 2Q) = E_K(A_1 \oplus Q) \\ &= X_A[1] \\ X_B[i] &= X_A[i] \end{aligned}$$

for $i \in \{3, ..., a\}$ and therefore

$$\begin{aligned} TA_B &= E_K(X_B[1] \oplus X_B[2] \oplus X_B[3] \oplus ... \oplus X_B[a-1] \oplus B_a \oplus 2^{a-1}3^2Q) \\ &= E_K(X_A[2] \oplus X_A[1] \oplus X_A[3] \oplus ... \oplus X_A[a-1] \oplus A_a \oplus 2^{a-1}3^2Q) \\ &= TA_A \end{aligned}$$

More general, for given AD A like before, one can define for arbitrary $p, q \in [a]$, $p \neq q$ and $i \in [n]$, $i \neq p, q$ a new set of associated data $B' = B_1||...||B_a$ as

$$\begin{aligned} B'_q &= A_p \oplus 2^{p-1}Q \oplus 2^{q-1}Q \\ B'_p &= A_q \oplus 2^{p-1}Q \oplus 2^{q-1}Q \\ B'_i &= A_i. \end{aligned} \tag{3.2.1}$$

It is not hard to see that with the same argument as before (which is the case for $p = 1$ and $q = 2$), that $TA_{B'} = TA_A$.

3.2.2 Finding Collisions when Associated Data Processed in Serial

We now consider the case when associated data is processed in serial as described in Algorithm 6 with output TA. For a given set of AD $A = A_1||...||A_a$ with $A_i \in \{0,1\}^n$, we want to produce a second set of AD $B = B_1||...||B_{a-1}$, where $A \neq B$, such that $TA_A = TA_B$.

Again, we define some intermediate variables $X_A[i]$ for $i \in [a-1]$ as:

$$\begin{aligned} X_A[1] &= A_1 \\ X_A[j] &= A_j \oplus E_K\big(X_A[j-1]\big) \\ X_A[a] &= 4Q \oplus A_a \oplus E_K\big(X_A[a-1]\big) \end{aligned}$$

for $j \in \{2, ..., a-1\}$ and $Q = E_K(0^n)$ such that

$$TA_A = E_K\big(X_A[a]\big).$$

In Figure 3.10 we demonstrate the definition of $X[i]$.

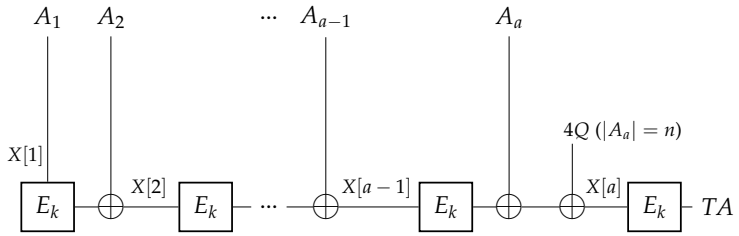

Figure 3.10: Illustration of serial processing of associated data with indication of the variable $X[i]$. Again, there is no padding applied since $|A_a| = n$.

We construct a second set of associated data $B = B_1||...||B_{a-1}$, where

$$\begin{aligned} B_1 &= A_2 \oplus E_K(A_1) \\ B_i &= A_{i+1} \end{aligned}$$

for $i \in \{2, ..., a-1\}$. We claim that this causes the TA values to coincide. We want to show that $X_A[j] = X_B[j-1] \forall j \in [a]$ in a proof by induction on $2 \leq j \leq a-1$ and first observe that

$$X_A[2] = A_2 \oplus E_K\big(X_A[1]\big) = A_2 \oplus E_K(A_1) = X_B[1].$$

We now assume $X_A[j] = X_B[j-1]$ as the induction hypothesis. It holds

$$\begin{aligned} X_A[j+1] &= A_{j+1} \oplus E_K\big(X_A[j]\big) = B_j \oplus E_K\big(X_B[j-1]\big) \\ &= X_B[j]. \end{aligned}$$

Here the second equality uses the induction hypothesis and the definition of B. Thus, we may conclude by induction that

$$\begin{aligned} X_B[a-1] &= 4Q \oplus B_{a-1} \oplus E_K\big(X_B[a-2]\big) \\ &= 4Q \oplus A_a \oplus E_K\big(X_A[a-1]\big) \\ &= X_A[a]. \end{aligned}$$

And therefore we get

$$TA_B = E_K\big(X_B[a-1]\big) = E_K\big(X_A[a]\big) = TA_A. \tag{3.2.2}$$

This means that for any given AD $A = A_1||...||A_a$, the above constructed AD B produces the same TA value.

More general, one can construct the second set of associated data as $B' = A_p \oplus E_K(X_A[p-1])||A_{p+1}||...||A_a$ for an arbitrary $p \in \{2, ..., a\}$. Using the notation introduced above, it is easy to see that this also yields $TA_A = TA_{B'}$ $\forall p \in \{2, ..., a\}$.

Notice that the observations on the TA values made in Subsection 3.2.1 and 3.2.2 are made in the classical setting only. We now come back to the discussion of the quantum aspect of the attacks presented in [LCP22]. At a high level, after having made these observations in the classical setting, they use Simon's algorithm with respect to the quantum accessibility of the encryption oracle to generate such collisions. These collisions are then used to construct forgeries in order to break EUF-qCMA security.

3.3 IND-qCPA Insecurity of AES-OTR with Parallel Associated Data Processing

In this section, we show that AES-OTR is insecure in the IND-qCPA setting with random nonces when it is used as an AEAD scheme with associated data processed in parallel. We exploit the way how AD is processed as described in Subsection 3.2.1, by finding collisions for the output value TA of Algorithm 5 used to compute the tag, as well as the fact that the encryption algorithm essentially performs a one-time pad encryption when given a single-block message.

As a general attack strategy, we want to create a periodic function f_1, whose period can be computed using Simon's algorithm. We want Simon's function f_1 to capture the property as described in Subsection 3.2.1, using a similar approach as described in [MMPR22, Section 4.3] for breaking IND-qCPA security of OCB2. Note, that the way OCB2 authenticates AD in [MMPR22, Figure 3] is (up to multiplication with constants) essentially the same as for AES-OTR in Algorithm 5. Thus, we can follow a very similar argument.

We define $f_1 : \{0,1\}^n \to \{0,1\}^\tau$

$$f_1(A) = \text{OTR-}\mathcal{E}_{K,p}(N, A||A||0^n, \varepsilon).$$

As the plaintext is chosen to be empty, the ciphertext is empty as well which is the reason the quantum encryption oracle OTR-$\mathcal{E}_{K,p}(N,\cdot)$ returns a tag of length τ only. We notice that in the context of Simon's algorithm the domain and the co-domain of Simon's function are required to be of the same dimension. Since the size of the tag τ is possibly less than n, we technically need to append an additional $n-\tau$ bits of zeros (or any other fixed bit string of size $n-\tau$). Importantly, this does not change the periodicity of f_1, as these bits are fixed. For the sake of convenience however, we refrain from appending them in each step of the analysis of f_1.

Also, Algorithm 3 treats the case where $m = 0$ is even and $|M_m| \neq n$, which produces output $TE = E_K(3^3U \oplus 0^n)$, where $U = E_K(\text{Format}(\tau, N))$. With

these observations, we notice that f_1 has period $s = 3Q = Q \oplus 2Q$ since

$$\begin{aligned}
f_1(A) &= \mathrm{msb}_\tau(TA \oplus TE) \\
&= \mathrm{msb}_\tau\Big(E_K\big(2^2 3^2 Q \oplus E_K(A \oplus Q) \oplus E_K(A \oplus 2Q)\big) \oplus TE\Big) \\
&= \mathrm{msb}_\tau\Big(E_K\big(2^2 3^2 Q \oplus E_K(A \oplus Q \oplus 2Q \oplus 2Q) \oplus E_K(A \oplus 2Q \oplus Q \oplus Q)\big) \\
&\quad \oplus TE\Big) \\
&= \mathrm{msb}_\tau\Big(E_K\big(2^2 3^2 Q \oplus E_K(A \oplus 3Q \oplus 2Q) \oplus E_K(A \oplus 3Q \oplus Q)\big) \oplus TE\Big) \\
&= f_1(A \oplus s).
\end{aligned}$$

Following the arguments made in [MMPR22, Section 3.1], using a single quantum query to the encryption oracle OTR-$\mathcal{E}_{K,p}(N, \cdot)$, the function f_1 can be computed in superposition. We need this to be done in a single quantum query, as the nonce N changes with each quantum query made to the oracle. We thus can apply Simon's algorithm to f_1, which computes a vector $y \in \{0,1\}^n$ that is orthogonal to the period $s = 3Q = 3E_K(0^n)$. Notice that there do not exist any "unwanted periods" as defined in Subsection 2.2.1 with overwhelming probability, since we can apply the same reasoning as in [KLLN16, Section 5.3] using that AES is a PRP.

It is important that the period is independent of the nonce N, such that despite the nonce changing with each quantum query, Simon's algorithm still returns a random vector y orthogonal to the fixed period. This means, after recovering $O(n)$ such independent orthogonal vectors y, we can recover the value $3Q = 3E_K(0^n)$ and thus the value $E_K(0^n)$ as well with $O(n)$ quantum queries to the AES-OTR encryption oracle.

For the rest of this section, we assume that AES-OTR is instantiated using the recommended parameter sets from [Min16]. In particular, we assume that the size of the tag is 16-bytes, i.e., $\tau = n$. This assumption is necessary for our attack to succeed. We will further discuss this assumption in Subsection 3.3.1 after having presented the attack.

As a next step towards breaking IND-qCPA security we want to gain raw block cipher access i.e. the ability to compute $E_K(inp)$ for any given input $inp \in \{0,1\}^n$. To realize this, we use Deutsch's algorithm like done in [MMPR22] as follows.

Having recovered the value $Q = E_K(0^n)$, define two fixed single-block associated data inputs $\alpha_0 = 3^2 Q$ and $\alpha_1 = 3^2 Q \oplus inp$ for any given input $inp \in \{0,1\}^n$. We continue by considering the n functions $f^{(i)} : \{0,1\} \to \{0,1\}$,

$$f^{(i)}(b) = i\text{th bit of } \{\text{OTR-}\mathcal{E}_{K,p}(N, \alpha_b, \varepsilon)\} \tag{3.3.1}$$

for a random nonce N and empty message. Here again, the output of OTR-$\mathcal{E}_{K,p}(N,\cdot)$ is only the tag of length $\tau = n$, as the message is kept empty. Following the argument in [MMPR22], we can compute $f^{(i)}(b)$ in superposition with a single quantum query to the AES-OTR encryption oracle by also truncating out the unneeded $n-1$ bits of the output of OTR-$\mathcal{E}_{K,p}(N,\cdot)$. This gives us the ability to apply Deutsch's algorithm on $f^{(i)}$ and recover the value

$$\begin{aligned} f^{(i)}(0) \oplus f^{(i)}(1) &= i\text{th bit of}\{TE \oplus E_K(\alpha_0 \oplus 3^2Q)\} \\ &\quad \oplus i\text{th bit of}\{TE \oplus E_K(\alpha_1 \oplus 3^2Q)\} \\ &= i\text{th bit of}\{E_K(0^n) \oplus E_K(inp)\} \end{aligned} \tag{3.3.2}$$

with a single quantum query, where $TE = E_K(3^3U \oplus 0^n)$ as argued above. Thus, by applying Deutsch's algorithm to each of the n functions $f^{(i)}$, we are able to recover all n bits of $\big(E_K(0^n) \oplus E_K(inp)\big)$ and from this, since we already know $E_K(0^n)$, we can recover $E_K(inp)$. It is worth pointing out, that despite the nonce N changes with each application of Deutsch's algorithm, we are still able to recover $\big(E_K(0^n) \oplus E_K(inp)\big)$ since it is independent of N.

As a result of the observations above, we can now sketch our IND-qCPA attack against AES-OTR with parallel AD processing:

1. Recover the value $3Q$ and thus the value $E_K(0^n)$ using $O(n)$ quantum queries with Simon's algorithm as discussed above.
2. Pick arbitrary but different single-block messages M_0 and M_1 and define the associated data to be empty, i.e. $A = \varepsilon$. We now give these values as an input to the challenger and record the nonce N which is used to encrypt M_b as well as the output (C^*, T) of the challenger.
3. Using Deutsch's algorithm $2n$ times (or equivalently using raw block cipher access twice) as described above, we compute the value

   $$V = E_K\Big(E_K\big(\text{Format}(\tau, N)\big)\Big)$$

   using a total of $2n$ quantum queries. Here, we use the nonce N the challenger used to encrypt the challenge query.
4. Output the bit $b'' = b'$ if $C^* = V \oplus M_{b'}$.

It remains to show that our attack outputs the correct bit b'':

To see this, we notice that for a single-block message M and empty AD the output of the encryption oracle is the ciphertext-tag pair (C^*, T) where

$$C^* = \text{msb}_{|M|}\big(E_K(L)\big) \oplus M$$

with $L = E_K(\text{Format}(\tau, N))$ following the description of Algorithm 3. Since the attack relies on recomputing exactly the value used to encrypt M_b in a one-time pad-like manner, the attack succeeds with high probability.

3.3.1 On the Necessity of the Assumption $\tau = n$

Even if AES-OTR is recommended to be instantiated using untruncated tags as mentioned in Section 3.3, we want to discuss the consequences if this is not the case.

Before we gained raw block cipher access in Section 3.3, we assumed that $\tau = n$. This is important because AES-OTR is designed to output the value $T = \text{msb}_\tau(TE \oplus TA)$ as the tag of the corresponding plaintext-AD pair. If the plaintext is empty as it is defined for Function 3.3.1, but $\tau < n$ instead, as a consequence the tag only gives us the first τ bits of $TE \oplus TA$ by definition. This means that Equation 3.3.2 would now be

$$f^{(i)}(0) \oplus f^{(i)}(1) = i\text{th bit of} \Big\{ \text{msb}_\tau\big(E_K(0^n) \oplus E_K(inp)\big) \Big\}$$

for $i \in [\tau]$ and thus instead of recovering $E_K(inp)$ we would only be able to recover $\text{msb}_\tau(E_K(inp))$ by following the same reasoning as above.

This difference affects our attack, as the ability to compute the value V in Step 3 is based on having raw block cipher access to all of the n bits of the (inner) $E_K(\text{Format}(\tau, N))$ value, because we need to apply the raw block cipher access again to this value (of now only τ bits) to compute V. Thus, we cannot compute V as straight forward as before, as we are missing information on the last $n - \tau$ bits. We can now distinguish two cases: either $n - \tau = \mathcal{O}(1)$ or $n - \tau = \mathcal{O}(n)$.

In the first case, only a constant amount of bits are truncated out from the tag and we can salvage the IND-qCPA attack by just brute-forcing all possible values of $E_K(\text{Format}(\tau, N))$, which means considering all possibilities of missing bits.

Indeed, let's choose M_0 and M_1 such that their first τ bits are different. This means that the first τ bits of their ciphertext will be different as well because we are in the case of one-time pad encryption here. Assume now that we incorrectly guess the remaining bits of $E_K(\text{Format}(\tau, N))$ and call this value W. Then, we produce a value $\tilde{V} = \text{msb}_\tau(E_K(W))$ using raw block cipher access again. But then $\text{msb}_\tau(M_0) \oplus \tilde{V} \neq \text{msb}_\tau(C^*)$ *and* $\text{msb}_\tau(M_1) \oplus \tilde{V} \neq \text{msb}_\tau(C^*)$. This means we can go on and repeat this process with our next guess for $\tilde{V}$. Eventually, we will guess the correct bits and exactly one of the above computations will hold with equality. We here are also assuming that there are no false positives, which is a valid assumption since E_K is AES which is a PRP. This procedure of course increases the complexity of our attack, but asymptotically, the run-time is still polynomial.

For the second case, we cannot adapt the attack as easily as in the first scenario, since we are truncating out a non-constant amount of bits that makes brute-forcing all possible values inefficient.

We want to point out that the IND-qCPA attack on OCB2 in [MMPR22, Section 4.3] also runs into the very same issue but it isn't addressed accordingly. This issue was confirmed by the authors in [MMPR22] and that they indeed consider only untruncated tags in their attacks [Mar23].

3.4 IND-qCPA Insecurity of AES-OTR with Serial Associated Data Processing

The aim of this section is to break IND-qCPA security of AES-OTR used as an AEAD scheme with random nonces but now with associated data processed in serial by Algorithm 6. We again exploit the way AD is processed, where we described the required observation in Subsection 3.2.2.

Similar to the attack above, we define a periodic function f_2 whose period can be computed using Simon's algorithm. In this case however, as a consequence of our choice of f_2, computing the period of the function already gives us raw block-cipher access. This is in contrast to the IND-qCPA attack in [MMPR22, Section 4.3] and our previous attack in Section 3.3 as well, where we first recover $E_K(0^n)$ and then gain raw block-cipher access via Deutsch's algorithm. In this section, Simon's function is defined in a very different fashion than before, as the function can distinguish two different cases depending on an input bit b and treat them accordingly. This affects either having one or two blocks of associated data as an input to the quantum encryption oracle respectively. We therefore also need to argue why we actually have quantum access to Simon's function we define below. This is done by coming up with a suitable quantum circuit where it is important that the circuit only uses a *single* quantum query to the encryption oracle. At this point however, we make the assumption that we have quantum access to Simon's function and are able to compute it with a *single* quantum query to the quantum encryption oracle. We emphasize this due to the changing nonces as described in Section 3.3. We validate this assumption and further discuss the quantum accessibility as well as the issue of having to achieve this with a single quantum query in Subsection 3.4.1.

We realize gaining raw block cipher access by choosing an arbitrary $B \in \{0,1\}^n$ (for which we want to know its encryption $E_K(B)$) and set Simon's function to be $f_2 : \{0,1\}^{n+1} \to \{0,1\}^\tau$,

$$f_2(b||A) = \begin{cases} \text{OTR-}\mathcal{E}_{K,s}(N, B||A, \varepsilon) & \text{if } b = 0 \\ \text{OTR-}\mathcal{E}_{K,s}(N, A, \varepsilon) & \text{if } b = 1 \end{cases}$$

where $b \in \{0,1\}$ is a single bit and $A \in \{0,1\}^n$ represents one block of associated data of size n. We follow the same reasoning as in Section 3.3 to omit adding the additional $n + 1 - \tau$ bits so that the dimensions of the

domain and the co-domain are the same. Note that here f_2 gives us a value in $\{0,1\}^\tau$, since the ciphertext is empty as a result of the plaintext being chosen as empty. More precisely, for a general set of associated data D and empty plaintext we get by Algorithm 3 and Algorithm 2 that

$$\text{OTR-}\mathcal{E}_{K,s}(N,D,\varepsilon) = \text{msb}_\tau\Big(E_K\Big(3^3 2\Big(TA_D \oplus E_K\big(\text{Format}(\tau,N)\big)\Big)\Big)\Big)$$

where $TA_D = \text{AF-S}_K(D)$. This implies that the period s of f_2 only depends on the function $\text{AF-S}_K(D)$. Therefore, we define a new function $g : \{0,1\}^{n+1} \to \{0,1\}^n$,

$$g(b||A) = \begin{cases} \text{AF-S}_K(B||A) & \text{if } b = 0 \\ \text{AF-S}_K(A) & \text{if } b = 1 \end{cases}$$

We claim that g, and therefore f_2 as well, has period $s = 1||E_K(B)$. Indeed:

$$\begin{aligned} g\big(0||A \oplus 1||E_K(B)\big) &= g\big(1||A \oplus E_K(B)\big) = \text{AF-S}_K\big(A \oplus E_K(B)\big) \\ &= \text{AF-S}_K(B||A) = g(0||A) \end{aligned} \tag{3.4.1}$$

$$\begin{aligned} g(1||A \oplus 1||E_K(B)) &= g(0||A \oplus E_K(B)) = \text{AF-S}_K(B||A \oplus E_K(B)) \\ &= E_K\Big(4Q \oplus A \oplus E_K(B) \oplus E_K(B)\Big) = E_K(4Q \oplus A) \\ &= \text{AF-S}_K(A) = g(1||A) \end{aligned} \tag{3.4.2}$$

where $Q = E_K(0^n)$ and for Equations 3.4.1 we used the property of the TA values described in Subsection 3.2.2 (for comparison, set $A_1 = B$ and $A_2 = A$ then $TA_{A_1||A_2} = TA_{A_2 \oplus E_K(A_1)}$ as described in Equation 3.2.2.). For Equations 3.4.2 we used the definition of Algorithm 6.

Under the assumption that we can in fact compute f_2 in superposition using a *single* quantum query to the encryption oracle $\text{OTR-}\mathcal{E}_{K,s}(N,\cdot)$, we can apply Simon's algorithm to f_2. Hence, with a similar argument as in Section 3.3 we can recover the value $s = 1||E_K(B)$ and thus the value $E_K(B)$ for any $B \in \{0,1\}^n$ with $O(n)$ quantum queries. It is important to mention that the period s is again independent of the nonce. So even though the nonce, and hence f_2 changes with each quantum query, Simon's algorithm still returns a random vector orthogonal to the *fixed* period $s = 1||E_K(B)$ in each of the n iterations. We conclude that this grants us raw block-cipher access without having to use Deutsch's algorithm like in the attack described in Section 3.3.

Unlike our IND-qCPA attack in Section 3.3, this attack succeeds with high probability even if the tags were truncated. This is justified because in the previous section truncation only became an issue when we applied Deutsch's algorithm. Here, we do not use Deutsch's algorithm but Simon's algorithm only. Since the function f_2 is still periodic as its periodicity is

unaffected by the truncation of the tag, running Simon's algorithm does not run into any issues.

Now we can again sketch an IND-qCPA attack against AES-OTR but this time with serial AD processing:

1. Pick arbitrary but different single-block messages M_0 and M_1 and define the AD to be empty. We give these values as an input to the challenger and record the nonce N which was used to encrypt either M_0 or M_1 as well as the output (C^*, T) of the challenger.
2. Compute the value

$$V = E_K\Big(2 \cdot E_K\big(\text{Format}(\tau, N)\big)\Big)$$

   using $2\mathcal{O}(n)$ quantum encryption queries via two applications of Simon's algorithm $\big($using the raw block-cipher access twice as discussed above, once for $B_1 = \text{Format}(\tau, N)$ and then for $B_2 = 2B_1\big)$.
3. Output the bit $b'' = b'$ if $M_{b'} = C^* \oplus V$.

To see that the above attack succeeds we note that for a single-block message M of size n and empty AD we have

$$\text{OTR-}\mathcal{E}_{K,s}(N, \varepsilon, M)\Big|_C = E_K\Big(2 \cdot E_K\big(\text{Format}(\tau, N)\big)\Big) \oplus M.$$

where by $|_C$ we indicate truncating out the tag T. This is essentially a one-time pad encryption with mask V. Since we are able to compute V, we also recover the correct bit b''.

3.4.1 On the Quantum Accessibility of the Function f_2

It remains to argue that we actually have quantum access to the function f_2 with a *single* query to the encryption oracle each time we query f_2. We need this to be done in a single quantum query as the nonce N changes with each quantum query made to the oracle. This is done by coming up with a suitable quantum circuit that describes f_2. It turned out to be quite challenging to come up with a circuit that only uses a single query to the encryption oracle because f_2 has this case distinction, where in one case the function gives two blocks of AD to the oracle and in the other it only sends a single AD block. Therefore we first present two circuits that do not meet the required properties, one that uses two quantum encryption queries and another with a first failed attempt to use only one such query. The former will also showcase why we need it to be done with a single query. Afterwards, we present one right way to achieve this.

Quantum circuit using two AES-OTR quantum encryption oracle gates. The circuit uses the two AES-OTR quantum encryption oracle unitary gates

$U^{(i)}_{\text{OTR-}\mathcal{E}_{K,s}}$ for $i = 1, 2$ and one so called Fredkin gate (as described e.g. in [TRK14, Section 2.2]), which is a controlled swap gate.

The former encryption gates model access to the AES-OTR quantum encryption oracle OTR-$\mathcal{E}_{K,s}(N, \cdot, \varepsilon)$ where one takes only a single block of AD as input and the other takes two:

$$U^{(1)}_{\text{OTR-}\mathcal{E}_{K,s}}|X\rangle|Y\rangle = |X\rangle|Y \oplus \text{OTR-}\mathcal{E}_{K,s}(N, X, \varepsilon)\rangle$$

or

$$U^{(2)}_{\text{OTR-}\mathcal{E}_{K,s}}|X\rangle|Y\rangle|Z\rangle = |X\rangle|Y\rangle|Z \oplus \text{OTR-}\mathcal{E}_{K,s}(N, X||Y, \varepsilon)\rangle$$

for arbitrary n-bit strings X, Y, Z.

On the other hand the Fredkin gate, upon input basis state (b, I_1, I_2) with b a single bit, produces output (b, O_1, O_2) where $O_1 = \bar{b}I_1 + bI_2$, $O_2 = bI_1 + \bar{b}I_2$. So the gate essentially swaps the inputs when $b = 1$ and does nothing if $b = 0$. We make use of this gate to select either $f_2(0||A)$ or $f_2(1||A)$ depending on the input bit. Note, that the function f_2 output register is the second register from the bottom.

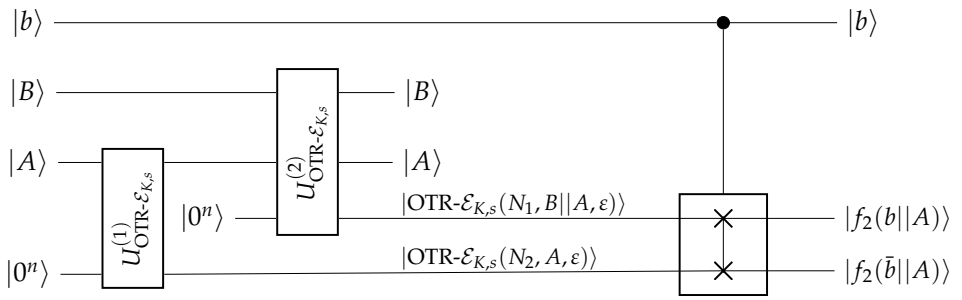

Figure 3.11: The quantum circuit implementing the function f_2 using two quantum queries via the gates $U^{(i)}_{\text{OTR-}\mathcal{E}_{K,s}}$ for $i = 1, 2$.

However, this design of the quantum circuit demonstrates the problem of using two quantum queries to the encryption oracle OTR-$\mathcal{E}_{K,s}(N, \cdot)$ when wanting to compute f_2. Indeed, assume that we compute f_2 using this quantum circuit. Since the quantum encryption oracle is called twice, the challenger used two random nonces N_1 and N_2, which are distinct with high probability, for each of the queries (as indicated in Figure 3.11). Now let us take a closer look at the periodicity argument:

$$\begin{aligned} f_2(0||A \oplus 1||E_K(B)) = f_2(1||A \oplus E_K(B)) &\stackrel{\text{def}}{=} \text{OTR-}\mathcal{E}_{K,s}(N_2, A \oplus E_K(B), \varepsilon) \\ &\neq \text{OTR-}\mathcal{E}_{K,s}(N_1, B||A, \varepsilon) \stackrel{\text{def}}{=} f_2(0||A) \end{aligned}$$

and

$$f_2(1||A \oplus 1||E_K(B)) = f_2(0||A \oplus E_K(B)) \stackrel{\text{def}}{=} \text{OTR-}\mathcal{E}_{K,s}(N_1, B||A \oplus E_K(B), \varepsilon)$$
$$\neq \text{OTR-}\mathcal{E}_{K,s}(N_2, A, \varepsilon) \stackrel{\text{def}}{=} f_2(1||A).$$

Note, that the only reason this does not hold with equality is that different nonces N_1 and N_2 were used with high probability. Otherwise we would have equality following the argument from Equations 3.4.1 and 3.4.2. This is the reason why we cannot use more than one quantum encryption query in the quantum circuit for f_2.

An unsuccessful attempt at designing a quantum circuit using a single AES-OTR quantum encryption oracle gate. We here show another attempt at a quantum circuit that uses only a single $U_{\text{OTR-}\mathcal{E}_{K,s}}$ gate, but fails to compute f_2 due to a different issue. The idea was to just swap the states $|A\rangle$ and $|B||A\rangle$ with a Fredkin gate before an application of the gate $U_{\text{OTR-}\mathcal{E}_{K,s}}$.

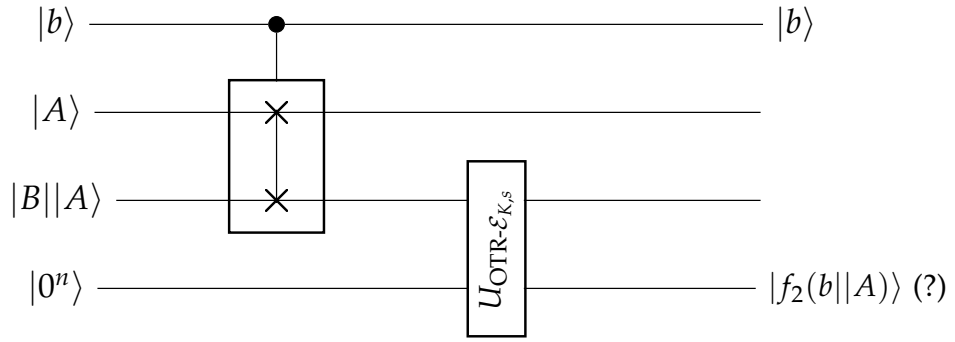

Figure 3.12: The (wrong) quantum circuit implementing the function f_2 using a single quantum query via the gate $U_{\text{OTR-}\mathcal{E}_{K,s}}$. With the question mark indicating the incorrectness.

Where this circuit falls short is already at the controlled swap gate. This is because the swap gate can only operate on inputs that are of the same length, as the swapping is done as a bitwise operation. By notating the circuit like this, only the first n bit of the state $|B||A\rangle$ are swapped with the state $|A\rangle$. Thus, the gate $U_{\text{OTR-}\mathcal{E}_{K,s}}$ would either have $|B||A\rangle$ as an input state (case $b = 0$) or $|A||A\rangle$ as input state (case $b = 1$) which obviously is not the way f_2 is intended to operate. Therefore the intent to swap the $|A\rangle$ and $|B||A\rangle$ cannot be conducted like this.

Furthermore we run into dimension issues with the $U_{\text{OTR-}\mathcal{E}_{K,s}}$ gate. As each gate of a quantum circuit can be described with a unitary matrix, a matrix handling the $|A\rangle$ state would need to have dimension $2n \times 2n$ and dimension $3n \times 3n$ in the case of the $|B||A\rangle$ state which of course is not possible at the same time.

How to correctly get quantum access to f_2. We now present a way on how to get quantum access to f_2 using a single quantum encryption query. To achieve this, we have to modify the quantum encryption oracle queries in a slight way: we add an additional n-qubit input register which encodes the length of our message. By doing so, the encryption oracle knows how many bits of the message it should actually encrypt. Thus, we also need to define f_2 in a different manner:

$$f_2(b||A) = \begin{cases} \text{OTR-}\tilde{\mathcal{E}}_{K,s}(N, \text{bin}(2n,n)||B||A, \varepsilon) & \text{if } b = 0. \\ \text{OTR-}\tilde{\mathcal{E}}_{K,s}(N, \text{bin}(n,n)||A||0^n, \varepsilon) & \text{if } b = 1. \end{cases}$$

Note, that this way both $|\text{bin}(2n,n)||B||A\rangle$ and $|\text{bin}(n,n)||A||0^n\rangle$ are $3n$ qubit states and hence can be swapped (as discussed above). In this case, when the encryption oracle gets such an input, it first parses the first n qubits of the query to figure out how many blocks of the input have to be encrypted. For $b = 0$ it just takes $B||A$ as an input but if $b = 1$ then it ignores the remaining 0^n block of the query and only encrypts A. The corresponding quantum circuit therefore looks as follows:

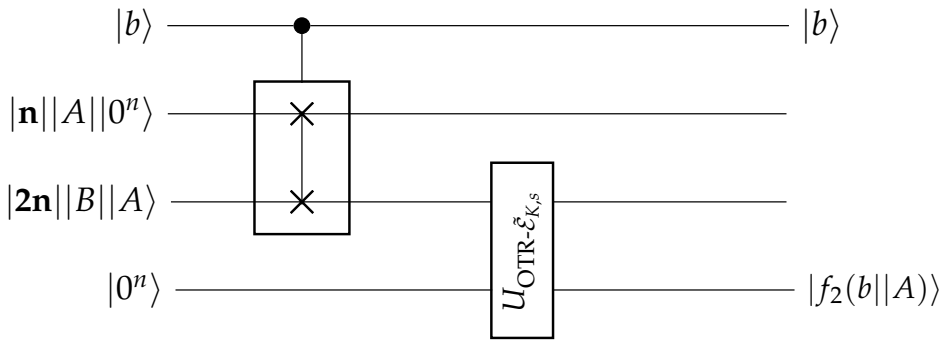

Figure 3.13: The quantum circuit implementing the function f_2 using a single quantum query via the gate $U_{\text{OTR-}\tilde{\mathcal{E}}_{K,s}}$. Note, that we indicate the n-bit encoding of n and $2n$ with bold letters.

Note that the output of the second register is set to $|\mathbf{n}||A||0^n\rangle$ if $b = 0$ and $|\mathbf{2n}||B||A\rangle$ if $b = 1$. For the third register it is the other way round.

3.5 IND-qCPA Insecurity of AES-OTR when Nonces are Chosen Adaptively

So far, our IND-qCPA analysis of AES-OTR relied on exploiting the processing of AD in either parallel or serial manner. We now want to break confidentiality by considering AES-OTR as a pure AE scheme. To do so, we consider a stronger adversary that can adaptively pick the nonces (in a non repeating manner) and hand it to the challenger. The challenger then has to

respond to encryption queries in the IND-qCPA security game using these nonces. We adapt the attack model from [MMPR22, Section 4.4], as well as the general strategy.

The strategy in this case is to recover the value $U = E_K(\text{Format}(\tau, N))$ for a randomly chosen nonce N using Simon's algorithm and picking a new nonce based on the recovered value. The challenger then has to use this newly picked nonce to answer the challenge query which enables us to perform an IND-qCPA attack. However, the formatting of the nonce causes our attack to be a non-trivial extension to the one in [MMPR22]: before picking the new nonce, we have to check whether U has a desired format and if this is not the case, we have to repeat the recovery step. Observe, that here compared to our attacks in Sections 3.3 and 3.4, the value we want to recover is dependent on the nonce N. So the strategy to use $\mathcal{O}(n)$ quantum oracle queries in an application of Simon's algorithm with respect to a function as used in the previous section would not work. The reason is that if Simon's function has a period depending on U, then in each step of Simon's algorithm, we would recover a vector orthogonal to a *different* value of the period. This issue arises because the nonce changes with every query and hence, U and the corresponding period also change. To resolve this obstacle, we define a function with numerous independent periods and an input that consists of multiple plaintext blocks (specifically, $4n$ blocks). By using this approach, that is inspired by the attack presented in [BBC$^+$21, Section 3.2] (and [MMPR22, Section 4.4]), we can overcome the challenge of changing nonces, as we describe in this section.

Note, that even though encryption of AES-OTR has some slight differences when AD is processed in either parallel or serial, Algorithm 3 executes the same steps where only the value U differs by a factor of 2 if the associated data is kept empty. Thus, our attack works in both settings, but we treat the parallel case here.

Let $m \neq 0$ be even. Recall that for a message $M = M_1||...||M_m||0^n$ with $M_i \in \{0,1\}^n \ \forall i \in [m]$ we get ciphertext $C = C_1||...||C_m||C_{m+1}$ where

$$C_{2k-1} = E_K\Big(2^{k-1}U \oplus M_{2k-1}\Big) \oplus M_{2k}$$
$$C_{2k} = E_K\Big(2^{k-1}3U \oplus C_{2k-1}\Big) \oplus M_{2k-1}$$

for $k \in [m/2]$ are n-bit blocks as well. We set the last plaintext block to be all zeros since this block is encrypted differently, and we only want to focus on properties of the encryption of the first m blocks.

We define functions $h_{2k-1} : \{0,1\}^n \to \{0,1\}^n$ such that

$$h_{2k-1}(M) = E_K\Big(2^{k-1}U \oplus M\Big) \oplus M$$

for $k \in [m/2]$ which correspond to the $(2k-1)$-th ciphertext block C_{2k-1} when $M_{2k-1} = M_{2k} =: M$. Further, by defining $s := 2^{k-1}U \oplus 2^k U$ we find that

$$\begin{aligned} h_{2k-1}(M \oplus s) \oplus h_{2(k+1)-1}(M \oplus s) &= E_K\big(2^{k-1}U \oplus M \oplus s\big) \oplus E_K\big(2^k U \oplus M \oplus s\big) \\ &= E_K\big(2^k U \oplus M\big) \oplus E_K\big(2^{k-1}U \oplus M\big) \\ &= h_{2(k+1)-1}(M) \oplus h_{2k-1}(M). \end{aligned} \tag{3.5.1}$$

So, if we define $H_{2k-1,2(k+1)-1} : \{0,1\}^n \to \{0,1\}^n$ as

$$H_{2k-1,2(k+1)-1}(M) = h_{2k-1}(M) \oplus h_{2(k+1)-1}(M)$$

this function is in fact periodic with period $s = 2^{k-1}U \oplus 2^k U$, as the above calculations indicate.

With the function $H_{2k-1,2(k+1)-1}$ we basically link four consecutive plaintext blocks (more precisely we set $M = M_{2k-1} = M_{2k} = M_{2(k+1)-1} = M_{2(k+1)}$) in order to create a periodic function. We will further develop this idea by first encrypting $4(n+1)+1$ plaintext blocks followed by an application of a linear function that captures exactly this observation.

Consider the function $g : \{0,1\}^{n(4(n+1)+1)+\tau} \to \{0,1\}^{n(n+1)}$

$$g(C_1, ..., C_{4(n+1)+1}, T) = (C_1, C_5 \oplus C_7, C_9 \oplus C_{11}, ..., C_{4n+1} \oplus C_{4n+3})$$

Here, the C_i's are n-bit blocks and T is a τ-bit block. It is not hard to see that g satisfies $g(C \oplus C') = g(C) \oplus g(C')$ for any valid inputs C and C' i.e. g is a linear function.

We further define the function $f_N : \{0,1\}^{n^2} \to \{0,1\}^{n(n+1)}$ such that

$$f_N(M_1, ..., M_n) = g \circ \text{OTR-}\mathcal{E}_{K,p}(N, \varepsilon, 0^{4n} || M_1^4 || ... || M_n^4 || 0^n)$$

for some randomly chosen nonce N and empty AD[1]. Here, we included the last plaintext block 0^n since the last block (note, that we have an odd amount of blocks) is treated differently by the encryption algorithm and we want to avoid having to analyze encryption in this case. This choice allows us to just ignore the last block of ciphertext as it is irrelevant for our attack. We can also write f_N in terms of the functions h_{2k-1} and $H_{2k-1,2(k+1)-1}$ from above. To be precise, it holds

$$f_N(M_1, ..., M_n) = \big(h_1(0^n), H_{5,7}(M_1), ..., H_{4n+1,4n+3}(M_n)\big). \tag{3.5.2}$$

We proceed by claiming that f_N has the n linearly independent periods $\langle s_i \rangle_{i \in [n]}$ with $s_i = (s_{i,1}, ..., s_{i,n})$ and $s_{i,j} \in \{0,1\}^n$, where

$$s_i = \Big((0^n)^{i-1} || 2^{2i} U \oplus 2^{2i+1} U || (0^n)^{n-i}\Big).$$

[1] Here, by M_i^4 we mean the concatenation of four copies of M_i.

Hence, only the entry $s_{i,i}$ is non-zero $\forall i \in [n]$. This is the crucial property required for our attack to succeed, as we then are able to let f_N be Simon's function to which we can apply Simon's algorithm. Indeed, when we take a look at $f_N(M_1, ..., M_i \oplus 2^{2i}U \oplus 2^{2i+1}U, ..., M_n)$ it is enough to consider the $(i+1)$-st entry (as it is the only entry affected by the period s_i) given from Equation 3.5.2 for which it holds

$$H_{4i+1,4i+3}(M_i \oplus s_{i,i}) = H_{4i+1,4i+3}(M_i),$$

as we have shown in Equation 3.5.1 (with suiting choice of indices).

Hence we can follow the reasoning in [MMPR22, Section 4.4] and apply [BBC$^+$21, Lemma 2] that assures the ability to compute a linear function of an output of a quantum oracle. Using this Lemma, we can compute f_N with a single quantum query to the OTR-$\mathcal{E}_{K,p}(N, \cdot)$ oracle. Again, with similar arguments as in [MMPR22] this in turn allows us to apply Simon's algorithm to f_N where with a *single* quantum query we are able to recover a vector $y = (y_1, ..., y_n) \in \{0,1\}^{n^2}$, $y_i \in \{0,1\}^n \forall i \in [n]$ that is orthogonal to each of the n periods $\langle s_i \rangle_{i \in [n]}$.

With overwhelming probability the algorithm successfully computes such a vector y because we can apply a similar argument as in [BBC$^+$21] to argue that there do not exist any "unwanted periods" in f_N to which y could be orthogonal to. To be precise, if we assume the existence of an unwanted period s' of f_N with a probability greater than $\frac{1}{2}$, then at least one of the $H_{4i+1,4i+3}$ in Equation 3.5.2 would also admit an unwanted period $s'_{4i+1,4i+3}$ with probability greater than $\frac{1}{2n}$. But this is impossible as E_K being AES does not have a high-probability higher-order differential.

Hence, we can continue by solving the n equations we get from orthogonality

$$y_i \cdot (2^{2i}U \oplus 2^{2i+1}U) = 0$$

for $i \in [n]$ and we are thus able to recover the value $U = E_K(\text{Format}(\tau, N))$. Using only a single quantum query to recover U is crucial. Indeed, if we want to compute $U = E_K(\text{Format}(\tau, N))$ and we make a second quantum encryption query, then the nonce changes to an independent N'. Hence, the corresponding output of Simon's algorithm with respect to the second quantum encryption query depends on N' but is (most likely) independent of N. Therefore it is not cleat how the system of linear equations obtained with respect to the second query (with independent nonce N') helps in recovering $U = E_K(\text{Format}(\tau, N))$.

Furthermore, as described in [MMPR22] we are also able to recover the *fixed* first ciphertext block $C_1 = E_K(U)$ by measuring the quantum register corresponding to the output of f_N as a part of Simon's algorithm. It is

important that the plaintext block is fixed to 0^n (or any other fixed classical value), so that C_1 is a classical value.

Using these values we can now formulate our IND-qCPA attack in a setting with adaptively chosen nonces. Note however, that in difference to the attack in [MMPR22], we need an additional step as here the value U differs from the one used in OCB2: for AES-OTR there is an additional formatting applied to the nonce N which does not exist for OCB2. This creates the problem that we cannot just define our new nonce N^* to be the recovered value U as it was the case in the [MMPR22] attack. because it may not have the correct format to satisfy $C_1 = E_K(U) = E_K(E_K(\text{Format}(\tau, N^*))$ which is what we need in order for our attack to work. We resolve this issue by iterating the above steps for different initial nonces N, until we find one that satisfies the desired condition. We will formalize this idea now:

1. For a randomly chosen nonce N we recover in a *single* quantum encryption query the classical values $U = E_K(\text{Format}(\tau, N))$ and $C_1 = E_K(U)$ using Simon's algorithm as described above.
2. Check if U is of the form

   $$U = \text{Format}(\tau, N') = \text{bin}(\tau \bmod n, 7)||0^{n-8-|N'|}||1||N'$$

   for some $N' \in \{0,1\}^{8i}$ where $i \in [15]$. If this condition is not satisfied, we repeat step 1 for a different nonce N and else we continue with step 3.
3. Choose the nonce N^* to be $N^* = N'$ such that we guarantee the corresponding initial offset U^* in the challenge query to be

   $$U^* = E_K(\text{Format}(\tau, N^*)) = E_K(U) = C_1$$

   where C_1 is the value we recovered in the first step.
4. Define $m_0 = U^* \oplus U$, $m_1 = 0^n$, where U is the value we recovered in the first step with the desired format of step 2, and a random $m_0' \in \{0,1\}^n$ such that $m_0 \neq m_0'$. Select the two 2-block messages as $M_0 = m_0||m_1$ and $M_1 = m_0'||m_1$ and $A = \epsilon$ for the challenge query.
5. Record the the response (C^*, T^*) from the challenger, where $C^* = C_1^*||C_2^*$ with $C_i^* \in \{0,1\}^n$ and output $b' = 0$ if $C_2^* = C_1$ and $b' = 1$ else.

The attack succeeds because as a result of our choice of messages and nonce for the challenge query the response (C^*, T^*) satisfies

$$C_2^* = \begin{cases} E_K(U^* \oplus m_0) \oplus m_1 = E_K(U) = C_1 & \text{if } M_0 \text{ was encrypted.} \\ E_K(U^* \oplus m_0') \oplus m_1 = E_K(U^* \oplus m_1') \neq C_1 & \text{if } M_1 \text{ was encrypted.} \end{cases}$$

Note, that when we choose M_0 and M_1 to have two blocks of plaintext, they are encrypted following the case in lines 11 to 14 of Algorithm 3.

Also, Step 2 succeeds with high probability. Indeed, if we assume that the size of the nonce is $\kappa = 120$ (in general $\kappa \in \{8, 16, ..., 120\}$) we essentially require the first 8 bits of U to be fixed to $\text{bin}(\tau \bmod n, 7)||1$. This happens with probability $(1/2)^8$ and is even bigger when we allow $\kappa \in \{8, 16, ..., 120\}$ (note, that then in general $n - \kappa$ bits need to be fixed). Therefore this step eventually is successful.

3.6 Refining the EUF-qCMA Attacks on AES-OTR

We conclude the analysis on AES-OTR by refining the EUF-qCMA attacks presented in [LCP22] on the mode with respect to our IND-qCPA attacks in Sections 3.3 and 3.4. Following Definition 2.5 of EUF-qCMA security, after having made q many encryption queries, the adversary has to produce $q+1$ forgeries (N, A, C, T) with any nonces of its own choice.

3.6.1 AD Processed in Parallel

We sketch the EUF-qCMA attack on AES-OTR below assuming associated data is processed in parallel:

1. Recover the value $Q = E_K(0^n)$ using $\mathcal{O}(n)$ quantum encryption queries as outlined in Section 3.3.
2. Query the authenticated encryption C, T of $M, A = A_1||...||A_k$ for an arbitrary message M and arbitrary but pairwise different blocks of AD A_i such that $\binom{k}{2} \geq n + 2$ and $A_i \oplus A_j \neq 2^{i-1}Q \oplus 2^{j-1}Q \ \forall i, j \in [k], i \neq j$ and record the nonce N chosen by the challenger.
3. Output the forgeries $(N, B^{(p,q)}, C, T)$, where N, C, T as above and associated data $B^{(p,q)} = B_1||...||B_k$ for all $p, q \in [n]$, $p \neq q$ such that

$$\begin{aligned} B'_q &= A_p \oplus 2^{p-1}Q \oplus 2^{q-1}Q \\ B'_p &= A_q \oplus 2^{p-1}Q \oplus 2^{q-1}Q \\ B'_i &= A_i \end{aligned}$$

   for $i \in [k] \setminus \{p, q\}$.

We choose $A_i \oplus A_j \neq 2^{i-1}Q \oplus 2^{j-1}Q \ \forall i, j \in [k], i \neq j$ because otherwise $B^{(i,j)} = A$ and thus $(N, B^{(i,j)}, C, T) = (N, A, C, T)$ would only be counted as a single forgery. But given that there are $\binom{k}{2}$ constraints, and roughly of the order 2^n choices for each A_i satisfying the above constraints, we can always choose the above $A = A_1||...||A_k$.

This attack is successful as we made n (quantum) $+1$ (classical) encryption queries and produce $\binom{k}{2} \geq n+2$ pairwise different sets of associated data that all generate the same value TA as we described in Equations 3.2.1 in Subsection 3.2.1. Therefore following AES-OTR encryption when AD is processed in parallel as described in Algorithm 1, it holds that OTR-$\mathcal{E}_{K,p}(N, B^{(p,q)}, M) = (C, T)$ for all $p, q \in [k]$, $p \neq q$.

3.6.2 AD Processed in Serial

In the setting of AD being processed in serial we sketch the EUF-qCMA attack on AES-OTR below:

1. Choose some $A_1 \in \{0,1\}^n$ and compute $E_K(A_1)$ via one application of the raw block cipher access in $\mathcal{O}(n)$ quantum encryption queries as outlined in Section 3.4.
2. Choose pairwise different $A_{2,i} \in \{0,1\}^n$ for $i \in [n+1]$ and for each $A^{(i)} = A_1 || A_{2,i}$ query the authenticated encryption C_i, T_i of $M_i, A^{(i)}$, where M_i is an arbitrary message and record the corresponding nonce N_i the challenger used.
3. Output the forgeries $(N_i, A^{(i)}, C_i, T_i)$ and $(N_i, E_K(A_1) \oplus A_{2,i}, C_i, T_i)$ for $i \in [n+1]$.

This attack is successful as we produce $2n+2$ forgeries having made $2n+1$ (quantum) encryption queries. The forgeries are valid, as half of them are produced by the oracle and the other half are constructed following the arguments in Subsection 3.2.2, where we argue that for any AD $A = A_1 || A_2$, the AD $B = E_K(A_1) \oplus A_2$ produce the same value TA computed by Algorithm 6. Therefore, following AES-OTR encryption when associated data is processed in serial in Algorithm 2, it holds that OTR-$\mathcal{E}_{K,s}(N_i, E_K(A_1) \oplus A_{(2,i)}, M_1) = (C_i, T_i)$ $\forall i \in [n+1]$.

Chapter 4

The Key (-Recovery) Issue of OPP

The *Offset Public Permutation Mode* (OPP) was proposed in [GJMN15] and [GJMN16] and it essentially tries to generalize OCB3 by replacing the underlying block cipher by a public permutation and applying a different form of masking. In this chapter we will show that using a public permutation the way OPP does, actually leads to a devastating key recovery attack in the quantum setting.

The attack uses a similar strategy as the one in Section 3.5 but instead of recovering $E_k(Format(\tau, N^*))$ and choosing a new nonce adaptively to break IND-qCPA security, we are able to recover the value $\Omega := P(X||K)$ where P is a efficiently invertible public permutation, X can be seen as a formatting of the nonce and K is the key. In contrast to OTR being based on a block cipher that may only be inverted knowing the key, we are here dealing with a public permutation that can be inverted efficiently. This is the key issue of OPP and the reason we are able to recover the key knowing Ω.

4.1 Specifications of OPP

In this chapter, it is assumed that the plaintexts we are dealing with always have a size that is a multiple of the block length. As a consequence, it is not necessary to treat the last block of the plaintext any differently in our analysis, and furthermore we are also excluding the specifications for how OPP encrypts plaintexts that do not meet this assumption. In the same manner this also applies to the way we describe the processing of associated data.

Let n, k, τ, κ as labeled in Chapter 2 such that $\kappa \leq n - k - 1$. We begin by describing the OPP mode as proposed in [GJMN15] in a simplified manner that also maintains consistent labeling of variables in previous descriptions of modes such as OTR.

A set of functions $\Phi = \{\alpha, \beta, \gamma\}$ is given by $\alpha, \beta, \gamma : \{0,1\}^n \to \{0,1\}^n$, $\alpha(x) = \varphi(x)$, $\beta(x) = \varphi(x) \oplus x$ and $\gamma(x) = \varphi^2(x) \oplus \varphi(x) \oplus x$ where for $x = x_0||...||x_{15}$ and $x_i \in \{0,1\}^{64}$ the function $\varphi : \{0,1\}^{1024} \to \{0,1\}^{1024}$ is defined as

$$\varphi(x_0, ..., x_{15}) = \big(x_1, ..., x_{15}, (x_0 \lll 53) \oplus (x_5 \ll 13)\big).$$

OPP uses the so called tweakable Even-Mansour construction MEM, where a tweak space $\mathcal{T}$ of the form $\mathcal{T} \subseteq \{0,1\}^{n-k} \times \mathbb{N}^3$, as outlined in [GJMN15, Lemma 4], is considered. For further details about tweaks and tweakable block ciphers we refer to [GJMN15] as this specific notion is not relevant for the subsequent attack.

The encryption function $\widetilde{E} : \{0,1\}^k \times \mathcal{T} \times \{0,1\}^n \to \{0,1\}^n$ is then defined as

$$\widetilde{E}(K, X, \bar{i}, M) = P\big(\delta(K, X, \bar{i}) \oplus M\big) \oplus \delta(K, X, \bar{i})$$

where $\delta : \{0,1\}^k \times \mathcal{T} \to \{0,1\}^n$ is called the masking function and for $\bar{i} = (i_0, i_1, i_2) \in \mathbb{N}^3$ it is set to be

$$\delta(K, X, \bar{i}) = \gamma^{i_2} \circ \beta^{i_1} \circ \alpha^{i_0}\big(P(X||K)\big)$$

For convenience the shorthand notation $\widetilde{E}^{\bar{i}}_{K,X}(M) = \widetilde{E}(K, X, \bar{i}, M)$ is being used in the algorithmic description of OPP.

It should be noted that P is a public permutation of which it is required that P^{-1} can be computed efficiently, as its inverse is necessary to perform decryption. In particular decryption is done as follows:

$$\widetilde{D}(K, X, \bar{i}, C) = P^{-1}\big(\delta(K, X, \bar{i}) \oplus C\big) \oplus \delta(K, X, \bar{i}).$$

Below, we present a simplified description of the OPP mode based on the specification in [GJMN15]. We only provide a description of the encryption and authentication part of the algorithm, as the details regarding decryption are not relevant to our attack and are therefore omitted. To be precise, the authentication part of OPP is described in Algorithm 9 and the encryption part corresponds to Algorithm 8.

Algorithm 7 OPP-$\mathcal{E}(K, N, AD, M)$

1: $X \leftarrow \text{pad}^0_{n-\kappa-k}(N)$
2: $C, S \leftarrow \text{OPPEnc}(K, X, M)$
3: $T \leftarrow \text{OPPAbs}(K, X, AD, S)$
4: **return** C, T

Algorithm 8 OPPEnc(K, X, M)

$M_0||...||M_{m-1} \leftarrow M$, s.t. $|M_i| = n$
$C \leftarrow \varepsilon$
$S \leftarrow 0^n$
for $i \in \{0, ..., m-1\}$ **do**
    $C_i \leftarrow \widetilde{E}^{i,0,1}_{K,X}(M_i)$
    $C \leftarrow C||C_i$
    $S \leftarrow S \oplus M_i$
return $C, \widetilde{E}^{m-1,2,1}_{K,X}(S)$

Algorithm 9 OPPAbs(K, X, A, S)

$A_0||...||A_{a-1} \leftarrow A$, s.t. $|A_i| = n$
$S' \leftarrow 0^n$
for $i \in \{0, ..., a-1\}$ **do**
    $S' \leftarrow S' \oplus \widetilde{E}^{i,0,0}_{K,X}(A_i)$
return $\mathrm{msb}_\tau(S' \oplus S)$

4.2 Quantum Key-Recovery Attack on OPP

In this attack, we employ the same techniques as in Section 3.5, which were adapted from the methods in [MMPR22] for breaking IND-qCPA security of AES-OTR with adaptively chosen nonces. However, in this case we are able to recover the key instead. Our attack is focused solely on the encryption part, so OPP it is used as a pure AE scheme, and does not make use of the way associated data is processed. This is in contrast to our previous attacks in Sections 3.3 and 3.4 as there we could exploit the fact that AD processing was not dependent on a nonce but rather on the constant value $Q = E_k(0^n)$. In the case of OPP this is different because the nonce is used in the associated data processing, as the value $P(X||K)$ where $X = \mathrm{pad}^0_{n-\kappa-k}(N)$ is dependent on the nonce N (see Algorithm 7). Since the nonce changes with each call to the encryption oracle, OPP never processes a fixed set of AD the same way. This is the reason we can't apply Simon's algorithm (which calls the oracle multiple times) in the same manner.

Before we formulate the attack itself, we observe a crucial property of the xor of two consecutive ciphertext blocks considered as a function of its corresponding plaintext blocks. Define $\Omega = P(X||K)$ and recall that OPP encrypts the i-th plaintext block M_i as $C_i = P(\delta(K, X, (i, 0, 1)) \oplus M_i) \oplus \delta(K, X, (i, 0, 1))$ where $\delta(K, X, (i, 0, 1)) = \varphi^{i+2}(\Omega) \oplus \varphi^{i+1}(\Omega) \oplus \varphi^i(\Omega)$. We now define functions $f_i : \{0,1\}^n \to \{0,1\}^n$ such that

$$f_i(M) = P\big(\delta(K, X, (i, 0, 1)) \oplus M\big) \oplus \delta(K, X, (i, 0, 1)),$$

which correspond to the i-th ciphertext block C_i considered as a function of its underlying plaintext block M. Further, by defining $s := \varphi^{i+3}(\Omega) \oplus \varphi^i(\Omega)$ we see that

$$\begin{aligned}
&f_i(M \oplus s) \oplus f_{i+1}(M \oplus s) \\
&\quad = P\Big(\varphi^{i+2}(\Omega) \oplus \varphi^{i+1}(\Omega) \oplus \varphi^i(\Omega) \oplus M \oplus s\Big) \\
&\qquad \oplus P\Big(\varphi^{i+3}(\Omega) \oplus \varphi^{i+2}(\Omega) \oplus \varphi^{i+1}(\Omega) \oplus M \oplus s\Big) \oplus \varphi^{i+3}(\Omega) \oplus \varphi^i(\Omega) \\
&\quad = P\Big(\varphi^{i+3}(\Omega) \oplus \varphi^{i+2}(\Omega) \oplus \varphi^{i+1}(\Omega) \oplus M\Big) \\
&\qquad \oplus P\Big(\varphi^{i+2}(\Omega) \oplus \varphi^{i+1}(\Omega) \oplus \varphi^i(\Omega) \oplus M\Big) \oplus \varphi^{i+3}(\Omega) \oplus \varphi^i(\Omega) \\
&\quad = f_{i+1}(M) \oplus f_i(M)
\end{aligned}$$

So if we define $F_{i,i+1} : \{0,1\}^n \to \{0,1\}^n$ as $F_{i,i+1}(M) = f_i(M) \oplus f_{i+1}(M)$ we see from the above calculations that $F_{i,i+1}(M \oplus s) = F_{i,i+1}(M)$, i.e., $F_{i,i+1}$ is a periodic function with period $s = \varphi^{i+3}(\Omega) \oplus \varphi^i(\Omega)$.

The idea is now to apply a linear function to $2n+1$ ciphertext blocks to capture this observation and create a periodic function that itself contains n copies of the periodic function $F_{i,i+1}$ from above. To do so consider the function $g : \{0,1\}^{(2n+1)n+\tau} \to \{0,1\}^{(n+1)n}$

$$g(C_0, C_1, ..., C_{2n}, t) = (C_0, C_1 \oplus C_2, ..., C_{2n-1} \oplus C_{2n}).$$

Here, the C_i's are n-bit blocks and t is a τ-bit block. It is not hard to see that g is in fact a linear function - i.e., it satisfies $g(C \oplus C') = g(C) \oplus g(C')$ for any valid inputs C and C'. Furthermore let $\tilde{f}_N : \{0,1\}^{n^2} \to \{0,1\}^{(n+1)n}$ such that

$$\tilde{f}_N(M_1, ..., M_n) = g \circ \text{OPP-}\mathcal{E}(K, N, \varepsilon, 0^n || M_1 || M_1 || M_2 || ... || M_n || M_n) \quad (4.2.1)$$

for some randomly chosen nonce N and empty associated data. We can also reformulate $\tilde{f}_N$ in terms of the functions f_i and $F_{i,i+1}$ from above. To be precise, it holds

$$\tilde{f}_N(M_1, ..., M_n) = \big(f_0(0^n), F_{1,2}(M_1), ..., F_{2n-1,2n}(M_n)\big). \quad (4.2.2)$$

Moreover, we have included an all-zero plaintext block at the beginning, which will be useful later on for verifying correctness of the recovered key.

The crucial property required for the success of the attack is that $\tilde{f}_N$ has n linearly independent periods $\langle s_i \rangle_{i \in [n]}$ where

$$s_i = \Big((0^n)^{i-1} || \varphi^{2i+2}(\Omega) \oplus \varphi^{2i-1}(\Omega) || (0^n)^{n-i}\Big).$$

This directly follows from the observation on the periodicity of $F_{i,i+1}$ and the fact that each pair of the two consecutive ciphertext blocks C_{2i-1} and C_{2i} for $i \in [n]$ encrypt the same plaintext block M_i but with different mask δ.

Following the same argument as in Section 3.5, we can apply [BBC+21, Lemma 2] which assures the ability to compute a linear function of a quantum oracle's output. Therefore, we can compute $\tilde{f}_N$ with a single quantum query to the OPP-$\mathcal{E}(K, N, \cdot)$ oracle. Once more, we can apply Simon's algorithm to $\tilde{f}_N$ which, with a single quantum query, recovers a vector $y = (y_1, ..., y_n) \in \{0,1\}^{n^2}$ with $y_i \in \{0,1\}^n \ \forall i \in [n]$ that is orthogonal to each of the periods s_i. The algorithm successfully computes such a vector with overwhelming probability as there do not exist any "unwanted periods" to which y could be orthogonal to.

We justify this claim by building upon the argument presented in [BBC+21, Section 3.2], which treats the absence of "unwanted periods" in a very similar attack on a variant of OCB. We recall that OCB uses a block cipher instead of a public permutation like it is the case for OPP, so we need to adjust the reasoning to the setting of a public permutation. If we assume the existence of an unwanted period $\tilde{s}$ of $\tilde{f}_N$ with a probability greater than $\frac{1}{2}$, then at least one of the $F_{i,i+1}$ in Equation 4.2.2 would also have to admit an unwanted period $\tilde{s}_{i,i+1}$ with probability greater than $\frac{1}{2n}$. We now draw upon the reasoning presented in [KLLN16, Section 3.2], which shows the non-occurrence of higher order differentials in the Even-Mansour construction. More accurately, $F_{i,i+1}$ admitting such an unwanted period is equivalent to saying that P admits a high-probability higher-order differential. But these only happen with negligible probability for a random choice of P according to [KLLN16]. This argument makes sure that the probability for unwanted periods to appear is bounded and thus Simon's algorithm computes a vector y as described above with overwhelming probability.

By orthogonality of y we get n equations of the form

$$y_i \cdot \left(\varphi^{2i+2}(\Omega) \oplus \varphi^{2i-1}(\Omega)\right) = 0. \tag{4.2.3}$$

Before we can proceed, we recall that for $x = x_0||...||x_{15}$ and $x_i \in \{0,1\}^{64}$ the function φ is defined as

$$\varphi(x_0, ..., x_{15}) = \left(x_1, ..., x_{15}, (x_0 \lll 53) \oplus (x_5 \ll 13)\right)$$

As described in [GJMN15] we see that the function φ is in fact a linear map and it therefore can be represented by a matrix M. Following this, Equation 4.2.3 is equivalent to

$$y_i \cdot \left(M^{2i+2} \cdot \Omega \oplus M^{2i-1} \cdot \Omega\right) = 0 \tag{4.2.4}$$

Knowing M and using associativity of matrix multiplication, we are able to solve the n Equations in 4.2.4 and thus recover the value $\Omega = P(X||K)$. But since P is a public permutation and its inverse is assumed to be computable efficiently due P^{-1} being needed for decryption, we can just apply P^{-1} to Ω and we thus are able to acquire $X||K$. In particular, we gain possession of the key K.

Observe, that in addition when running Simon's algorithm, we recover the fixed classical value of the first ciphertext block $C_0 = \widetilde{E}^{0,0,1}_{K,X}(0^n)$ when we measure the quantum register corresponding to the output of $\tilde{f}_N$ as part of our application of Simon's algorithm. It is important that we fix the value to 0^n (or any other arbitrary fixed classical value of length n also works) in order for C_0 to be a classical value.

We sketch our key-recovery attack below:

1. Given access to a quantum encryption oracle of OPP for a random nonce N, i.e., OPP-$\mathcal{E}(K, N, \varepsilon, \cdot)$, we recover with a single quantum encryption query the classical values $\Omega = P(X||K)$ and $C_0 = \widetilde{E}^{0,0,1}_{K,X}(0^n)$ as discussed above. In particular, we recover the classical value of the key K.
2. We perform a sanity check on the key: using Ω we recompute the encryption $\widetilde{C}$ of the one-block plaintext 0^n with respect to the same key K and nonce N as used in the above encryption oracle query as

$$\widetilde{C} = P\big(\varphi^2(\Omega) \oplus \varphi(\Omega) \oplus \Omega\big) \oplus \varphi^2(\Omega) \oplus \varphi(\Omega) \oplus \Omega.$$

   Check that $C_0 = \widetilde{C}$. If this turns out to be false we repeat step 1., else we are certain to have recovered the right key K.

It remains to argue why this attack is successful.

We perform a sanity check in order to assure correctness of the key K. This is where the first ciphertext block is useful. Indeed, having recovered the key K as described above, we can now just recompute the encryption of 0^n, again with respect to the same key-nonce pair (K, N) as in the provided encryption oracle, as

$$\begin{aligned}\widetilde{C} &= P\big(\delta(K, X, (0,0,1)) \oplus 0^n\big) \oplus \delta(K, X, (0,0,1)) \\ &= P\big(\varphi^2(\Omega) \oplus \varphi(\Omega) \oplus \Omega\big) \oplus \varphi^2(\Omega) \oplus \varphi(\Omega) \oplus \Omega\end{aligned}$$

where $\Omega = P(X||K)$ and $X = \mathrm{pad}^0_{n-\kappa-k}(N)$. If now $\widetilde{C} = C_0$, i.e. the encryption of the oracle and our manual computation coincide, we can be sure that we recovered the right key K. Else, we can just repeat the attack until the assertion returns to be true.

With the included sanity check, we are certain to recover the key K at some point, as step one of our attack involves an application of Simon's algorithm that already succeeds with high probability thanks to the non-existence of "unwanted periods" as argued before.

4.2.1 Consequences

Key-recovery is already devastating on its own but we still want to discuss some of the implications. After recovering a key K using a single quantum-query – for which we do not need to know the underlying nonce N – we can subsequently encrypt or decrypt arbitrary messages or ciphertexts on our own provided we know the corresponding nonce to be used.

Note that in view of the IND-qCPA, EUF-qCMA and UUF-qCMA security definitions in [MMPR22] the challenger always forwards a randomly chosen nonce to the adversary which it then uses to answer the adversary's quantum encryption queries. Thus, after just a single quantum encryption query – where we do not need to know the random nonce N – to recover the key K, we are able to perform decryption of challenge ciphertexts to break IND-qCPA security or to produce forgeries of arbitrary message and associated data pairs to break both EUF- and UUF-qCMA security. Note, that these attacks also apply in a setting where the nonces are chosen by the challenger *after* the adversary submits the encryption queries. This implies that the adversary can no longer make encryption queries adaptively after gaining knowledge of the the nonce N by choosing the queries depending on said nonce. This makes our attacks even stronger.

4.2.2 Comparison with Quantum Key-Recovery Attack on MEM in [KLLN16]

To conclude the discussion on OPP, we want to mention that there does already exist a quantum key-recovery attack on the Even-Mansour construction first described in [KM12] and further discussed in [KLLN16, Section 3.2]. We want to highlight the novelty of our quantum key-recovery attack on OPP with respect to the existing attacks on Even-Mansour based schemes (as OPP itself uses the Even-Mansour construction). We first give a short recap on the attack as described in [KLLN16].

Starting from the block cipher

$$E_{k_1,k_2}(M) = P(M \oplus k_1) \oplus k_2$$

where P is some public permutation, [KLLN16, Section 3.2] shows that either there exists a classical distinguishing attack (though this case is negligible with random P), or, using Simon's algorithm, we are able to successfully

recover k_1. This is done by applying Simon's algorithm to the periodic function defined by

$$f : \{0,1\}^n \to \{0,1\}^n$$
$$f(M) = E_{k_1,k_2}(M) \oplus P(M) = P(M \oplus k_1) \oplus P(M) \oplus k_2$$

where $f(M \oplus k_1) = f(M)$. As a result, k_1 is recovered in n quantum queries. It is important to highlight that, in this particular case of the *isolated* Even-Mansour construction, both k_1 and k_2 are keys of a fixed value. If we compare this to the setting of OPP, we have that $k_1 = k_2 = \delta(K, X, (i, 0, 1)) = \varphi^{i+2}(\Omega) \oplus \varphi^{i+1}(\Omega) \oplus \varphi^i(\Omega)$ with $\Omega = P(X||K)$ and $X = \text{pad}^0_{n-\kappa-k}(N)$. The OPP setting is more complex as there encryption is dependent on a nonce N, which changes with each call to the encryption oracle. As a result, the value of f, and even more importantly the period k_1 of f, would change in each of the n iterations during an application of Simon's algorithm. This is why we cannot just extend the attack on MEM in [KLLN16] to OPP, as in each application it would recover a vector y orthogonal to a different period. Recall that in our quantum key-recovery attack in Section 4.2 we overcame this issue as we only use one *single* quantum query to the OPP-$\mathcal{E}(K, N, \cdot)$ oracle to compute the periodic function $\tilde{f}_N$ in 4.2.1 in order to recover Ω and thus the key K using Simon's algorithm.

To emphasize, the main difference between our attack and the existing one is twofold. Firstly, we are focusing on the authenticated encryption setting, which includes changing nonces that make the straight forward application of Simon's algorithm infeasible. Secondly, our attack is more efficient in terms of numbers of queries, as it only requires one single quantum encryption query, whereas the existing attack requires n queries.

Bibliography

[ABKM22] Gorjan Alagic, Chen Bai, Jonathan Katz, and Christian Majenz. Post-quantum security of the even-mansour cipher. In Orr Dunkelman and Stefan Dziembowski, editors, *Advances in Cryptology – EUROCRYPT 2022, Part III*, volume 13277 of *Lecture Notes in Computer Science*, pages 458–487, Trondheim, Norway, May 30 – June 3, 2022. Springer, Heidelberg, Germany.

[ACD+22] Gorjan Alagic, David Cooper, Quynh Dang, Thinh Dang, John M. Kelsey, Jacob Lichtinger, Yi-Kai Liu, Carl A. Miller, Dustin Moody, Rene Peralta, Ray Perlner, Angela Robinson, Daniel Smith-Tone, and Daniel Apon. Status report on the third round of the nist post-quantum cryptography standardization process, 2022-07-05 04:07:00 2022.

[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*, pages 44–63, Fukuoka, Japan, February 24–26, 2016. Springer, Heidelberg, Germany.

[BBC+21] Ritam Bhaumik, Xavier Bonnetain, André Chailloux, Gaëtan Leurent, María Naya-Plasencia, André Schrottenloher, and Yannick Seurin. QCB: Efficient quantum-secure authenticated encryption. In Mehdi Tibouchi and Huaxiong Wang, editors, *Advances in Cryptology – ASIACRYPT 2021, Part I*, volume 13090 of *Lecture Notes in Computer Science*, pages 668–698, Singapore, December 6–10, 2021. Springer, Heidelberg, Germany.

[BHN+19] Xavier Bonnetain, Akinori Hosoyamada, María Naya-Plasencia, Yu Sasaki, and André Schrottenloher. Quantum attacks with-

out superposition queries: The offline Simon's algorithm. In Steven D. Galbraith and Shiho Moriai, editors, *Advances in Cryptology – ASIACRYPT 2019, Part I*, volume 11921 of *Lecture Notes in Computer Science*, pages 552–583, Kobe, Japan, December 8–12, 2019. Springer, Heidelberg, Germany.

[BLNS21] Xavier Bonnetain, Gaëtan Leurent, María Naya-Plasencia, and André Schrottenloher. Quantum linearization attacks. In Mehdi Tibouchi and Huaxiong Wang, editors, *Advances in Cryptology – ASIACRYPT 2021, Part I*, volume 13090 of *Lecture Notes in Computer Science*, pages 422–452, Singapore, December 6–10, 2021. Springer, Heidelberg, Germany.

[CAE19] Caesar: Competition for authenticated encryption: Security, applicability, and robustness, 2012-2019. Last accessed 23 March 2023, https://competitions.cr.yp.to/caesar.html.

[Deu85] D. Deutsch. Quantum theory, the Church-Turing principle and the universal quantum computer. *Proceedings of the Royal Society of London Series A*, 400(1818):97–117, July 1985.

[GJMN15] Robert Granger, Philipp Jovanovic, Bart Mennink, and Samuel Neves. Improved masking for tweakable blockciphers with applications to authenticated encryption. Cryptology ePrint Archive, Paper 2015/999, 2015. https://eprint.iacr.org/2015/999.

[GJMN16] Robert Granger, Philipp Jovanovic, Bart Mennink, and Samuel Neves. Improved masking for tweakable blockciphers with applications to authenticated encryption. In Marc Fischlin and Jean-Sébastien Coron, editors, *Advances in Cryptology – EUROCRYPT 2016, Part I*, volume 9665 of *Lecture Notes in Computer Science*, pages 263–293, Vienna, Austria, May 8–12, 2016. Springer, Heidelberg, Germany.

[Gro96] Lov K. Grover. A fast quantum mechanical algorithm for database search. In *Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing*, STOC '96, page 212–219, New York, NY, USA, 1996. Association for Computing Machinery.

[IIMP19] Akiko Inoue, Tetsu Iwata, Kazuhiko Minematsu, and Bertram Poettering. Cryptanalysis of OCB2: Attacks on authenticity and confidentiality. In Alexandra Boldyreva and Daniele Micciancio, editors, *Advances in Cryptology – CRYPTO 2019, Part I*, volume 11692 of *Lecture Notes in Computer Science*, pages 3–31, Santa

Barbara, CA, USA, August 18–22, 2019. Springer, Heidelberg, Germany.

[IM19] Akiko Inoue and Kazuhiko Minematsu. Parallelizable authenticated encryption with small state size. In Kenneth G. Paterson and Douglas Stebila, editors, *SAC 2019: 26th Annual International Workshop on Selected Areas in Cryptography*, volume 11959 of *Lecture Notes in Computer Science*, pages 618–644, Waterloo, ON, Canada, August 12–16, 2019. Springer, Heidelberg, Germany.

[JST21] Joseph Jaeger, Fang Song, and Stefano Tessaro. Quantum key-length extension. In Kobbi Nissim and Brent Waters, editors, *TCC 2021: 19th Theory of Cryptography Conference, Part I*, volume 13042 of *Lecture Notes in Computer Science*, pages 209–239, Raleigh, NC, USA, November 8–11, 2021. Springer, Heidelberg, Germany.

[KLLN16] Marc Kaplan, Gaëtan Leurent, Anthony Leverrier, and María Naya-Plasencia. Breaking symmetric cryptosystems using quantum period finding. In Matthew Robshaw and Jonathan Katz, editors, *Advances in Cryptology – CRYPTO 2016, Part II*, volume 9815 of *Lecture Notes in Computer Science*, pages 207–237, Santa Barbara, CA, USA, August 14–18, 2016. Springer, Heidelberg, Germany.

[KM12] Hidenori Kuwakado and Masakatu Morii. Security on the quantum-type even-mansour cipher. *2012 International Symposium on Information Theory and its Applications*, pages 312–316, 2012.

[KR11] Ted Krovetz and Phillip Rogaway. The software performance of authenticated-encryption modes. In Antoine Joux, editor, *Fast Software Encryption – FSE 2011*, volume 6733 of *Lecture Notes in Computer Science*, pages 306–327, Lyngby, Denmark, February 13–16, 2011. Springer, Heidelberg, Germany.

[LCP22] Xiangru Wang Lipeng Chang, Yuechuan Wei and Xiaozhong Pan. Collision forgery attack on the aes-otr algorithm under quantum computing. *Symmetry*, 2022. https://doi.org/10.3390/sym14071434.

[Mar23] Varun Maram. Private communication, 2023.

[Min14] Kazuhiko Minematsu. Parallelizable rate-1 authenticated encryption from pseudorandom functions. In Phong Q. Nguyen

and Elisabeth Oswald, editors, *Advances in Cryptology – EUROCRYPT 2014*, volume 8441 of *Lecture Notes in Computer Science*, pages 275–292, Copenhagen, Denmark, May 11–15, 2014. Springer, Heidelberg, Germany.

[Min16] Kazuhiko Minematsu. Aes-otr v3.1. Third-Round Candidate Submission to CAESAR Competition, 2016. https://competitions.cr.yp.to/round3/aesotrv31.pdf.

[MMPR22] Varun Maram, Daniel Masny, Sikhar Patranabis, and Srinivasan Raghuraman. On the quantum security of OCB. Cryptology ePrint Archive, Report 2022/699, 2022. https://eprint.iacr.org/2022/699.

[RBBK01] Phillip Rogaway, Mihir Bellare, John Black, and Ted Krovetz. OCB: A block-cipher mode of operation for efficient authenticated encryption. In Michael K. Reiter and Pierangela Samarati, editors, *ACM CCS 2001: 8th Conference on Computer and Communications Security*, pages 196–205, Philadelphia, PA, USA, November 5–8, 2001. ACM Press.

[Rog04] Phillip Rogaway. Efficient instantiations of tweakable blockciphers and refinements to modes OCB and PMAC. In Pil Joong Lee, editor, *Advances in Cryptology – ASIACRYPT 2004*, volume 3329 of *Lecture Notes in Computer Science*, pages 16–31, Jeju Island, Korea, December 5–9, 2004. Springer, Heidelberg, Germany.

[Sho99] Peter W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Review*, 41(2):303–332, 1999.

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

[TRK14] Himanshu Thapliyal, N. Ranganathan, and Saurabh Kotiyal. *Reversible Logic Based Design and Test of Field Coupled Nanocomputing Circuits*, pages 133–172. 06 2014.