On the Quantum Security of OCB

Varun Maram

Applied Cryptography Group

ETH Zurich

Joint work with Daniel Masny, Sikhar Patranabis and Srinivasan Raghuraman

[Full version of paper: *https://eprint.iacr.org/2022/699.pdf*]

Introduction

Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer*

Peter W. Shor†

Abstract

A digital computer is generally believed to be an efficient universal computing device; that is, it is believed able to simulate any physical computing device with an increase in computation time by at most a polynomial factor. This may not be true when quantum mechanics is taken into consideration. This paper considers factoring integers and finding discrete logarithms, two problems which are generally thought to be hard on a classical computer and which have been used as the basis of several proposed cryptosystems. Efficient randomized algorithms are given for these two problems on a hypothetical quantum computer. These algorithms take a number of steps polynomial in the input size, e.g., the number of digits of the integer to be factored.

[FOCS'94]

Introduction

Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer*

Peter W. Shor†

Abstract

A digital computer is generally believed to be an efficient universal computing device; that is, it is believed able to simulate any physical computing device with an increase in computation time by at most a polynomial factor. This may not be true when quantum mechanics is taken into consideration. This paper considers factoring integers and finding discrete logarithms, two problems which are generally thought to be hard on a classical computer and which have been used as the basis of several proposed cryptosystems. Efficient randomized algorithms are given for these two problems on a hypothetical quantum computer. These algorithms take a number of steps polynomial in the input size, e.g., the number of digits of the integer to be factored.

[FOCS'94]

NIST Kicks Off Effort to Defend Encrypted Data from Quantum Computer Threat

April 28, 2016

What will happen to computer security if quantum computers are built? A new NIST publication looks to the road ahead.
Credit: Hanacek/NIST

If an exotic *quantum computer* is invented that could break the codes we depend on to protect confidential electronic information, what will we do to maintain our security and privacy? That's the overarching question posed by a new report from the National Institute of Standards and Technology (NIST), whose cryptography specialists are beginning the long journey toward effective answers.

NIST Internal Report (NISTIR) 8105: *Report on Post-Quantum Cryptography* details the status of research into quantum computers, which would exploit the often counterintuitive world of quantum physics to solve problems that are intractable for conventional computers. If such devices are ever built, they will be able to defeat many of our modern cryptographic systems, such as the computer algorithms used to protect online bank transactions. NISTIR 8105 outlines a long-term approach for avoiding this vulnerability before it arises.

MEDIA CONTACT
Chad Boutin
charles.boutin@nist.gov
(301) 975-4261

ORGANIZATIONS
Information Technology Laboratory
Computer Security Division

SIGN UP FOR UPDATES FROM NIST

Enter Email Address

"It will be a long process involving public vetting of quantum-resistant algorithms," Moody said. "And we're not expecting to have just one winner. There are several systems in use that could be broken by a quantum computer—public-key encryption and digital signatures, to take two examples—and we will need different solutions for each of those systems."

Introduction

Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer*

Peter W. Shor†

Abstract

A digital computer is generally believed to be an efficient universal computing device; that is, it is believed able to simulate any physical computing device with an increase in computation time by at most a polynomial factor. This may not be true when quantum mechanics is taken into consideration. This paper considers factoring integers and finding discrete logarithms, two problems which are generally thought to be hard on a classical computer and which have been used as the basis of several proposed cryptosystems. Efficient randomized algorithms are given for these two problems on a hypothetical quantum computer. These algorithms take a number of steps polynomial in the input size, e.g., the number of digits of the integer to be factored.

[FOCS'94]

Symmetric-key crypto?

NIST Kicks Off Effort to Defend Encrypted Data from Quantum Computer Threat

April 28, 2016

What will happen to computer security if quantum computers are built? A new NIST publication looks to the road ahead.
Credit: Hanacek/NIST

If an exotic *quantum computer* is invented that could break the codes we depend on to protect confidential electronic information, what will we do to maintain our security and privacy? That's the overarching question posed by a new report from the National Institute of Standards and Technology (NIST), whose cryptography specialists are beginning the long journey toward effective answers.

NIST Internal Report (NISTIR) 8105: *Report on Post-Quantum Cryptography* details the status of research into quantum computers, which would exploit the often counterintuitive world of quantum physics to solve problems that are intractable for conventional computers. If such devices are ever built, they will be able to defeat many of our modern cryptographic systems, such as the computer algorithms used to protect online bank transactions. NISTIR 8105 outlines a long-term approach for avoiding this vulnerability before it arises.

MEDIA CONTACT

Chad Boutin
charles.boutin@nist.gov
(301) 975-4261

ORGANIZATIONS

Information Technology Laboratory
Computer Security Division

SIGN UP FOR UPDATES FROM NIST

Enter Email Address

"It will be a long process involving public vetting of quantum-resistant algorithms," Moody said. "And we're not expecting to have just one winner. There are several systems in use that could be broken by a quantum computer—public-key encryption and digital signatures, to take two examples—and we will need different solutions for each of those systems."

Introduction

A fast quantum mechanical algorithm for database search

Lov K. Grover
3C-404A, Bell Labs
600 Mountain Avenue
Murray Hill NJ 07974
lkgrover@bell-labs.com

Summary

Imagine a phone directory containing N names arranged in completely random order. In order to find someone's phone number with a probability of $\frac{1}{2}$, any classical algorithm (whether deterministic or probabilistic) will need to look at a minimum of $\frac{N}{2}$ names. Quantum mechanical systems can be in a superposition of states and simultaneously examine multiple names. By properly adjusting the phases of various operations, successful computations reinforce each other while others interfere randomly. As a result, the desired phone number can be obtained in only $O(\sqrt{N})$ steps. The algorithm is within a small constant factor of the fastest possible quantum mechanical algorithm.

This paper applies quantum computing to a mundane problem in information processing and presents an algorithm that is significantly faster than any classical algorithm can be. The problem is this: there is an unsorted database containing N items out of which just one item satisfies a given condition - that one item has to be retrieved. Once an item is examined, it is possible to tell whether or not it satisfies the condition in one step. However, there does not exist any sorting on the database that would aid its selection. The most efficient classical algorithm for this is to examine the items in the database one by one. If an item satisfies the required condition stop; if it does not, keep track of this item so that it is not examined again. It is easily seen that this algorithm will need to look at an average of $\frac{N}{2}$ items before finding the desired item.

[STOC'96]

Introduction

A fast quantum mechanical algorithm for database search

Lov K. Grover
3C-404A, Bell Labs
600 Mountain Avenue
Murray Hill NJ 07974
lkgrover@bell-labs.com

Summary

Imagine a phone directory containing N names arranged in completely random order. In order to find someone's phone number with a probability of $\frac{1}{2}$, any classical algorithm (whether deterministic or probabilistic) will need to look at a minimum of $\frac{N}{2}$ names. Quantum mechanical systems can be in a superposition of states and simultaneously examine multiple names. By properly adjusting the phases of various operations, successful computations reinforce each other while others interfere randomly. As a result, the desired phone number can be obtained in only $O(\sqrt{N})$ steps. The algorithm is within a small constant factor of the fastest possible quantum mechanical algorithm.

This paper applies quantum computing to a mundane problem in information processing and presents an algorithm that is significantly faster than any classical algorithm can be. The problem is this: there is an unsorted database containing N items out of which just one item satisfies a given condition - that one item has to be retrieved. Once an item is examined, it is possible to tell whether or not it satisfies the condition in one step. However, there does not exist any sorting on the database that would aid its selection. The most efficient classical algorithm for this is to examine the items in the database one by one. If an item satisfies the required condition stop; if it does not, keep track of this item so that it is not examined again. It is easily seen that this algorithm will need to look at an average of $\frac{N}{2}$ items before finding the desired item.

[STOC'96]

Just double the key-length?

Introduction

Polynomial-time
“superposition” attacks

Quantum Distinguisher Between the 3-Round Feistel Cipher and the Random Permutation

Hidenori Kuwakado
Graduate School of Engineering
Kobe University
1-1 Rokkodai-cho Nada-ku Kobe 657-8501, Japan

Masakatu Morii
Graduate School of Engineering
Kobe University
1-1 Rokkodai-cho Nada-ku Kobe 657-8501, Japan

[ISIT’10]

Introduction

Polynomial-time
“superposition” attacks

Quantum Distinguisher Between the 3-Round Feistel Cipher and the Random Permutation

Hidenori Kuwakado
Graduate School of Engineering
Kobe University
1-1 Rokkodai-cho Nada-ku Kobe 657-8501, Japan

Masakatu Morii
Graduate School of Engineering
Kobe University
1-1 Rokkodai-cho Nada-ku Kobe 657-8501, Japan

[ISIT’10]

Security on the Quantum-type Even-Mansour Cipher

Hidenori Kuwakado
Graduate School of Engineering
Kobe University
1-1 Rokkodai-cho Nada-ku Kobe 657-8501, Japan
Email: kuwakado@kobe-u.ac.jp

Masakatu Morii
Graduate School of Engineering
Kobe University
1-1 Rokkodai-cho Nada-ku Kobe 657-8501, Japan
Email: mmorii@kobe-u.ac.jp

[ISITA’12]

Breaking Symmetric Cryptosystems using Quantum Period Finding

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

[1] LTCI, Télécom ParisTech, 23 avenue d’Italie, 75214 Paris CEDEX 13, France
[2] School of Informatics, University of Edinburgh, 10 Crichton Street, Edinburgh EH8 9AB, UK
[3] Inria Paris, France

[CRYPTO’16]

USING SIMON’S ALGORITHM TO ATTACK SYMMETRIC-KEY CRYPTOGRAPHIC PRIMITIVES

THOMAS SANTOLI[a]
Mathematical Institute, University of Oxford; Andrew Wiles Building, Radcliffe Observatory Quarter Woodstock Road, Oxford, OX2 6GG

CHRISTIAN SCHAFFNER[b]
ILLC, University of Amsterdam
Centrum Wiskunde & Informatica (CWI)
QuSoft
P.O. Box 94242, 1090 GE Amsterdam, Netherlands

[QI&C’17]

Quantum Attacks without Superposition Queries: the Offline Simon’s Algorithm

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

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

[ASIACRYPT’19]

Simon's Algorithm

ON THE POWER OF QUANTUM COMPUTATION

DANIEL R. SIMON
MICROSOFT CORP.
ONE MICROSOFT WAY
REDMOND WA 98052-6399

DANSIMON@MICROSOFT.COM

Abstract. The quantum model of computation is a model, analogous to the probabilistic Turing Machine, in which the normal laws of chance are replaced by those obeyed by particles on a quantum mechanical scale, rather than the rules familiar to us from the macroscopic world. We present here a problem of distinguishing between two fairly natural classes of function, which can provably be solved exponentially faster in the quantum model than in the classical probabilistic one, when the function is given as an oracle drawn equiprobably from the uniform distribution on either class. We thus offer compelling evidence that the quantum model may have significantly more complexity theoretic power than the probabilistic Turing Machine. In fact, drawing on this work, Shor has recently developed remarkable new quantum polynomial-time algorithms for the discrete logarithm and integer factoring problems.

[SIAM JoC'97]

$f: \{0,1\}^n \rightarrow \{0,1\}^n$

$\exists s \in \{0,1\}^n$ s.t.
$f(x) = f(x \oplus s)\ \forall x$

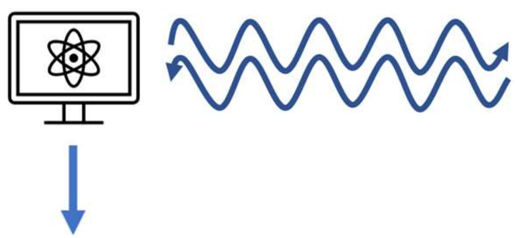
Simon's Algorithm

ON THE POWER OF QUANTUM COMPUTATION

DANIEL R. SIMON
MICROSOFT CORP.
ONE MICROSOFT WAY
REDMOND WA 98052-6399
DANSIMON@MICROSOFT.COM

Abstract. The quantum model of computation is a model, analogous to the probabilistic Turing Machine, in which the normal laws of chance are replaced by those obeyed by particles on a quantum mechanical scale, rather than the rules familiar to us from the macroscopic world. We present here a problem of distinguishing between two fairly natural classes of function, which can provably be solved exponentially faster in the quantum model than in the classical probabilistic one, when the function is given as an oracle drawn equiprobably from the uniform distribution on either class. We thus offer compelling evidence that the quantum model may have significantly more complexity theoretic power than the probabilistic Turing Machine. In fact, drawing on this work, Shor has recently developed remarkable new quantum polynomial-time algorithms for the discrete logarithm and integer factoring problems.

[SIAM JoC'97]



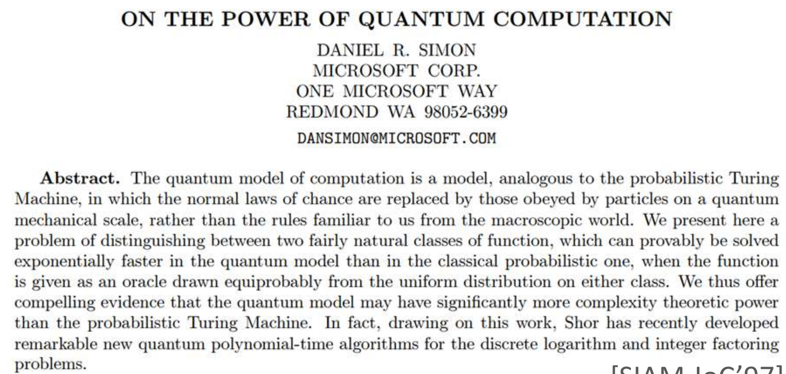
$f: \{0,1\}^n \rightarrow \{0,1\}^n$

$\exists s \in \{0,1\}^n$ s.t.
$f(x) = f(x \oplus s)\ \forall x$

s

(in $O(n)$ queries)

Simon's Algorithm



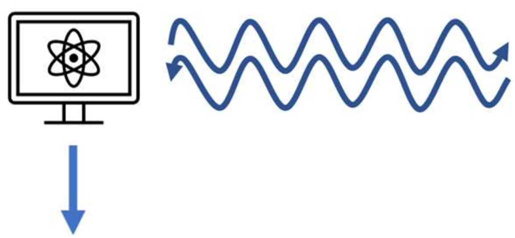
ON THE POWER OF QUANTUM COMPUTATION

DANIEL R. SIMON
MICROSOFT CORP.
ONE MICROSOFT WAY
REDMOND WA 98052-6399

DANSIMON@MICROSOFT.COM

Abstract. The quantum model of computation is a model, analogous to the probabilistic Turing Machine, in which the normal laws of chance are replaced by those obeyed by particles on a quantum mechanical scale, rather than the rules familiar to us from the macroscopic world. We present here a problem of distinguishing between two fairly natural classes of function, which can provably be solved exponentially faster in the quantum model than in the classical probabilistic one, when the function is given as an oracle drawn equiprobably from the uniform distribution on either class. We thus offer compelling evidence that the quantum model may have significantly more complexity theoretic power than the probabilistic Turing Machine. In fact, drawing on this work, Shor has recently developed remarkable new quantum polynomial-time algorithms for the discrete logarithm and integer factoring problems.

[SIAM JoC'97]



$f: \{0,1\}^n \rightarrow \{0,1\}^n$

$\exists s \in \{0,1\}^n$ s.t.
$f(x) = f(x \oplus s)\ \forall x$

s
(in $O(n)$ queries)

Each iteration returns an independent y orthogonal to s w.h.p.

Quantum Attacks Against Symmetric Crypto

Breaking Symmetric Cryptosystems using Quantum Period Finding

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

[1] LTCI, Télécom ParisTech, 23 avenue d'Italie, 75214 Paris CEDEX 13, France
[2] School of Informatics, University of Edinburgh, 10 Crichton Street, Edinburgh EH8 9AB, UK
[3] Inria Paris, France

[CRYPTO'16]

We obtain attacks with very strong implications. First, we show that the most widely used modes of operation for authentication and authenticated encryption (*e.g.* CBC-MAC, PMAC, GMAC, GCM, and OCB) are completely broken in this security model. Our attacks are also appli-

OCB Mode of Authenticated Encryption

- Is a popular AE mode of block-cipher operation with a very high efficiency.
  - Requires l block-cipher calls to process an l-block message; is parallelizable.

OCB Mode of Authenticated Encryption

- Is a popular AE mode of block-cipher operation with a very high efficiency.
  - Requires l block-cipher calls to process an l-block message; is parallelizable.

- Has three versions:
  - OCB1 [Rogaway et. al., CCS'01] is listed in IEEE 802.11 standard as an option for protecting wireless networks.

OCB Mode of Authenticated Encryption

- Is a popular AE mode of block-cipher operation with a very high efficiency.
  - Requires l block-cipher calls to process an l-block message; is parallelizable.

- Has three versions:
  - OCB1 [Rogaway et. al., CCS'01] is listed in IEEE 802.11 standard as an option for protecting wireless networks.
  - OCB2 [Rogaway, ASIACRYPT'04] was in the ISO/IEC 19772:2009 standard.
    - It was later shown to be insecure by Inoue et. al. [CRYPTO'19].

OCB Mode of Authenticated Encryption

- Is a popular AE mode of block-cipher operation with a very high efficiency.
  - Requires l block-cipher calls to process an l-block message; is parallelizable.

- Has three versions:
  - OCB1 [Rogaway et. al., CCS'01] is listed in IEEE 802.11 standard as an option for protecting wireless networks.
  - OCB2 [Rogaway, ASIACRYPT'04] was in the ISO/IEC 19772:2009 standard.
    - It was later shown to be insecure by Inoue et. al. [CRYPTO'19].
  - OCB3 [Krovetz and Rogaway, FSE'11] is specified in RFC 7253 as an IETF Internet standard; is in the final portfolio of CAESAR competition.

OCB Encryption Algorithm

1) ***Initialization*****:** The initialization stage completes two tasks, partition of the message M into blocks $M_1 \cdots M_m$, where all but the last block are full, and calculation of the initial offset Δ_0.

- In OCB1: $\Delta_0 = E_K(N \oplus L)$, where $L = E_K(0^{128})$.
- In OCB2: $\Delta_0 = E_K(N)$.
- In OCB3: $\Delta_0 = H_K(N)$, where H is a universal hash function.

2) ***Ciphertext Generation*****:** During this stage, the plaintext blocks are encrypted to get ciphertext blocks along with offsets updated.

$$C_i \leftarrow E_K(M_i \oplus \Delta_i) \oplus \Delta_i,\ i = 1, \cdots, m-1.$$

The offset Δ_i can be easily updated from previous Δ_{i-1}.

- In OCB1, $\Delta_i = \Delta_0 \oplus \gamma_i \cdot L = \Delta_{i-1} \oplus 2^{ntz(i)} \cdot L$, where γ_i is the ith element of the Gray code, $L = E_K(0^n)$ and $\Delta_0 = E_K(N \oplus L)$.
- In OCB2, $\Delta_i = 2^i \cdot \Delta_0 = 2 \cdot \Delta_{i-1}$.
- In OCB3, $\Delta_i = \Delta_0 \oplus 4 \cdot \gamma_i \cdot L = \Delta_{i-1} \oplus 2^{2+ntz(i)} \cdot L$.

3) ***Tag Generation*****:** In this stage, Checksum is calculated, and then encrypted into Tag:

$$\text{Checksum} \leftarrow M_1 \oplus \cdots \oplus M_{m-1} \oplus g_K(M_m),$$
$$\text{Tag} \leftarrow E_K(\text{Checksum} \oplus \Delta_*) \oplus h_K(A).$$

OCB Encryption Algorithm

1) ***Initialization*****:** The initialization stage completes two tasks, partition of the message M into blocks $M_1 \cdots M_m$, where all but the last block are full, and calculation of the initial offset Δ_0.

- In OCB1: $\Delta_0 = E_K(N \oplus L)$, where $L = E_K(0^{128})$.
- In OCB2: $\Delta_0 = E_K(N)$.
- In OCB3: $\Delta_0 = H_K(N)$, where H is a universal hash function.

2) ***Ciphertext Generation*****:** During this stage, the plaintext blocks are encrypted to get ciphertext blocks along with offsets updated.

$$C_i \leftarrow E_K(M_i \oplus \Delta_i) \oplus \Delta_i,\ i = 1, \cdots, m-1.$$

The offset Δ_i can be easily updated from previous Δ_{i-1}.

- In OCB1, $\Delta_i = \Delta_0 \oplus \gamma_i \cdot L = \Delta_{i-1} \oplus 2^{ntz(i)} \cdot L$, where γ_i is the ith element of the Gray code, $L = E_K(0^n)$ and $\Delta_0 = E_K(N \oplus L)$.
- In OCB2, $\Delta_i = 2^i \cdot \Delta_0 = 2 \cdot \Delta_{i-1}$.
- In OCB3, $\Delta_i = \Delta_0 \oplus 4 \cdot \gamma_i \cdot L = \Delta_{i-1} \oplus 2^{2+ntz(i)} \cdot L$.

3) ***Tag Generation*****:** In this stage, Checksum is calculated, and then encrypted into Tag:

$$\text{Checksum} \leftarrow M_1 \oplus \cdots \oplus M_{m-1} \oplus g_K(M_m),$$
$$\text{Tag} \leftarrow E_K(\text{Checksum} \oplus \Delta_*) \oplus h_K(A).$$

OCB Encryption Algorithm

1) ***Initialization*****:** The initialization stage completes two tasks, partition of the message M into blocks $M_1 \cdots M_m$, where all but the last block are full, and calculation of the initial offset Δ_0.

- In OCB1: $\Delta_0 = E_K(N \oplus L)$, where $L = E_K(0^{128})$.
- In OCB2: $\Delta_0 = E_K(N)$.
- In OCB3: $\Delta_0 = H_K(N)$, where H is a universal hash function.

2) ***Ciphertext Generation*****:** During this stage, the plaintext blocks are encrypted to get ciphertext blocks along with offsets updated.

$$C_i \leftarrow E_K(M_i \oplus \Delta_i) \oplus \Delta_i,\ i = 1, \cdots, m-1.$$

The offset Δ_i can be easily updated from previous Δ_{i-1}.

- In OCB1, $\Delta_i = \Delta_0 \oplus \gamma_i \cdot L = \Delta_{i-1} \oplus 2^{ntz(i)} \cdot L$, where γ_i is the ith element of the Gray code, $L = E_K(0^n)$ and $\Delta_0 = E_K(N \oplus L)$.
- In OCB2, $\Delta_i = 2^i \cdot \Delta_0 = 2 \cdot \Delta_{i-1}$.
- In OCB3, $\Delta_i = \Delta_0 \oplus 4 \cdot \gamma_i \cdot L = \Delta_{i-1} \oplus 2^{2+ntz(i)} \cdot L$.

3) ***Tag Generation*****:** In this stage, Checksum is calculated, and then encrypted into Tag:

$$\text{Checksum} \leftarrow M_1 \oplus \cdots \oplus M_{m-1} \oplus g_K(M_m),$$
$$\text{Tag} \leftarrow E_K(\text{Checksum} \oplus \Delta_*) \oplus h_K(A).$$

OCB Encryption Algorithm

1) ***Initialization*****:** The initialization stage completes two tasks, partition of the message M into blocks $M_1 \cdots M_m$, where all but the last block are full, and calculation of the initial offset Δ_0.

- In OCB1: $\Delta_0 = E_K(N \oplus L)$, where $L = E_K(0^{128})$.
- In OCB2: $\Delta_0 = E_K(N)$.
- In OCB3: $\Delta_0 = H_K(N)$, where H is a universal hash function.

2) ***Ciphertext Generation*****:** During this stage, the plaintext blocks are encrypted to get ciphertext blocks along with offsets updated.

$$C_i \leftarrow E_K(M_i \oplus \Delta_i) \oplus \Delta_i, \ i = 1, \cdots, m-1.$$

The offset Δ_i can be easily updated from previous Δ_{i-1}.

- In OCB1, $\Delta_i = \Delta_0 \oplus \gamma_i \cdot L = \Delta_{i-1} \oplus 2^{ntz(i)} \cdot L$, where γ_i is the ith element of the Gray code, $L = E_K(0^n)$ and $\Delta_0 = E_K(N \oplus L)$.
- In OCB2, $\Delta_i = 2^i \cdot \Delta_0 = 2 \cdot \Delta_{i-1}$.
- In OCB3, $\Delta_i = \Delta_0 \oplus 4 \cdot \gamma_i \cdot L = \Delta_{i-1} \oplus 2^{2+ntz(i)} \cdot L$.

3) ***Tag Generation*****:** In this stage, Checksum is calculated, and then encrypted into Tag:

$$\text{Checksum} \leftarrow M_1 \oplus \cdots \oplus M_{m-1} \oplus g_K(M_m),$$
$$\text{Tag} \leftarrow E_K(\text{Checksum} \oplus \Delta_*) \oplus h_K(A).$$

Quantum Attacks on Integrity: OCB1/OCB3

$$
\begin{aligned}
f_{\dot{N}} \colon \{0,1\}^n &\to \{0,1\}^n \\
m \quad &\mapsto c_1 \oplus c_2, \text{where } (c_1, c_2, \tau) = \mathrm{OCB}_k(N, m \parallel m, \varepsilon) \\
f_{\dot{N}}(m) = E_k(m \oplus \Delta_1^N) &\oplus \Delta_1^N \oplus E_k(m \oplus \Delta_2^N) \oplus \Delta_2^N
\end{aligned}
$$

$$c_i = E_k(m_i \oplus \Delta_i^N) \oplus \Delta_i^N$$

(Following is the description of an attack by Kaplan et. al. [CRYPTO'16].)

Quantum Attacks on Integrity: OCB1/OCB3

$$f_{\dot{N}}: \{0,1\}^n \to \{0,1\}^n$$
$$m \mapsto c_1 \oplus c_2, \text{where } (c_1, c_2, \tau) = \mathrm{OCB}_k(N, m \parallel m, \varepsilon)$$
$$f_{\dot{N}}(m) = E_k(m \oplus \Delta_1^N) \oplus \Delta_1^N \oplus E_k(m \oplus \Delta_2^N) \oplus \Delta_2^N$$

$$c_i = E_k(m_i \oplus \Delta_i^N) \oplus \Delta_i^N$$

(Following is the description of an attack by Kaplan et. al. [CRYPTO'16].)

- Function satisfies $f_N(m \oplus \Delta_1^N \oplus \Delta_2^N) = f_N(m)$.

Quantum Attacks on Integrity: OCB1/OCB3

$$f_N\colon \{0,1\}^n \to \{0,1\}^n$$
$$m \mapsto c_1 \oplus c_2, \text{ where } (c_1, c_2, \tau) = \mathrm{OCB}_k(N, m \,\|\, m, \varepsilon)$$
$$f_N(m) = E_k(m \oplus \Delta_1^N) \oplus \Delta_1^N \oplus E_k(m \oplus \Delta_2^N) \oplus \Delta_2^N$$

$$c_i = E_k(m_i \oplus \Delta_i^N) \oplus \Delta_i^N$$

(Following is the description of an attack by Kaplan et. al. [CRYPTO'16].)

- Function satisfies $f_N\left(m \oplus \Delta_1^N \oplus \Delta_2^N\right) = f_N(m)$.
- In OCB1 and OCB3, $\Delta_1^N \oplus \Delta_2^N$ is independent of nonce N. E.g., in OCB3, period of f_N is $\Delta_1^N \oplus \Delta_2^N = 4(\gamma_1 \oplus \gamma_2)E_k(0^n)$.

Quantum Attacks on Integrity: OCB1/OCB3

$$f_N\colon \{0,1\}^n \to \{0,1\}^n$$
$$m \mapsto c_1 \oplus c_2, \text{ where } (c_1, c_2, \tau) = \mathrm{OCB}_k(N, m \parallel m, \varepsilon)$$
$$f_N(m) = E_k(m \oplus \Delta_1^N) \oplus \Delta_1^N \oplus E_k(m \oplus \Delta_2^N) \oplus \Delta_2^N$$

$$c_i = E_k(m_i \oplus \Delta_i^N) \oplus \Delta_i^N$$

(Following is the description of an attack by Kaplan et. al. [CRYPTO'16].)

- Function satisfies $f_N\left(m \oplus \Delta_1^N \oplus \Delta_2^N\right) = f_N(m)$.
- In OCB1 and OCB3, $\Delta_1^N \oplus \Delta_2^N$ is independent of nonce N. E.g., in OCB3, period of f_N is $\Delta_1^N \oplus \Delta_2^N = 4(\gamma_1 \oplus \gamma_2)E_k(0^n)$.
- Can apply Simon's algorithm w.r.t. f_N to recover $\Delta_1^N \oplus \Delta_2^N$.

Quantum Attacks on Integrity: OCB1/OCB3

$$f_N : \{0,1\}^n \to \{0,1\}^n$$
$$m \mapsto c_1 \oplus c_2, \text{where } (c_1, c_2, \tau) = \mathrm{OCB}_k(N, m \parallel m, \varepsilon)$$
$$f_N(m) = E_k(m \oplus \Delta_1^N) \oplus \Delta_1^N \oplus E_k(m \oplus \Delta_2^N) \oplus \Delta_2^N$$

$$c_i = E_k(m_i \oplus \Delta_i^N) \oplus \Delta_i^N$$

(Following is the description of an attack by Kaplan et. al. [CRYPTO'16].)

- Function satisfies $f_N(m \oplus \Delta_1^N \oplus \Delta_2^N) = f_N(m)$.
- In OCB1 and OCB3, $\Delta_1^N \oplus \Delta_2^N$ is independent of nonce N. E.g., in OCB3, period of f_N is $\Delta_1^N \oplus \Delta_2^N = 4(\gamma_1 \oplus \gamma_2)E_k(0^n)$.
- Can apply Simon's algorithm w.r.t. f_N to recover $\Delta_1^N \oplus \Delta_2^N$.

- Existential forgery: Under a random nonce N, if $\mathrm{OCB}_k(N, m \| m, A) = (c_1, c_2, \tau)$, then $((c_2 \oplus \Delta_1^N \oplus \Delta_2^N), (c_1 \oplus \Delta_1^N \oplus \Delta_2^N), \tau) = \mathrm{OCB}_k(N, (m \oplus \Delta_1^N \oplus \Delta_2^N) \| (m \oplus \Delta_1^N \oplus \Delta_2^N), A)$.

Quantum Attacks on Confidentiality: OCB1/OCB3

- We extended the previous attacks to show OCB1 and OCB3 are insecure in the "IND-qCPA" sense – even when the nonces are hidden and random.
  - (In a similar spirit to the "classical" break of OCB2.)

Quantum Attacks on Confidentiality: OCB1/OCB3

- We extended the previous attacks to show OCB1 and OCB3 are insecure in the "IND-qCPA" sense – even when the nonces are hidden and random.
  - (In a similar spirit to the "classical" break of OCB2.)

- IND-qCPA [Boneh and Zhandry, CRYPTO'13]: Extension of IND-CPA notion in a quantum setting.

Quantum Attacks on Confidentiality: OCB1/OCB3

- We extended the previous attacks to show OCB1 and OCB3 are insecure in the "IND-qCPA" sense – even when the nonces are **hidden** and random.
  - (In a similar spirit to the "classical" break of OCB2.)

- IND-qCPA [Boneh and Zhandry, CRYPTO'13]: Extension of IND-CPA notion in a quantum setting.
  - Attacker can ask for encryption of messages in superposition.
  - However, in challenge phase, attacker should forward two classical messages.

Quantum Attacks on Confidentiality: OCB1/OCB3

- We extended the previous attacks to show OCB1 and OCB3 are insecure in the "IND-qCPA" sense – even when the nonces are **hidden** and random.
  - (In a similar spirit to the "classical" break of OCB2.)

- IND-qCPA [Boneh and Zhandry, CRYPTO'13]: Extension of IND-CPA notion in a quantum setting.
  - Attacker can ask for encryption of messages in superposition.
  - However, in challenge phase, attacker should forward two classical messages.

- Our attacks exploit the fact that the last block of messages are encrypted differently, compared to other blocks, in OCB.

IND-qCPA Insecurity of OCB1

```
Partition M into M_1 ··· M_m
```

$L \leftarrow E_K(0^n)$

$\Delta_0 \leftarrow E_K(N \oplus L)$

$\texttt{Checksum} \leftarrow 0^n$

$\texttt{for } i = 1 \texttt{ to } m \texttt{ do}$

$\quad \Delta_i \leftarrow \gamma_i \cdot L \oplus \Delta_0$

$\texttt{for } i = 1 \texttt{ to } m-1 \texttt{ do}$

$\quad C_i \leftarrow E_K(M_i \oplus \Delta_i) \oplus \Delta_i$

$\quad \texttt{Checksum} \leftarrow \texttt{Checksum} \oplus M_i$

$X_m \leftarrow len(M_m) \oplus L \cdot 2^{-1} \oplus \Delta_m$

$Y_m \leftarrow E_K(X_m)$

$C_m \leftarrow Y_m \oplus M_m$

$\texttt{Checksum} \leftarrow \texttt{Checksum} \oplus C_m 0^* \oplus Y_m$

$Tag = E_K(\texttt{Checksum} \oplus \Delta_m)$

IND-qCPA Insecurity of OCB1

$$
\begin{aligned}
&\texttt{Partition } M \texttt{ into } M_1 \cdots M_m \\
&L \leftarrow E_K(0^n) \\
&\Delta_0 \leftarrow E_K(N \oplus L) \\
&\texttt{Checksum} \leftarrow 0^n \\
&\texttt{for } i = 1 \texttt{ to } m \texttt{ do} \\
&\quad \Delta_i \leftarrow \gamma_i \cdot L \oplus \Delta_0 \\
&\texttt{for } i = 1 \texttt{ to } m-1 \texttt{ do} \\
&\quad C_i \leftarrow E_K(M_i \oplus \Delta_i) \oplus \Delta_i \\
&\quad \texttt{Checksum} \leftarrow \texttt{Checksum} \oplus M_i \\
&X_m \leftarrow len(M_m) \oplus L \cdot 2^{-1} \oplus \Delta_m \\
&Y_m \leftarrow E_K(X_m) \\
&C_m \leftarrow Y_m \oplus M_m \\
&\texttt{Checksum} \leftarrow \texttt{Checksum} \oplus C_m 0^* \oplus Y_m \\
&Tag = E_K(\texttt{Checksum} \oplus \Delta_m)
\end{aligned}
$$

IND-qCPA attack:

- Quantum phase: Use Simon's algorithm to recover $L = E_k(0^n)$, as seen w.r.t. existential forgery of OCB1.

IND-qCPA Insecurity of OCB1

$$
\begin{aligned}
&\texttt{Partition } M \texttt{ into } M_1 \cdots M_m \\
&L \leftarrow E_K(0^n) \\
&\Delta_0 \leftarrow E_K(N \oplus L) \\
&\texttt{Checksum} \leftarrow 0^n \\
&\texttt{for } i = 1 \texttt{ to } m \texttt{ do} \\
&\quad \Delta_i \leftarrow \gamma_i \cdot L \oplus \Delta_0 \\
&\texttt{for } i = 1 \texttt{ to } m-1 \texttt{ do} \\
&\quad C_i \leftarrow E_K(M_i \oplus \Delta_i) \oplus \Delta_i \\
&\quad \texttt{Checksum} \leftarrow \texttt{Checksum} \oplus M_i \\
&X_m \leftarrow len(M_m) \oplus L \cdot 2^{-1} \oplus \Delta_m \\
&Y_m \leftarrow E_K(X_m) \\
&C_m \leftarrow Y_m \oplus M_m \\
&\texttt{Checksum} \leftarrow \texttt{Checksum} \oplus C_m 0^* \oplus Y_m \\
&Tag = E_K(\texttt{Checksum} \oplus \Delta_m)
\end{aligned}
$$

IND-qCPA attack:

- Quantum phase: Use Simon's algorithm to recover $L = E_k(0^n)$, as seen w.r.t. existential forgery of OCB1.
- Classical phase: Pick 1-block messages $M_0^* = n \oplus L \cdot 2^{-1}$ and arbitrary $M_1^* \neq M_0^*$.

IND-qCPA Insecurity of OCB1

$$
\begin{aligned}
&\texttt{Partition } M \texttt{ into } M_1 \cdots M_m \\
&L \leftarrow E_K(0^n) \\
&\Delta_0 \leftarrow E_K(N \oplus L) \\
&\texttt{Checksum} \leftarrow 0^n \\
&\texttt{for } i = 1 \texttt{ to } m \texttt{ do} \\
&\quad \Delta_i \leftarrow \gamma_i \cdot L \oplus \Delta_0 \\
&\texttt{for } i = 1 \texttt{ to } m-1 \texttt{ do} \\
&\quad C_i \leftarrow E_K(M_i \oplus \Delta_i) \oplus \Delta_i \\
&\quad \texttt{Checksum} \leftarrow \texttt{Checksum} \oplus M_i \\
&X_m \leftarrow len(M_m) \oplus L \cdot 2^{-1} \oplus \Delta_m \\
&Y_m \leftarrow E_K(X_m) \\
&C_m \leftarrow Y_m \oplus M_m \\
&\texttt{Checksum} \leftarrow \texttt{Checksum} \oplus C_m 0^* \oplus Y_m \\
&Tag = E_K(\texttt{Checksum} \oplus \Delta_m)
\end{aligned}
$$

IND-qCPA attack:

- Quantum phase: Use Simon's algorithm to recover $L = E_k(0^n)$, as seen w.r.t. existential forgery of OCB1.
- Classical phase: Pick 1-block messages $M_0^* = n \oplus L \cdot 2^{-1}$ and arbitrary $M_1^* \neq M_0^*$.
- Upon receiving (C, Tag), return $b = 0$ iff $C \oplus Tag = M_0^*$.

IND-qCPA Insecurity of OCB1

```
Partition M into M_1 ··· M_m
L ← E_K(0^n)
Δ_0 ← E_K(N ⊕ L)
Checksum ← 0^n
for i = 1 to m do
    Δ_i ← γ_i · L ⊕ Δ_0
for i = 1 to m − 1 do
    C_i ← E_K(M_i ⊕ Δ_i) ⊕ Δ_i
    Checksum ← Checksum ⊕M_i
X_m ← len(M_m) ⊕ L · 2^{−1} ⊕ Δ_m
Y_m ← E_K(X_m)
C_m ← Y_m ⊕ M_m
Checksum ← Checksum ⊕C_m0^* ⊕ Y_m
Tag = E_K(Checksum ⊕Δ_m)
```

IND-qCPA attack:

- Quantum phase: Use Simon's algorithm to recover $L = E_k(0^n)$, as seen w.r.t. existential forgery of OCB1.
- Classical phase: Pick 1-block messages $M_0^* = n \oplus L \cdot 2^{-1}$ and arbitrary $M_1^* \neq M_0^*$.
- Upon receiving (C, Tag), return $b = 0$ iff $C \oplus Tag = M_0^*$.
- Note that if M_0^* was encrypted, $X_m = M_0^* \oplus \Delta_m = \mathtt{Checksum} \oplus \Delta_m$. Hence, $Y_m = Tag$.

IND-qCPA Insecurity of OCB1

Partition M into $M_1 \cdots M_m$
$L \leftarrow E_K(0^n)$
$\Delta_0 \leftarrow E_K(N \oplus L)$
Checksum $\leftarrow 0^n$
for $i = 1$ to m do
  $\Delta_i \leftarrow \gamma_i \cdot L \oplus \Delta_0$
for $i = 1$ to $m-1$ do
  $C_i \leftarrow E_K(M_i \oplus \Delta_i) \oplus \Delta_i$
  Checksum $\leftarrow$ Checksum $\oplus M_i$
$X_m \leftarrow len(M_m) \oplus L \cdot 2^{-1} \oplus \Delta_m$
$Y_m \leftarrow E_K(X_m)$
$C_m \leftarrow Y_m \oplus M_m$
Checksum $\leftarrow$ Checksum $\oplus C_m 0^* \oplus Y_m$
$Tag = E_K$(Checksum $\oplus \Delta_m$)

IND-qCPA attack:

- Quantum phase: Use Simon's algorithm to recover $L = E_k(0^n)$, as seen w.r.t. existential forgery of OCB1.
- Classical phase: Pick 1-block messages $M_0^* = n \oplus L \cdot 2^{-1}$ and arbitrary $M_1^* \neq M_0^*$.
- Upon receiving (C, Tag), return $b = 0$ iff $C \oplus Tag = M_0^*$.
- Note that if M_0^* was encrypted, $X_m = M_0^* \oplus \Delta_m =$ Checksum $\oplus \Delta_m$. Hence, $Y_m = Tag$.
- Also, attacker doesn't need to know the nonces.

IND-qCPA Insecurity of OCB1

Partition M into $M_1 \cdots M_m$
$L \leftarrow E_K(0^n)$
$\Delta_0 \leftarrow E_K(N \oplus L)$
Checksum $\leftarrow 0^n$
for $i = 1$ to m do
    $\Delta_i \leftarrow \gamma_i \cdot L \oplus \Delta_0$
for $i = 1$ to $m-1$ do
    $C_i \leftarrow E_K(M_i \oplus \Delta_i) \oplus \Delta_i$
    Checksum $\leftarrow$ Checksum $\oplus M_i$
$X_m \leftarrow len(M_m) \oplus L \cdot 2^{-1} \oplus \Delta_m$
$Y_m \leftarrow E_K(X_m)$
$C_m \leftarrow Y_m \oplus M_m$
Checksum $\leftarrow$ Checksum $\oplus C_m 0^* \oplus Y_m$
$Tag = E_K($Checksum $\oplus \Delta_m)$

IND-qCPA attack:

- Quantum phase: Use Simon's algorithm to recover $L = E_k(0^n)$, as seen w.r.t. existential forgery of OCB1.
- Classical phase: Pick 1-block messages $M_0^* = n \oplus L \cdot 2^{-1}$ and arbitrary $M_1^* \neq M_0^*$.
- Upon receiving (C, Tag), return $b = 0$ iff $C \oplus Tag = M_0^*$.
- Note that if M_0^* was encrypted, $X_m = M_0^* \oplus \Delta_m =$ Checksum $\oplus \Delta_m$. Hence, $Y_m = Tag$.
- Also, attacker doesn't need to know the nonces.
- Attack can be extended to OCB3 (with some additional steps).

IND-qCPA Insecurity of OCB2

Algorithm $\mathcal{E}_{E_K}(N, A, M)$

$L \leftarrow E_K(N)$

$(M[1], \ldots, M[m]) \xleftarrow{n} M$

for $i \leftarrow 1$ **to** $m - 1$

$\quad C[i] \leftarrow 2^i L \oplus E_K(2^i L \oplus M[i])$

$\text{Pad} \leftarrow E_K(2^m L \oplus \texttt{len}(M[m]))$

$C[m] \leftarrow M[m] \oplus \texttt{msb}_{|M[m]|}(\text{Pad})$

$\Sigma \leftarrow C[m] \,\|\, 0^* \oplus \text{Pad}$

$\Sigma \leftarrow M[1] \oplus \cdots \oplus M[m-1] \oplus \Sigma$

$T \leftarrow E_K(2^m 3L \oplus \Sigma)$

if $A \neq \varepsilon$ **then** $T \leftarrow T \oplus \text{PMAC}_{E_K}(A)$

$T \leftarrow \texttt{msb}_\tau(T)$

return (C, T)

IND-qCPA Insecurity of OCB2

Algorithm $\mathcal{E}_{E_K}(N, A, M)$

$L \leftarrow E_K(N)$

$(M[1], \ldots, M[m]) \xleftarrow{n} M$

for $i \leftarrow 1$ **to** $m-1$

$\quad C[i] \leftarrow 2^i L \oplus E_K(2^i L \oplus M[i])$

$\mathrm{Pad} \leftarrow E_K(2^m L \oplus \mathtt{len}(M[m]))$

$C[m] \leftarrow M[m] \oplus \mathtt{msb}_{|M[m]|}(\mathrm{Pad})$

$\Sigma \leftarrow C[m] \,\|\, 0^* \oplus \mathrm{Pad}$

$\Sigma \leftarrow M[1] \oplus \cdots \oplus M[m-1] \oplus \Sigma$

$T \leftarrow E_K(2^m 3L \oplus \Sigma)$

if $A \neq \varepsilon$ **then** $T \leftarrow T \oplus \mathrm{PMAC}_{E_K}(A)$

$T \leftarrow \mathtt{msb}_\tau(T)$

return (C, T)

Algorithm $\mathrm{PMAC}_{E_K}(A)$

$S \leftarrow 0^n$

$V \leftarrow 3^2 E_K(0^n)$

$(A[1], \ldots, A[a]) \xleftarrow{n} A$

for $i \leftarrow 1$ **to** $a-1$

$\quad S \leftarrow S \oplus E_K(2^i V \oplus A[i])$

$S \leftarrow S \oplus A[a] \,\|\, 10^*$

if $|A[a]| = n$

$\quad Q \leftarrow E_K(2^a 3V \oplus S)$

else $Q \leftarrow E_K(2^a 3^2 V \oplus S)$

return Q

IND-qCPA Insecurity of OCB2

Algorithm $\mathcal{E}_{E_K}(N, A, M)$

$L \leftarrow E_K(N)$
$(M[1], \ldots, M[m]) \xleftarrow{n} M$
for $i \leftarrow 1$ **to** $m - 1$
$\quad C[i] \leftarrow 2^i L \oplus E_K(2^i L \oplus M[i])$
$\text{Pad} \leftarrow E_K(2^m L \oplus \texttt{len}(M[m]))$
$C[m] \leftarrow M[m] \oplus \texttt{msb}_{|M[m]|}(\text{Pad})$
$\Sigma \leftarrow C[m] \,\|\, 0^* \oplus \text{Pad}$
$\Sigma \leftarrow M[1] \oplus \cdots \oplus M[m-1] \oplus \Sigma$
$T \leftarrow E_K(2^m 3L \oplus \Sigma)$
if $A \neq \varepsilon$ **then** $T \leftarrow T \oplus \text{PMAC}_{E_K}(A)$
$T \leftarrow \texttt{msb}_\tau(T)$
return (C, T)

IND-qCPA attack:

- Quantum phase: Use Simon's algorithm to recover $E_k(0^n)$, as done by Kaplan et. al. [CRYPTO'16].

IND-qCPA Insecurity of OCB2

Algorithm $\mathcal{E}_{E_K}(N, A, M)$

$L \leftarrow E_K(N)$
$(M[1], \ldots, M[m]) \stackrel{n}{\leftarrow} M$
for $i \leftarrow 1$ **to** $m-1$
$\quad C[i] \leftarrow 2^i L \oplus E_K(2^i L \oplus M[i])$
$\mathrm{Pad} \leftarrow E_K(2^m L \oplus \mathtt{len}(M[m]))$
$C[m] \leftarrow M[m] \oplus \mathtt{msb}_{|M[m]|}(\mathrm{Pad})$
$\Sigma \leftarrow C[m] \,\|\, 0^* \oplus \mathrm{Pad}$
$\Sigma \leftarrow M[1] \oplus \cdots \oplus M[m-1] \oplus \Sigma$
$T \leftarrow E_K(2^m 3L \oplus \Sigma)$
if $A \neq \varepsilon$ **then** $T \leftarrow T \oplus \mathrm{PMAC}_{E_K}(A)$
$T \leftarrow \mathtt{msb}_\tau(T)$
return (C, T)

IND-qCPA attack:

- Quantum phase: Use Simon's algorithm to recover $E_k(0^n)$, as done by Kaplan et. al. [CRYPTO'16].
- Classical phase: (…)

IND-qCPA Insecurity of OCB2

Algorithm $\mathcal{E}_{E_K}(N, A, M)$

$L \leftarrow E_K(N)$
$(M[1], \ldots, M[m]) \overset{n}{\leftarrow} M$
for $i \leftarrow 1$ **to** $m - 1$
    $C[i] \leftarrow 2^i L \oplus E_K(2^i L \oplus M[i])$
$\text{Pad} \leftarrow E_K(2^m L \oplus \texttt{len}(M[m]))$
$C[m] \leftarrow M[m] \oplus \texttt{msb}_{|M[m]|}(\text{Pad})$
$\Sigma \leftarrow C[m] \,\|\, 0^* \oplus \text{Pad}$
$\Sigma \leftarrow M[1] \oplus \cdots \oplus M[m-1] \oplus \Sigma$
$T \leftarrow E_K(2^m 3L \oplus \Sigma)$
if $A \neq \varepsilon$ **then** $T \leftarrow T \oplus \text{PMAC}_{E_K}(A)$
$T \leftarrow \texttt{msb}_\tau(T)$
return (C, T)

IND-qCPA attack:

- Quantum phase: Use Simon's algorithm to recover $E_k(0^n)$, as done by Kaplan et. al. [CRYPTO'16].
- Classical phase: (...)
- Unlike our attack against OCB1, cannot pick messages M_0^* and M_1^* depending on $L = E_k(N)$.

IND-qCPA Insecurity of OCB2

Algorithm $\mathcal{E}_{E_K}(N, A, M)$

$L \leftarrow E_K(N)$

$(M[1], \ldots, M[m]) \xleftarrow{n} M$

for $i \leftarrow 1$ **to** $m-1$

$\quad C[i] \leftarrow 2^i L \oplus E_K(2^i L \oplus M[i])$

$\mathrm{Pad} \leftarrow E_K(2^m L \oplus \mathtt{len}(M[m]))$

$C[m] \leftarrow M[m] \oplus \mathtt{msb}_{|M[m]|}(\mathrm{Pad})$

$\Sigma \leftarrow C[m] \,\|\, 0^* \oplus \mathrm{Pad}$

$\Sigma \leftarrow M[1] \oplus \cdots \oplus M[m-1] \oplus \Sigma$

$T \leftarrow E_K(2^m 3L \oplus \Sigma)$

if $A \neq \varepsilon$ **then** $T \leftarrow T \oplus \mathrm{PMAC}_{E_K}(A)$

$T \leftarrow \mathtt{msb}_\tau(T)$

return (C, T)

IND-qCPA attack:

- Quantum phase: Use Simon's algorithm to recover $E_k(0^n)$, as done by Kaplan et. al. [CRYPTO'16].
- Classical phase: (...)
- Unlike our attack against OCB1, cannot pick messages M_0^* and M_1^* depending on $L = E_k(N)$.
- Idea: Evaluate $L = E_k(N)$ in the post-challenge phase using Deutsch's algorithm – i.e., raw block-cipher access!

IND-qCPA Insecurity of OCB2

Algorithm $\mathcal{E}_{E_K}(N, A, M)$

$L \leftarrow E_K(N)$
$(M[1], \ldots, M[m]) \stackrel{n}{\leftarrow} M$
for $i \leftarrow 1$ **to** $m - 1$
$\quad C[i] \leftarrow 2^i L \oplus E_K(2^i L \oplus M[i])$
$\mathrm{Pad} \leftarrow E_K(2^m L \oplus \mathtt{len}(M[m]))$
$C[m] \leftarrow M[m] \oplus \mathtt{msb}_{|M[m]|}(\mathrm{Pad})$
$\Sigma \leftarrow C[m] \,\|\, 0^* \oplus \mathrm{Pad}$
$\Sigma \leftarrow M[1] \oplus \cdots \oplus M[m-1] \oplus \Sigma$
$T \leftarrow E_K(2^m 3L \oplus \Sigma)$
if $A \neq \varepsilon$ **then** $T \leftarrow T \oplus \mathrm{PMAC}_{E_K}(A)$
$T \leftarrow \mathtt{msb}_\tau(T)$
return (C, T)

IND-qCPA attack:

- Quantum phase: Use Simon's algorithm to recover $E_k(0^n)$, as done by Kaplan et. al. [CRYPTO'16].
- Classical phase: (...)
- Unlike our attack against OCB1, cannot pick messages M_0^* and M_1^* depending on $L = E_k(N)$.
- Idea: Evaluate $L = E_k(N)$ in the post-challenge phase using Deutsch's algorithm – i.e., raw block-cipher access!
- Assumption: Tags are untruncated – i.e., $\tau = n$.
  - We thank Melanie Jauch for pointing this issue.

IND-qCPA Security(!) of OCB2

- However, if attacker is not allowed to use AD, then OCB2 is an IND-qCPA secure AE mode when the underlying block-cipher is a quantum-secure PRP!

IND-qCPA Security(!) of OCB2

- However, if attacker is not allowed to use AD, then OCB2 is an IND-qCPA secure AE mode when the underlying block-cipher is a quantum-secure PRP!

- Recall that OCB1 and OCB3 are insecure even when AD is always empty.

IND-qCPA Security(!) of OCB2

- However, if attacker is not allowed to use AD, then OCB2 is an IND-qCPA secure AE mode when the underlying block-cipher is a quantum-secure PRP!

- Recall that OCB1 and OCB3 are insecure even when AD is always empty.

- Though OCB2, as a “pure” AE, is IND-CCA insecure [Inoue et. al., CRYPTO’19], it is still provably IND-CPA secure [Rogaway, ASIACRYPT’04].

IND-qCPA Security(!) of OCB2

- However, if attacker is not allowed to use AD, then OCB2 is an IND-qCPA secure AE mode when the underlying block-cipher is a quantum-secure PRP!

- Recall that OCB1 and OCB3 are insecure even when AD is always empty.

- Though OCB2, as a “pure” AE, is IND-CCA insecure [Inoue et. al., CRYPTO’19], it is still provably IND-CPA secure [Rogaway, ASIACRYPT’04].

- Classical IND-CPA proof interprets OCB2 as a tweakable block-cipher (XEX*) mode.
  - E is a secure PRP $\Rightarrow$ XEX* is indistinguishable from a “tweakable uniform random permutation”.

IND-qCPA Security(!) of OCB2

- Kaplan et. al. [CRYPTO'16] showed that XEX* is a "quantumly" insecure tweakable block-cipher, even if E is a quantum-secure PRP.

IND-qCPA Security(!) of OCB2

- Kaplan et. al. [CRYPTO'16] showed that XEX* is a "quantumly" insecure tweakable block-cipher, even if E is a quantum-secure PRP.
  - However, their result does not reflect how XEX* is used in OCB2.

IND-qCPA Security(!) of OCB2

- Kaplan et. al. [CRYPTO'16] showed that XEX* is a "quantumly" insecure tweakable block-cipher, even if E is a quantum-secure PRP.
    - However, their result does not reflect how XEX* is used in OCB2.

- Nonetheless, we extended their analysis to show that, in a quantum setting, XEX* remains insecure <u>even in the way it is used in OCB2</u>.

IND-qCPA Security(!) of OCB2

- Kaplan et. al. [CRYPTO'16] showed that XEX* is a "quantumly" insecure tweakable block-cipher, even if E is a quantum-secure PRP.
  - However, their result does not reflect how XEX* is used in OCB2.
- Nonetheless, we extended their analysis to show that, in a quantum setting, XEX* remains insecure <u>even in the way it is used in OCB2</u>.
- Hence to show IND-qCPA security of OCB2, must work at a block-cipher level while relying on quantum security of E.

IND-qCPA Security(!) of OCB2

- Kaplan et. al. [CRYPTO'16] showed that XEX* is a "quantumly" insecure tweakable block-cipher, even if E is a quantum-secure PRP.
  - However, their result does not reflect how XEX* is used in OCB2.

- Nonetheless, we extended their analysis to show that, in a quantum setting, XEX* remains insecure <u>even in the way it is used in OCB2</u>.

- Hence to show IND-qCPA security of OCB2, must work at a block-cipher level while relying on quantum security of E.

- We used techniques by Anand et. al. [PQCRYPTO'16] that were used to show IND-qCPA security of CBC mode.

Summary of IND-qCPA Results

OCB1
OCB2
OCB3

Summary of IND-qCPA Results

	Random Nonces, AEAD Mode
OCB1	N/A
OCB2	Insecure*
OCB3	Insecure

*when tags are untruncated.

Summary of IND-qCPA Results

	Random Nonces, AEAD Mode	Random Nonces, "Pure" AE Mode
OCB1	N/A	Insecure
OCB2	Insecure*	Secure
OCB3	Insecure	Insecure

*when tags are untruncated.

Summary of IND-qCPA Results

	Random Nonces, AEAD Mode	Random Nonces, "Pure" AE Mode	Adaptive Nonces, "Pure" AE Mode
OCB1	N/A	Insecure	Insecure
OCB2	Insecure*	Secure	Insecure
OCB3	Insecure	Insecure	Insecure

*when tags are untruncated.

Summary of IND-qCPA Results

	Random Nonces, AEAD Mode	Random Nonces, "Pure" AE Mode	Adaptive Nonces, "Pure" AE Mode
OCB1	N/A	Insecure	Insecure
OCB2	Insecure*	Secure	Insecure
OCB3	Insecure	Insecure	Insecure

Adapted a forgery attack by Bhaumik et. al. [ASIACRYPT'21] to break IND-qCPA security using only a single quantum encryption query!

*when tags are untruncated.

Extra Slides

Quantum Attack on Integrity: OCB2

$$f\colon \{0,1\}^n \rightarrow \{0,1\}^n$$
$$x \ \rightarrow \mathrm{OCB2}_k\ (N, x\|x\|0^n, \varepsilon)$$
$$f_N(x) = E_k\left(2^3 3V \oplus E_k(2V \oplus x) \oplus E_k(2^2 V \oplus x)\right) \oplus \varphi_k(N)$$

$$V \leftarrow 3^2 E_K(0^n)$$

(This is a refinement of the attack presented by Kaplan et. al. [CRYPTO'16].)

Quantum Attack on Integrity: OCB2

$$f: \{0,1\}^n \to \{0,1\}^n$$
$$x \to \mathrm{OCB2}_k\,(N, x\|x\|0^n, \varepsilon)$$
$$f_N(x) = E_k\left(2^3 3V \oplus E_k(2V \oplus x) \oplus E_k(2^2 V \oplus x)\right) \oplus \varphi_k(N)$$

$$V \leftarrow 3^2 E_K(0^n)$$

(This is a refinement of the attack presented by Kaplan et. al. [CRYPTO'16].)

- Function satisfies $f_N(x \oplus 2V \oplus 2^2 V) = f_N(x)$.

Quantum Attack on Integrity: OCB2

$$f: \{0,1\}^n \to \{0,1\}^n$$
$$x \to \mathrm{OCB2}_k\,(N, x\|x\|0^n, \varepsilon)$$
$$f_N(x) = E_k\left(2^3 3V \oplus E_k(2V \oplus x) \oplus E_k(2^2 V \oplus x)\right) \oplus \varphi_k(N)$$

$$V \leftarrow 3^2 E_K(0^n)$$

(This is a refinement of the attack presented by Kaplan et. al. [CRYPTO'16].)

- Function satisfies $f_N(x \oplus 2V \oplus 2^2 V) = f_N(x)$.
- $2V \oplus 2^2 V$ is independent of nonce N, since $V = 3^2 E_k(0^n)$.

Quantum Attack on Integrity: OCB2

$$f: \{0,1\}^n \rightarrow \{0,1\}^n$$
$$x \ \rightarrow \mathrm{OCB2}_k\,(N, x\|x\|0^n, \varepsilon)$$
$$f_N(x) = E_k\left(2^3 3V \oplus E_k(2V \oplus x) \oplus E_k\left(2^2 V \oplus x\right)\right) \oplus \varphi_k(N)$$

$$V \leftarrow 3^2 E_K(0^n)$$

(This is a refinement of the attack presented by Kaplan et. al. [CRYPTO'16].)

- Function satisfies $f_N\left(x \oplus 2V \oplus 2^2V\right) = f_N(x)$.
- $2V \oplus 2^2V$ is independent of nonce N, since $V = 3^2E_k(0^n)$.
- Can again apply Simon's algorithm w.r.t. f_N to recover $2V \oplus 2^2V$.

Quantum Attack on Integrity: OCB2

$$f: \{0,1\}^n \rightarrow \{0,1\}^n$$
$$x \rightarrow \mathrm{OCB2}_k\,(N, x\|x\|0^n, \varepsilon)$$
$$f_N(x) = E_k\left(2^3 3V \oplus E_k(2V \oplus x) \oplus E_k\left(2^2V \oplus x\right)\right) \oplus \varphi_k(N)$$

$$V \leftarrow 3^2 E_K(0^n)$$

(This is a refinement of the attack presented by Kaplan et. al. [CRYPTO'16].)

- Function satisfies $f_N\left(x \oplus 2V \oplus 2^2V\right) = f_N(x)$.
- $2V \oplus 2^2V$ is independent of nonce N, since $V = 3^2E_k(0^n)$.
- Can again apply Simon's algorithm w.r.t. f_N to recover $2V \oplus 2^2V$.

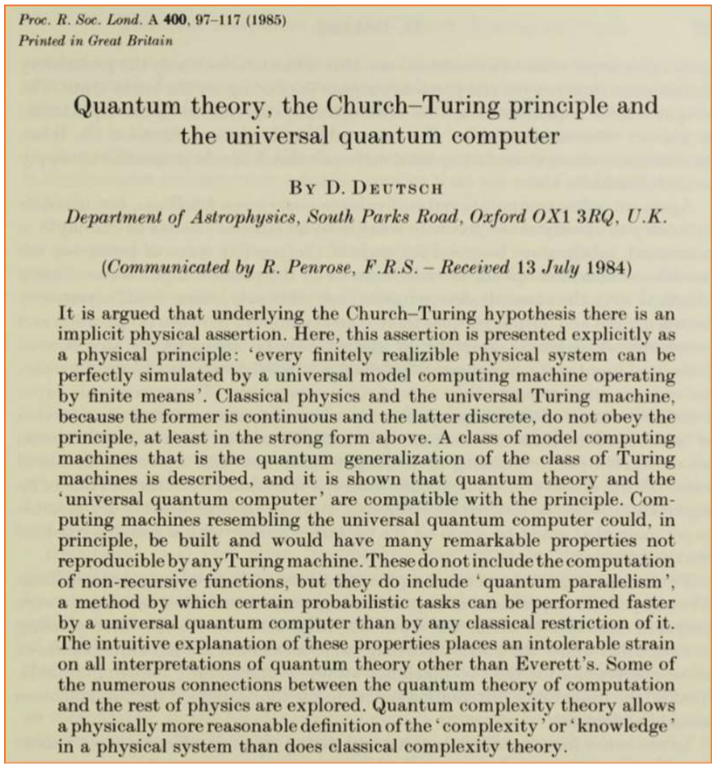
- Existential forgery: Under a random nonce N, if $\mathrm{OCB2}_k(N, A\|A\|0^n, m) = (C, T)$, then (C, T) $=\mathrm{OCB2}_k(N, (A \oplus 2V \oplus 2^2V)\|(A \oplus 2V \oplus 2^2V)\|0^n, m)$.

IND-qCPA Insecurity of OCB2

Algorithm $\mathcal{E}_{E_K}(N, A, M)$

$L \leftarrow E_K(N)$
$(M[1], \ldots, M[m]) \xleftarrow{n} M$
for $i \leftarrow 1$ **to** $m-1$
$\quad C[i] \leftarrow 2^i L \oplus E_K(2^i L \oplus M[i])$
$\mathrm{Pad} \leftarrow E_K(2^m L \oplus \mathtt{len}(M[m]))$
$C[m] \leftarrow M[m] \oplus \mathtt{msb}_{|M[m]|}(\mathrm{Pad})$
$\Sigma \leftarrow C[m] \,\|\, 0^* \oplus \mathrm{Pad}$
$\Sigma \leftarrow M[1] \oplus \cdots \oplus M[m-1] \oplus \Sigma$
$T \leftarrow E_K(2^m 3L \oplus \Sigma)$
if $A \neq \varepsilon$ **then** $T \leftarrow T \oplus \mathrm{PMAC}_{E_K}(A)$
$T \leftarrow \mathtt{msb}_\tau(T)$
return (C, T)

Deutsch’s Algorithm



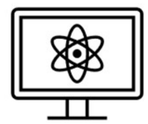
Proc. R. Soc. Lond. A **400**, 97–117 (1985)
Printed in Great Britain

Quantum theory, the Church–Turing principle and the universal quantum computer

By D. Deutsch
Department of Astrophysics, South Parks Road, Oxford OX1 3RQ, U.K.

(*Communicated by R. Penrose, F.R.S. – Received* 13 *July* 1984)

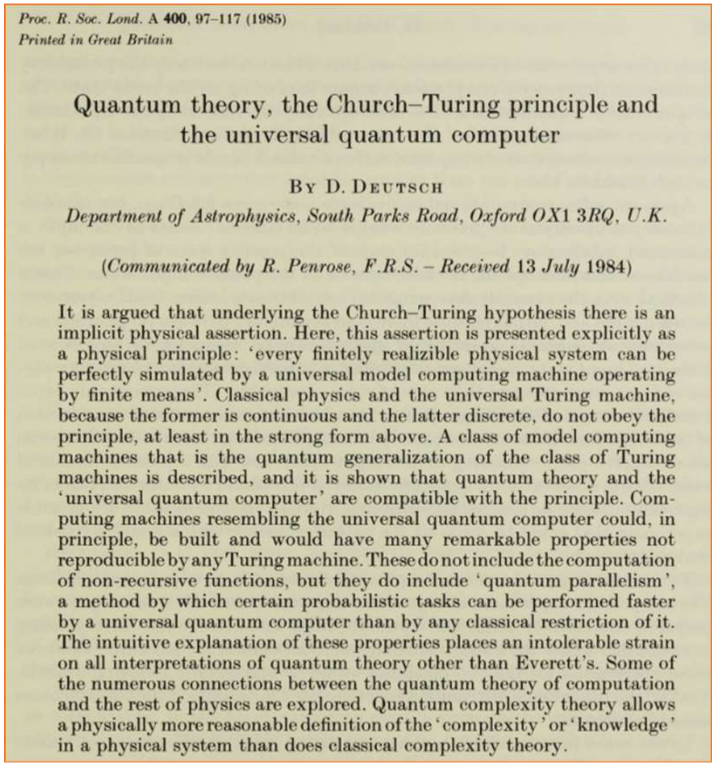
It is argued that underlying the Church–Turing hypothesis there is an implicit physical assertion. Here, this assertion is presented explicitly as a physical principle: ‘every finitely realizible physical system can be perfectly simulated by a universal model computing machine operating by finite means’. Classical physics and the universal Turing machine, because the former is continuous and the latter discrete, do not obey the principle, at least in the strong form above. A class of model computing machines that is the quantum generalization of the class of Turing machines is described, and it is shown that quantum theory and the ‘universal quantum computer’ are compatible with the principle. Computing machines resembling the universal quantum computer could, in principle, be built and would have many remarkable properties not reproducible by any Turing machine. These do not include the computation of non-recursive functions, but they do include ‘quantum parallelism’, a method by which certain probabilistic tasks can be performed faster by a universal quantum computer than by any classical restriction of it. The intuitive explanation of these properties places an intolerable strain on all interpretations of quantum theory other than Everett’s. Some of the numerous connections between the quantum theory of computation and the rest of physics are explored. Quantum complexity theory allows a physically more reasonable definition of the ‘complexity’ or ‘knowledge’ in a physical system than does classical complexity theory.



$f: \{0,1\} \rightarrow \{0,1\}$

Is f a constant function?

Deutsch's Algorithm



Proc. R. Soc. Lond. A **400**, 97–117 (1985)
Printed in Great Britain

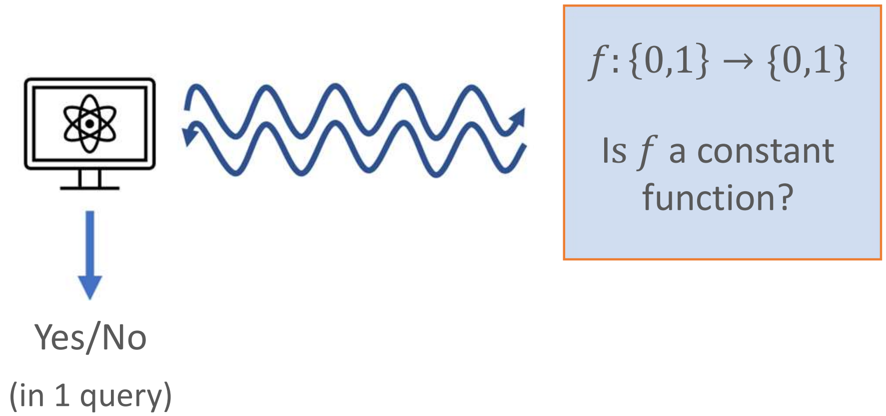
Quantum theory, the Church–Turing principle and the universal quantum computer

By D. Deutsch

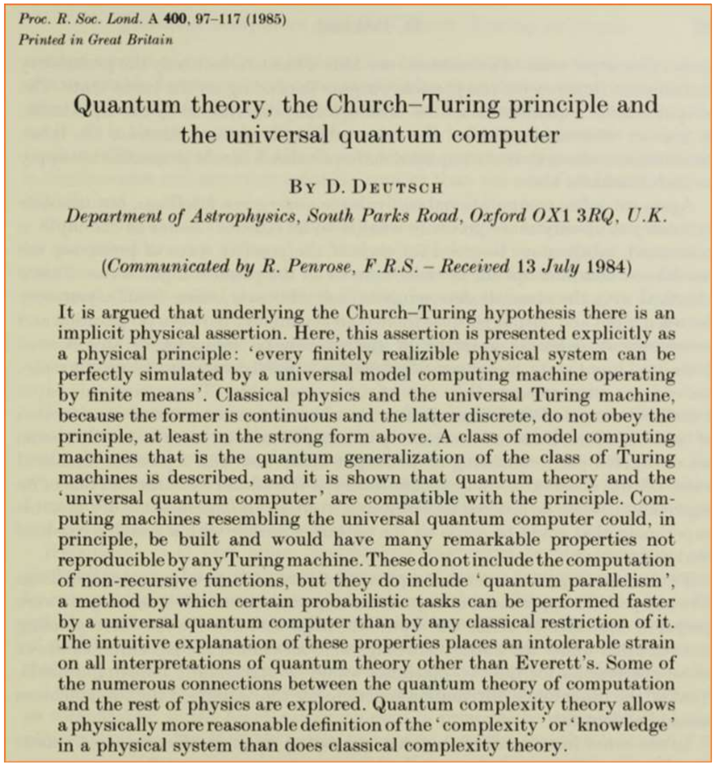
Department of Astrophysics, South Parks Road, Oxford OX1 3RQ, U.K.

(*Communicated by R. Penrose, F.R.S. – Received* 13 *July* 1984)

It is argued that underlying the Church–Turing hypothesis there is an implicit physical assertion. Here, this assertion is presented explicitly as a physical principle: 'every finitely realizible physical system can be perfectly simulated by a universal model computing machine operating by finite means'. Classical physics and the universal Turing machine, because the former is continuous and the latter discrete, do not obey the principle, at least in the strong form above. A class of model computing machines that is the quantum generalization of the class of Turing machines is described, and it is shown that quantum theory and the 'universal quantum computer' are compatible with the principle. Computing machines resembling the universal quantum computer could, in principle, be built and would have many remarkable properties not reproducible by any Turing machine. These do not include the computation of non-recursive functions, but they do include 'quantum parallelism', a method by which certain probabilistic tasks can be performed faster by a universal quantum computer than by any classical restriction of it. The intuitive explanation of these properties places an intolerable strain on all interpretations of quantum theory other than Everett's. Some of the numerous connections between the quantum theory of computation and the rest of physics are explored. Quantum complexity theory allows a physically more reasonable definition of the 'complexity' or 'knowledge' in a physical system than does classical complexity theory.



Deutsch's Algorithm


Proc. R. Soc. Lond. A **400**, 97–117 (1985)
Printed in Great Britain

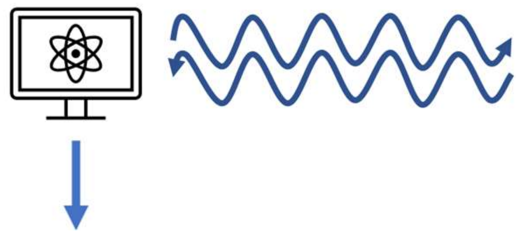
Quantum theory, the Church–Turing principle and the universal quantum computer

BY D. DEUTSCH

Department of Astrophysics, South Parks Road, Oxford OX1 3RQ, U.K.

(*Communicated by R. Penrose, F.R.S. – Received* 13 *July* 1984)

It is argued that underlying the Church–Turing hypothesis there is an implicit physical assertion. Here, this assertion is presented explicitly as a physical principle: 'every finitely realizible physical system can be perfectly simulated by a universal model computing machine operating by finite means'. Classical physics and the universal Turing machine, because the former is continuous and the latter discrete, do not obey the principle, at least in the strong form above. A class of model computing machines that is the quantum generalization of the class of Turing machines is described, and it is shown that quantum theory and the 'universal quantum computer' are compatible with the principle. Computing machines resembling the universal quantum computer could, in principle, be built and would have many remarkable properties not reproducible by any Turing machine. These do not include the computation of non-recursive functions, but they do include 'quantum parallelism', a method by which certain probabilistic tasks can be performed faster by a universal quantum computer than by any classical restriction of it. The intuitive explanation of these properties places an intolerable strain on all interpretations of quantum theory other than Everett's. Some of the numerous connections between the quantum theory of computation and the rest of physics are explored. Quantum complexity theory allows a physically more reasonable definition of the 'complexity' or 'knowledge' in a physical system than does classical complexity theory.



$f: \{0,1\} \to \{0,1\}$

Is f a constant function?

Yes/No

(in 1 query)

Deutsch's algorithm computes $f(0) \oplus f(1)$ with a single quantum query to f.

Raw Block-cipher Access: $E_k(inp)$

$$f^{(i)}: \{0,1\} \rightarrow \{0,1\}$$
$$b \rightarrow i\text{-th bit of } \{\mathrm{OCB2}_k(N, \alpha_b, \varepsilon)\},$$
$$\text{where } \alpha_0 = 2 \cdot 3V \text{ and } \alpha_1 = 2 \cdot 3V \oplus inp$$

$$V \leftarrow 3^2 E_K(0^n)$$

(This is inspired by the OCB3 forgery attack of Bonnetain et. al. [ASIACRYPT'21].)

Raw Block-cipher Access: $E_k(inp)$

$$f^{(i)}: \{0,1\} \to \{0,1\}$$
$$b \to i\text{-th bit of } \{\mathrm{OCB2}_k(N, \alpha_b, \varepsilon)\},$$
$$\text{where } \alpha_0 = 2 \cdot 3V \text{ and } \alpha_1 = 2 \cdot 3V \oplus inp$$

$$V \leftarrow 3^2 E_K(0^n)$$

(This is inspired by the OCB3 forgery attack of Bonnetain et. al. [ASIACRYPT'21].)

- We have $f^{(i)}(b) = i$–th bit of $\{E_k(3L) \oplus E_k(2 \cdot 3V \oplus \alpha_b)\}$.

Raw Block-cipher Access: $E_k(inp)$

$$f^{(i)}: \{0,1\} \to \{0,1\}$$
$$b \to i\text{-th bit of } \{\mathrm{OCB2}_k(N, \alpha_b, \varepsilon)\},$$
$$\text{where } \alpha_0 = 2 \cdot 3V \text{ and } \alpha_1 = 2 \cdot 3V \oplus inp$$

$$V \leftarrow 3^2 E_K(0^n)$$

(This is inspired by the OCB3 forgery attack of Bonnetain et. al. [ASIACRYPT'21].)

- We have $f^{(i)}(b) = i$–th bit of $\{E_k(3L) \oplus E_k(2 \cdot 3V \oplus \alpha_b)\}$.
- With a single quantum query to $f^{(i)}$, Deutsch's algorithm computes:
  - $f^{(i)}(0) \oplus f^{(i)}(1) = i$–th bit of $\{E_k(2 \cdot 3V \oplus \alpha_0) + E_k(2 \cdot 3V \oplus \alpha_1)\}$

Raw Block-cipher Access: $E_k(inp)$

$$f^{(i)}: \{0,1\} \to \{0,1\}$$
$$b \to i\text{-th bit of } \{\mathrm{OCB2}_k(N, \alpha_b, \varepsilon)\},$$
$$\text{where } \alpha_0 = 2 \cdot 3V \text{ and } \alpha_1 = 2 \cdot 3V \oplus inp$$

$$V \leftarrow 3^2 E_K(0^n)$$

(This is inspired by the OCB3 forgery attack of Bonnetain et. al. [ASIACRYPT'21].)

- We have $f^{(i)}(b) = i$–th bit of $\{E_k(3L) \oplus E_k(2 \cdot 3V \oplus \alpha_b)\}$.
- With a single quantum query to $f^{(i)}$, Deutsch's algorithm computes:
  - $f^{(i)}(0) \oplus f^{(i)}(1) = i$–th bit of $\{E_k(2 \cdot 3V \oplus \alpha_0) + E_k(2 \cdot 3V \oplus \alpha_1)\}$
    $= i$–th bit of $\{E_k(0^n) \oplus E_k(inp)\}$

Raw Block-cipher Access: $E_k(inp)$

$$f^{(i)}: \{0,1\} \to \{0,1\}$$
$$b \to i\text{-th bit of } \{\mathrm{OCB2}_k(N, \alpha_b, \varepsilon)\},$$
$$\text{where } \alpha_0 = 2 \cdot 3V \text{ and } \alpha_1 = 2 \cdot 3V \oplus inp$$

$$V \leftarrow 3^2 E_K(0^n)$$

(This is inspired by the OCB3 forgery attack of Bonnetain et. al. [ASIACRYPT'21].)

- We have $f^{(i)}(b) = i\text{−th bit of } \{E_k(3L) \oplus E_k(2 \cdot 3V \oplus \alpha_b)\}$.
- With a single quantum query to $f^{(i)}$, Deutsch's algorithm computes:
  - $f^{(i)}(0) \oplus f^{(i)}(1) = i\text{−th bit of } \{E_k(2 \cdot 3V \oplus \alpha_0) + E_k(2 \cdot 3V \oplus \alpha_1)\}$
    $= i\text{−th bit of } \{E_k(0^n) \oplus E_k(inp)\}$
- By applying Deutsch's algorithm $\forall i \in \{1, \dots, n\}$, we recover $E_k(0^n) \oplus E_k(inp)$.

Raw Block-cipher Access: $E_k(inp)$

$$f^{(i)}: \{0,1\} \rightarrow \{0,1\}$$
$$b \rightarrow i\text{-th bit of } \{\mathrm{OCB2}_k(N, \alpha_b, \varepsilon)\},$$
$$\text{where } \alpha_0 = 2 \cdot 3V \text{ and } \alpha_1 = 2 \cdot 3V \oplus inp$$

$$V \leftarrow 3^2 E_K(0^n)$$

(This is inspired by the OCB3 forgery attack of Bonnetain et. al. [ASIACRYPT'21].)

- We have $f^{(i)}(b) = i$–th bit of $\{E_k(3L) \oplus E_k(2 \cdot 3V \oplus \alpha_b)\}$.
- With a single quantum query to $f^{(i)}$, Deutsch's algorithm computes:
  - $f^{(i)}(0) \oplus f^{(i)}(1) = i$–th bit of $\{E_k(2 \cdot 3V \oplus \alpha_0) + E_k(2 \cdot 3V \oplus \alpha_1)\}$
    $= i$–th bit of $\{E_k(0^n) \oplus E_k(inp)\}$
- By applying Deutsch's algorithm $\forall i \in \{1, \ldots, n\}$, we recover $E_k(0^n) \oplus E_k(inp)$.

- Hence, prior knowledge of $E_k(0^n) \Rightarrow$ knowledge of $E_k(inp)$!

IND-qCPA Insecurity of OCB2

Algorithm $\mathcal{E}_{E_K}(N, A, M)$

$L \leftarrow E_K(N)$
$(M[1], \ldots, M[m]) \stackrel{n}{\leftarrow} M$
for $i \leftarrow 1$ **to** $m-1$
  $C[i] \leftarrow 2^i L \oplus E_K(2^i L \oplus M[i])$
$\mathrm{Pad} \leftarrow E_K(2^m L \oplus \mathtt{len}(M[m]))$
$C[m] \leftarrow M[m] \oplus \mathtt{msb}_{|M[m]|}(\mathrm{Pad})$
$\Sigma \leftarrow C[m] \,\|\, 0^* \oplus \mathrm{Pad}$
$\Sigma \leftarrow M[1] \oplus \cdots \oplus M[m-1] \oplus \Sigma$
$T \leftarrow E_K(2^m 3L \oplus \Sigma)$
if $A \neq \varepsilon$ **then** $T \leftarrow T \oplus \mathrm{PMAC}_{E_K}(A)$
$T \leftarrow \mathtt{msb}_\tau(T)$
return (C, T)

IND-qCPA attack:

- Quantum phase: Use Simon's algorithm to recover $E_k(0^n)$, as seen w.r.t. existential forgery of OCB2.

IND-qCPA Insecurity of OCB2

Algorithm $\mathcal{E}_{E_K}(N, A, M)$

$L \leftarrow E_K(N)$

$(M[1], \ldots, M[m]) \stackrel{n}{\leftarrow} M$

for $i \leftarrow 1$ **to** $m-1$

$\quad C[i] \leftarrow 2^i L \oplus E_K(2^i L \oplus M[i])$

$\text{Pad} \leftarrow E_K(2^m L \oplus \texttt{len}(M[m]))$

$C[m] \leftarrow M[m] \oplus \texttt{msb}_{|M[m]|}(\text{Pad})$

$\Sigma \leftarrow C[m] \,\|\, 0^* \oplus \text{Pad}$

$\Sigma \leftarrow M[1] \oplus \cdots \oplus M[m-1] \oplus \Sigma$

$T \leftarrow E_K(2^m 3L \oplus \Sigma)$

if $A \neq \varepsilon$ **then** $T \leftarrow T \oplus \text{PMAC}_{E_K}(A)$

$T \leftarrow \texttt{msb}_\tau(T)$

return (C, T)

IND-qCPA attack:

- Quantum phase: Use Simon's algorithm to recover $E_k(0^n)$, as seen w.r.t. existential forgery of OCB2.

- Classical phase: Pick arbitrary 1-block messages M_0^*, M_1^*, with $M_0^* \neq M_1^*$ (and set $A = \varepsilon$). Record the nonce N used by the challenger to generate the ciphertext (C, T).

IND-qCPA Insecurity of OCB2

Algorithm $\mathcal{E}_{E_K}(N, A, M)$

$L \leftarrow E_K(N)$
$(M[1], \ldots, M[m]) \xleftarrow{n} M$
for $i \leftarrow 1$ **to** $m-1$
  $C[i] \leftarrow 2^i L \oplus E_K(2^i L \oplus M[i])$
$\mathrm{Pad} \leftarrow E_K(2^m L \oplus \mathtt{len}(M[m]))$
$C[m] \leftarrow M[m] \oplus \mathtt{msb}_{|M[m]|}(\mathrm{Pad})$
$\Sigma \leftarrow C[m] \,\|\, 0^* \oplus \mathrm{Pad}$
$\Sigma \leftarrow M[1] \oplus \cdots \oplus M[m-1] \oplus \Sigma$
$T \leftarrow E_K(2^m 3L \oplus \Sigma)$
if $A \neq \varepsilon$ **then** $T \leftarrow T \oplus \mathrm{PMAC}_{E_K}(A)$
$T \leftarrow \mathtt{msb}_\tau(T)$
return (C, T)

IND-qCPA attack:

- Quantum phase: Use Simon's algorithm to recover $E_k(0^n)$, as seen w.r.t. existential forgery of OCB2.
- Classical phase: Pick arbitrary 1-block messages M_0^*, M_1^*, with $M_0^* \neq M_1^*$ (and set $A = \varepsilon$). Record the nonce N used by the challenger to generate the ciphertext (C, T).
- Quantum phase: Evaluate $L = E_k(N)$ using Deutsch's algorithm. Also compute the value $\mathbf{Pad} = E_k(2L \oplus n)$.

IND-qCPA Insecurity of OCB2

Algorithm $\mathcal{E}_{E_K}(N, A, M)$

$L \leftarrow E_K(N)$
$(M[1], \ldots, M[m]) \xleftarrow{n} M$
for $i \leftarrow 1$ **to** $m-1$
  $C[i] \leftarrow 2^i L \oplus E_K(2^i L \oplus M[i])$
$\text{Pad} \leftarrow E_K(2^m L \oplus \texttt{len}(M[m]))$
$C[m] \leftarrow M[m] \oplus \texttt{msb}_{|M[m]|}(\text{Pad})$
$\Sigma \leftarrow C[m] \,\|\, 0^* \oplus \text{Pad}$
$\Sigma \leftarrow M[1] \oplus \cdots \oplus M[m-1] \oplus \Sigma$
$T \leftarrow E_K(2^m 3L \oplus \Sigma)$
if $A \neq \varepsilon$ **then** $T \leftarrow T \oplus \text{PMAC}_{E_K}(A)$
$T \leftarrow \texttt{msb}_\tau(T)$
return (C, T)

IND-qCPA attack:

- Quantum phase: Use Simon's algorithm to recover $E_k(0^n)$, as seen w.r.t. existential forgery of OCB2.
- Classical phase: Pick arbitrary 1-block messages M_0^*, M_1^*, with $M_0^* \neq M_1^*$ (and set $A = \varepsilon$). Record the nonce N used by the challenger to generate the ciphertext (C, T).
- Quantum phase: Evaluate $L = E_k(N)$ using Deutsch's algorithm. Also compute the value $\mathbf{Pad} = E_k(2L \oplus n)$.
- Return $b = 0$ if and only if $C = M_0^* \oplus \mathbf{Pad}$.

Tweakable Block-ciphers

- A tweakable block-cipher (TBC) is a function $\widetilde{E}\colon K \times T \times M \to M$ such that $\forall\ (k,t) \in K \times T$, $\widetilde{E}(k,\ t,\ \cdot)$ is a permutation on M; here, t is the public *tweak*.

Tweakable Block-ciphers

- A tweakable block-cipher (TBC) is a function $\widetilde{E}\colon K \times T \times M \to M$ such that $\forall\ (k,t) \in K \times T$, $\widetilde{E}(k,\ t,\ \cdot)$ is a permutation on M; here, t is the public *tweak*.
  - A conventional block-cipher is a TBC where tweak-space T is singleton.

Tweakable Block-ciphers

- A tweakable block-cipher (TBC) is a function $\widetilde{E}: K \times T \times M \to M$ such that $\forall\ (k, t) \in K \times T$, $\tilde{E}(k,\ t,\ \cdot)$ is a permutation on M; here, t is the public *tweak*.
  - A conventional block-cipher is a TBC where tweak-space T is singleton.

- Like BC security, a TBC is secure if it's indistinguishable from a "tweakable uniform random permutation" (TURP) $f: T \times M \to M$.

OCB2: TBC Abstraction

Algorithm $\mathcal{E}_{E_K}(N, A, M)$

1. $L \leftarrow E_K(N)$
2. $(M[1], \ldots, M[m]) \stackrel{n}{\leftarrow} M$
3. **for** $i \leftarrow 1$ **to** $m-1$
4. $\quad C[i] \leftarrow 2^i L \oplus E_K(2^i L \oplus M[i])$
5. $\text{Pad} \leftarrow E_K(2^m L \oplus \texttt{len}(M[m]))$
6. $C[m] \leftarrow M[m] \oplus \texttt{msb}_{|M[m]|}(\text{Pad})$
7. $\Sigma \leftarrow C[m] \,\|\, 0^* \oplus \text{Pad}$
8. $\Sigma \leftarrow M[1] \oplus \cdots \oplus M[m-1] \oplus \Sigma$
9. $T \leftarrow E_K(2^m 3L \oplus \Sigma)$
10. **if** $A \neq \varepsilon$ **then** $T \leftarrow T \oplus \text{PMAC}_{E_K}(A)$
11. $T \leftarrow \texttt{msb}_\tau(T)$
12. **return** (C, T)

Algorithm $\Theta\text{CB2}.\mathcal{E}_{\widetilde{E}_K}(N, A, M)$

1. $(M[1], \ldots, M[m]) \stackrel{n}{\leftarrow} M$
2. **for** $i = 1$ **to** $m-1$
3. $\quad C[i] \leftarrow \widetilde{E}_K^{*,1,N,i,0}(M[i])$
4. $\text{Pad} \leftarrow \widetilde{E}_K^{*,0,N,m,0}(\texttt{len}(M[m]))$
5. $C[m] \leftarrow M[m] \oplus \texttt{msb}_{|M[m]|}(\text{Pad})$
6. $\Sigma \leftarrow C[m] \,\|\, 0^* \oplus \text{Pad}$
7. $\Sigma \leftarrow M[1] \oplus \cdots \oplus M[m-1] \oplus \Sigma$
8. $T \leftarrow \widetilde{E}_K^{*,0,N,m,1}(\Sigma)$
9. **return** (C, T)

OCB2: TBC Abstraction

Algorithm $\mathcal{E}_{E_K}(N, A, M)$

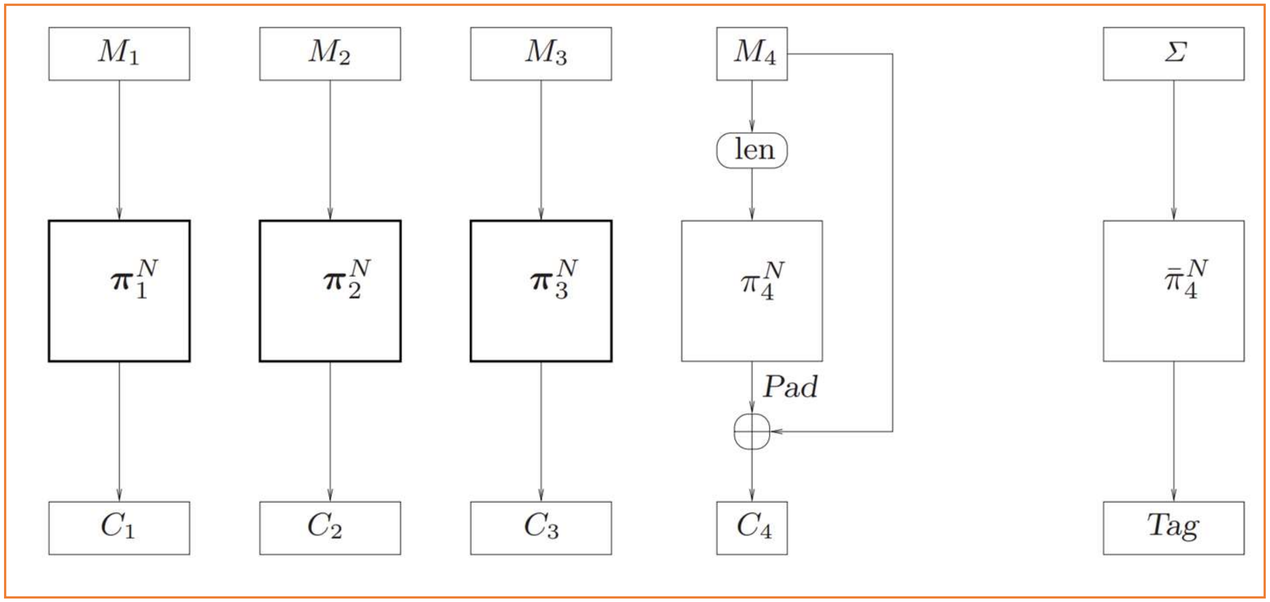
1. $L \leftarrow E_K(N)$
2. $(M[1], \ldots, M[m]) \stackrel{n}{\leftarrow} M$
3. **for** $i \leftarrow 1$ **to** $m-1$
4. $\quad C[i] \leftarrow 2^i L \oplus E_K(2^i L \oplus M[i])$
5. $\text{Pad} \leftarrow E_K(2^m L \oplus \mathtt{len}(M[m]))$
6. $C[m] \leftarrow M[m] \oplus \mathtt{msb}_{|M[m]|}(\text{Pad})$
7. $\Sigma \leftarrow C[m] \,\|\, 0^* \oplus \text{Pad}$
8. $\Sigma \leftarrow M[1] \oplus \cdots \oplus M[m-1] \oplus \Sigma$
9. $T \leftarrow E_K(2^m 3L \oplus \Sigma)$
10. **if** $A \neq \varepsilon$ **then** $T \leftarrow T \oplus \text{PMAC}_{E_K}(A)$
11. $T \leftarrow \mathtt{msb}_\tau(T)$
12. **return** (C, T)

Algorithm $\Theta\text{CB2}.\mathcal{E}_{\widetilde{E}_K}(N, A, M)$

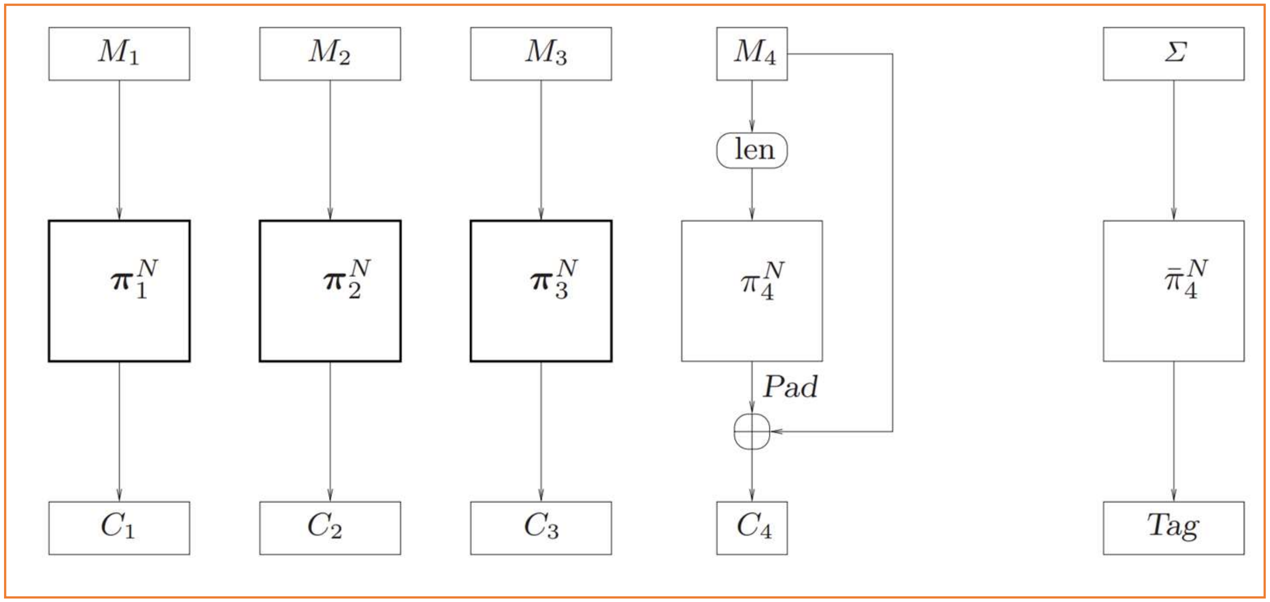
1. $(M[1], \ldots, M[m]) \stackrel{n}{\leftarrow} M$
2. **for** $i = 1$ **to** $m-1$
3. $\quad C[i] \leftarrow \widetilde{E}_K^{*,1,N,i,0}(M[i])$
4. $\text{Pad} \leftarrow \widetilde{E}_K^{*,0,N,m,0}(\mathtt{len}(M[m]))$
5. $C[m] \leftarrow M[m] \oplus \mathtt{msb}_{|M[m]|}(\text{Pad})$
6. $\Sigma \leftarrow C[m] \,\|\, 0^* \oplus \text{Pad}$
7. $\Sigma \leftarrow M[1] \oplus \cdots \oplus M[m-1] \oplus \Sigma$
8. $T \leftarrow \widetilde{E}_K^{*,0,N,m,1}(\Sigma)$
9. **return** (C, T)

$\widetilde{E}^*$: "Xor-Encrypt-Xor" (XEX*) TBC

OCB2: TBC Abstraction



OCB2: TBC Abstraction



IND-CPA advantage w.r.t. ideal TURP π = 0

OCB2: TBC Abstraction

- Kaplan et. al. [CRYPTO'16] show that an attacker querying classical tweaks and inputs in superposition can distinguish XEX* TBC from a TURP.

OCB2: TBC Abstraction

- Kaplan et. al. [CRYPTO'16] show that an attacker querying classical tweaks and inputs in superposition can distinguish XEX* TBC from a TURP.
- We extended their analysis to show that, in a quantum setting, XEX* is an insecure TBC even when tweaks are not allowed to repeat.
  - Since in OCB2, nonces are a part of tweaks.

OCB2: TBC Abstraction

- Kaplan et. al. [CRYPTO'16] show that an attacker querying classical tweaks and inputs in superposition can distinguish XEX* TBC from a TURP.

- We extended their analysis to show that, in a quantum setting, XEX* is an insecure TBC <u>even when tweaks are not allowed to repeat</u>.
  - Since in OCB2, nonces are a part of tweaks.

- Hence to show IND-qCPA security of OCB2, must work at a BC-level rather than at a TBC-level.

OCB2: TBC Abstraction

- Kaplan et. al. [CRYPTO'16] show that an attacker querying classical tweaks and inputs in superposition can distinguish XEX* TBC from a TURP.
- We extended their analysis to show that, in a quantum setting, XEX* is an insecure TBC <u>even when tweaks are not allowed to repeat</u>.
  - Since in OCB2, nonces are a part of tweaks.
- Hence to show IND-qCPA security of OCB2, must work at a BC-level rather than at a TBC-level.
- We used techniques by Anand et. al. [PQCRYPTO'16] that were used to show IND-qCPA security of CBC mode.

Other Results

- We presented quantum attacks breaking universal unforgeability of OCB2 and OCB3 in the random nonce setting.
  - Attacks use raw block-cipher access obtained via Deutsch's algorithm, as seen earlier.

Other Results

- We presented quantum attacks breaking universal unforgeability of OCB2 and OCB3 in the random nonce setting.
  - Attacks use raw block-cipher access obtained via Deutsch's algorithm, as seen earlier.
- Interestingly, the above universal forgery attacks do not extend to OCB1 – since OCB1 is a "pure" AE mode which does not process AD.
  - We were still able to break universal unforgeability of OCB1 in a quantum setting using adaptive nonces.

Other Results

- We presented quantum attacks breaking universal unforgeability of OCB2 and OCB3 in the random nonce setting.
  - Attacks use raw block-cipher access obtained via Deutsch's algorithm, as seen earlier.
- Interestingly, the above universal forgery attacks do not extend to OCB1 – since OCB1 is a "pure" AE mode which does not process AD.
  - We were still able to break universal unforgeability of OCB1 in a quantum setting using adaptive nonces.
- Our analysis of OCB2 can be used to show that the disk encryption standard XTS (IEEE P1619, NIST SP800-38E) is an IND-qCPA secure scheme when:
  - encrypted data is written on random disk sectors (to be interpreted as "nonces"), and
  - the length of messages is a multiple of block size.

Summary of IND-qCPA Results

	Random Nonces, AEAD Mode	Random Nonces, “Pure” AE Mode	Adaptive Nonces, “Pure” AE Mode
OCB1	N/A	Insecure	Insecure
OCB2	Insecure*	Secure	Insecure
OCB3	Insecure	Insecure	Insecure

*when tags are untruncated.

OCB2: TBC Abstraction

Algorithm $\mathcal{E}_{E_K}(N, A, M)$

$L \leftarrow E_K(N)$

$(M[1], \ldots, M[m]) \xleftarrow{n} M$

for $i \leftarrow 1$ **to** $m - 1$

$\quad C[i] \leftarrow 2^i L \oplus E_K(2^i L \oplus M[i])$

$\text{Pad} \leftarrow E_K(2^m L \oplus \texttt{len}(M[m]))$

$C[m] \leftarrow M[m] \oplus \texttt{msb}_{|M[m]|}(\text{Pad})$

$\Sigma \leftarrow C[m] \,\|\, 0^* \oplus \text{Pad}$

$\Sigma \leftarrow M[1] \oplus \cdots \oplus M[m-1] \oplus \Sigma$

$T \leftarrow E_K(2^m 3L \oplus \Sigma)$

if $A \neq \varepsilon$ **then** $T \leftarrow T \oplus \text{PMAC}_{E_K}(A)$

$T \leftarrow \texttt{msb}_\tau(T)$

return (C, T)

OCB2: TBC Abstraction

Algorithm $\mathcal{E}_{E_K}(N, A, M)$

$L \leftarrow E_K(N)$

$(M[1], \ldots, M[m]) \xleftarrow{n} M$

for $i \leftarrow 1$ **to** $m-1$

$\quad C[i] \leftarrow 2^i L \oplus E_K(2^i L \oplus M[i])$

$\text{Pad} \leftarrow E_K(2^m L \oplus \texttt{len}(M[m]))$

$C[m] \leftarrow M[m] \oplus \texttt{msb}_{|M[m]|}(\text{Pad})$

$\Sigma \leftarrow C[m] \,\|\, 0^* \oplus \text{Pad}$

$\Sigma \leftarrow M[1] \oplus \cdots \oplus M[m-1] \oplus \Sigma$

$T \leftarrow E_K(2^m 3L \oplus \Sigma)$

if $A \neq \varepsilon$ **then** $T \leftarrow T \oplus \text{PMAC}_{E_K}(A)$

$T \leftarrow \texttt{msb}_\tau(T)$

return (C, T)

$C[i] \leftarrow \widetilde{E}_K^{*,1,N,i,0}(M[i])$

OCB2: TBC Abstraction

Algorithm $\mathcal{E}_{E_K}(N, A, M)$

```
L ← E_K(N)
(M[1], ..., M[m]) ←^n M
for i ← 1 to m − 1
    C[i] ← 2^i L ⊕ E_K(2^i L ⊕ M[i])
Pad ← E_K(2^m L ⊕ len(M[m]))
C[m] ← M[m] ⊕ msb_{|M[m]|}(Pad)
Σ ← C[m] || 0* ⊕ Pad
Σ ← M[1] ⊕ ··· ⊕ M[m − 1] ⊕ Σ
T ← E_K(2^m 3L ⊕ Σ)
if A ≠ ε then T ← T ⊕ PMAC_{E_K}(A)
T ← msb_τ(T)
return (C, T)
```

$C[i] \leftarrow \widetilde{E}_K^{*,1,N,i,0}(M[i])$

$\widetilde{E}^{*}$: “Xor-Encrypt-Xor” (XEX*) TBC

OCB2: TBC Abstraction

Algorithm $\mathcal{E}_{E_K}(N, A, M)$

1. $L \leftarrow E_K(N)$
2. $(M[1], \ldots, M[m]) \stackrel{n}{\leftarrow} M$
3. **for** $i \leftarrow 1$ **to** $m-1$
4. $\quad C[i] \leftarrow 2^i L \oplus E_K(2^i L \oplus M[i])$
5. $\text{Pad} \leftarrow E_K(2^m L \oplus \texttt{len}(M[m]))$
6. $C[m] \leftarrow M[m] \oplus \texttt{msb}_{|M[m]|}(\text{Pad})$
7. $\Sigma \leftarrow C[m] \,\|\, 0^* \oplus \text{Pad}$
8. $\Sigma \leftarrow M[1] \oplus \cdots \oplus M[m-1] \oplus \Sigma$
9. $T \leftarrow E_K(2^m 3L \oplus \Sigma)$
10. **if** $A \neq \varepsilon$ **then** $T \leftarrow T \oplus \text{PMAC}_{E_K}(A)$
11. $T \leftarrow \texttt{msb}_\tau(T)$
12. **return** (C, T)

$C[i] \leftarrow \widetilde{E}_K^{*,1,N,i,0}(M[i])$

$\widetilde{E}^*$: "Xor-Encrypt-Xor" (XEX*) TBC

E is a secure PRP $\Rightarrow$
$\widetilde{E}^*$ is indistinguishable from a "tweakable uniform random permutation" π