Received: 16 February 2021 | Revised: 5 April 2021 | Accepted: 16 April 2021

IET Quantum Communication

DOI: 10.1049/qtc2.12010

The Institution of Engineering and Technology WILEY

ORIGINAL RESEARCH PAPER

Quantum and classical oblivious transfer: A comparative analysis

Manuel B. Santos[1,2] | **Armando N. Pinto**[3,4] | **Paulo Mateus**[1,2]

[1]Departamento de Matemática, Instituto Superior Técnico, Lisbon, Portugal

[2]Security and Quantum Information, Instituto de Telecomunicações, Lisbon, Portugal

[3]Departamento de Eletrónica, Telecomunicações e Informática, Universidade de Aveiro, Aveiro, Portugal

[4]Optical Quantum Communications and Technologies, Instituto de Telecomunicações, Aveiro, Portugal

Correspondence

Manuel B. Santos, Departamento de Matemática, Instituto Superior Técnico, Lisbon, Portugal.
Email: manuel.batalha.santos@ist.utl.pt

Funding information

European Regional Development Fund, Grant/Award Number: POCI-01-0247- FEDER-039728; Fundação para a Ciência e a Tecnologia, Grant/Award Numbers: SFRH/BD/144806/2019, UIDB/50008/2020 - UIDP/50008/2020

Abstract

Secure multiparty computation has the potential to be a disruptive technique in the realm of data analysis and computation. It enables several parties to compute virtually any function while preserving the privacy of their inputs. However, most of its protocols' security and efficiency relies on the security and efficiency of oblivious transfer (OT). In this work, we make a detailed comparison between the complexity of the hybrid quantum oblivious transfer (HQOT) protocol presented in [11] and the classical OT [12], which to the best of our knowledge, is the fastest OT protocol. We also propose an optimised version of HQOT and discuss several other OT protocols generated from oblivious keys.

1 | INTRODUCTION

The concept of oblivious transfer (OT) was introduced by Rabin [1] in 1981, and its use has been extended to a set of very important applications related to private computation: private set intersection [2], private set union [3], oblivious linear evaluation [4], and secure multiparty computation (SMC) protocols (Yao [5], GMW [6], SPDZ [7]).

A decade after Rabin's work, the first OT protocol was developed, leveraging the properties of quantum physics [8, 9]. The emergence of quantum computers and its threat to classical cryptographic techniques [10] has increased interest in these quantum secure protocols, leading to further studies on the topic of quantum OT. However, to the best of our knowledge, there is no comparative study on the complexity of the quantum and classical approaches. This can be explained by the discrepancy in technological maturity between quantum and classical techniques. Quantum technology is still in its infancy, whereas classical processors and communication have many decades of development. However, it is reasonable to benchmark the transfer phase of both quantum and classical approaches for two-phase-structure protocols. The hybrid quantum oblivious transfer (HQOT) protocol presented in [11] is a two-phase structure (precomputation and transfer) protocol that only uses quantum technology during the first (precomputation) phase. Similarly, most of the classical OT protocols are two-phase-structure protocols, where during the precomputation phase, the parties generate two keys that are used during the transfer phase of the protocol.

We compare the transfer phase complexity of an optimised version of HQOT and the transfer phase of the fastest reported classical OT implementation of which we are aware. We conclude that the HQOT transfer phase has less complexity than the OT extension version [12]. In addition, we discuss how to generate correlated OT, sender random OT, receiver random OT, and random OT protocols based on oblivious keys.

Besides this introduction, this paper comprises four other sections. In Section 2, we perform a security and efficiency assessment of classical protocols. In Section 3, we present an optimised version of HQOT and propose four other flavours of OT based on quantum oblivious keys (QOKs). In the last section, we compare the computational and communicational complexity of the optimised HQOT and the fastest classical implementation of OT [12].

This is an open access article under the terms of the Creative Commons Attribution-NonCommercial-NoDerivs License, which permits use and distribution in any medium, provided the original work is properly cited, the use is non-commercial and no modifications or adaptations are made.

© 2021 The Authors. *IET Quantum Communication* published by John Wiley & Sons Ltd on behalf of The Institution of Engineering and Technology.

2 | CLASSICAL OBLIVIOUS TRANSFER

Let us start by defining the functionality of OT. Several equivalent flavours of OT are present in the literature, but we will only focus on the 1-out-of-2 OT. Simply put, it allows some party S (usually called sender) with two l-bit strings (m_0, m_1) to send m_b to some other party R (usually called a receiver) where b is chosen by R. In addition, it must be concealing (the sender cannot learn the receiver's bit choice b) and must preserve obliviousness (the receiver cannot learn the message m_{1-b}). Schematically, the OT functionality is described in Figure 1 [13].

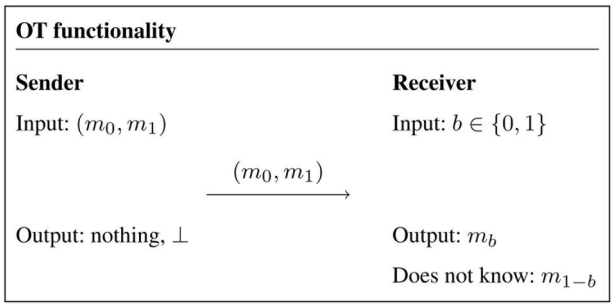


FIGURE 1 Oblivious transfer functionality

2.1 | Mathematical preliminaries

Before presenting a specific classical protocol that implements the OT functionality, we briefly provide the (informal) definitions of some important mathematical concepts:

- $\gcd(a, b)$, $a, b \in \mathbb{Z}$: denotes the greatest common divisor between integers a and b.
- $\mathbb{Z}_n = \{0, 1, \ldots, n-1\}$: denotes the set of integers $i \bmod n$.
- $\mathbb{Z}_n^*$: denotes the set of integers $i \in \mathbb{Z}_n$ that are coprime with n, that is $\gcd(i, n) = 1$. For n prime, $\mathbb{Z}_n^* = \{1, \ldots, n-1\}$ forms a multiplicative group of order $n-1$.
- A generator g of some group G is an element in G such that $\forall i \in G, \exists r\colon g^r = i$.
- The discrete logarithm base g of some element i, denoted by $\log_g i$, is the power r of g such that $g^r = i$.

2.2 | Bellare–Micali OT

In this section, we present the Bellare–Micali OT (BM OT) protocol [14] based on public-key Diffie–Hellman (DH) to shed some light on the issues related to OT classical implementations. The security and efficiency issues exposed in this section also apply to most of the major classical protocols [13, 15, 16].

In this protocol, we consider G_q to be a subgroup of $\mathbb{Z}_p^*$ with generator g and order q, where p is prime and $p = 2q + 1$. In addition, we assume public knowledge on the value of some constant $C \in G_q$. This constant guarantees that the receiver follows the protocol. For comparison purposes with the OT quantum version presented below, we split the BM OT protocol into two phases: precomputation and transfer. The first phase sets the necessary resources to execute the OT in the second phase. The protocol is shown in Figure 2.

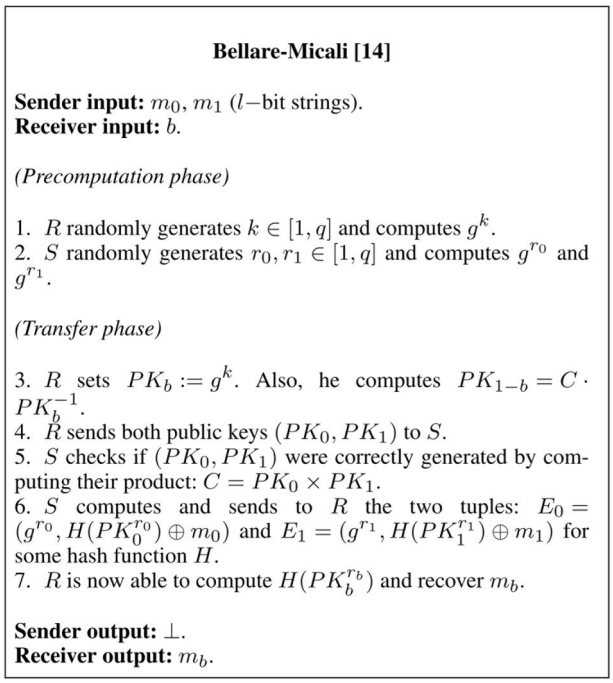


FIGURE 2 Bellare–Micali oblivious transfer protocol divided into two phases

2.2.1 | Security issues

The BM OT protocol is secure if it complies with both the concealing and the obliviousness property. The former is achieved because the receiver does not send any information to the sender that reveals his input bit choice. The latter relies on the sender's ability to keep his randomly generated elements r_0 and r_1 private. Thus, the obliviousness property is compromised if the receiver is able to compute the discrete logarithm of g^{r_i} for $i = 0, 1$ (discrete logarithm problem).

The hardness of the discrete logarithm problem on cyclic groups is the basis of several other important protocols, which makes it crucial to understand its security limits. Nevertheless, it remains to be proven whether, given a general cyclic group G with generator g and order n, a polynomial-time algorithm exists that computes r from g^r, where $r \in \mathbb{Z}_n$. Thus, the BM OT protocol's security relies on the assumption that the receiver has limited computational power and is not able to compute the discrete logarithm of a random number.

Although the general discrete logarithm problem is not known to be tractable in polynomial-time, there are specific cases where it is possible to compute it efficiently. This leads to some classical attacks where the structure of the cyclic group considered is not robust enough. As an example, if a prime p is

randomly generated without ensuring that $p-1$ contains a big prime p_b in its decomposition, it is possible to use a divide-and-conquer technique [17] along with some other methods (Shank's method [18], Pollard's rho [19], Pollard's lambda [19]) to compute the discrete logarithm of a number $a \in G_q$. In this case, the computation time will only depend on the size of p_b. So the smaller p_b is, the faster the algorithm can be. To avoid these types of attacks, safe primes are recommended, that is, $p = 2q + 1$ prime, where q is also prime. However, it is computationally more expensive to find safe primes because they occur less frequently than prime numbers. Beyond the cyclic group structure Otherwise, it is possible to compute the discrete logarithm in an acceptable time. As reported in [20], after one week of precomputation, it is possible to compute the discrete logarithm in a 512-bit group in one minute by using the number field sieve algorithm. So by following this method, after a week-long computation, the receiver would be able to receive both messages m_0 and m_1 of the BM OT protocol in one minute. In an SMC scenario with the Yao approach [5], where each OT performed corresponds to one evaluator's input and the chosen group parameters are fixed, the evaluator would be able to obtain the keys corresponding to both 0 and 1 bit, and after some reverse engineering, would be able to discover the garbler's inputs. Therefore, at the expense of efficiency, it is necessary to use big enough prime numbers (2048 bit or larger), for which these classical attacks could not be feasibly implemented.

We have just seen specific examples where it is possible to break the security of OT protocol using classical techniques. However, it is known that it is possible to break the general discrete logarithm problem with a quantum computer. In 1995, Peter Shor published a quantum algorithm that can solve both Prime Factorization and discrete logarithm problems in polynomial-time [10]. This remarkable finding poses a threat to most of our currently deployed asymmetric cryptographic protocols (Rivest–Shamir–Adleman, elliptic-curve cryptography, and DH key exchange), as their security is based on these computational assumptions. Therefore, in the BM OT protocol, the receiver R would be able to perform two attacks with the help of a quantum computer:

Quantum attack 1:

1. R computes the discrete logarithm of $g^{r_{1-b}}$ received from S using Shor's algorithm, that is, $r_{1-b} = \log_g g^{r_{1-b}}$.
2. R is then able to compute $H\left((g^{r_b})^k\right) = H(PK_b^{r_b})$ and $H(PK_{1-b}^{r_{1-b}})$ and receive both messages m_b and m_{b-1}.

Quantum attack 2:

1. R computes the discrete logarithm of PK_{1-b} with the Shor's algorithm, that is, $s = \log_g PK_{1-b}$.
2. R is then able to compute $H\left((g^{r_b})^k\right) = H(PK_b^{r_b})$ and $H((g^{r_{1-b}})^s) = H(PK_{1-b}^{r_{1-b}})$ and receive both messages m_b and m_{1-b}.

In the research literature, there are mainly two approaches to tackle this issue: the development of protocols with assumptions on the computational power of quantum computers and the development of protocols that make use of quantum technology. The former is known as post-quantum cryptography [21], and its public-key cryptography protocols are generally more demanding because of the nature of the computational assumptions used. It is also worth stressing that these computational assumptions are still unproven and have survived just a few years of scrutiny, making the post-quantum approach likely to be attacked in the near future. The latter is known as quantum cryptography [22] and provides solutions without any computational assumption. It has the potential to provide faster protocols but drastically increases the cost of the technological equipment required. Finally, it is important to note that, in general, quantum protocols do not suffer from *intercept now–decipher later* attacks because they base their security only on quantum theory. Conversely, this possible threat is always present in protocols based on computational assumptions.

2.2.2 | Efficiency issues

In the previous section, we noted that every mitigation process used to increase security would bring a downside in efficiency: generating safe primes is more demanding, computing bigger exponents and module primes is heavier in general, and finding post-quantum solutions requires stronger computational assumptions and thus tends to increase computational complexity.

Now, let us understand the efficiency limitations of the BM OT protocol. We start by looking at the operations used in the protocol (random number generation, modular multiplication, modular inversion, modular exponentiation, hash evaluation, XOR operation) from which the most demanding operation is modular exponentiation. For this reason, the complexity of BM OT heavily depends on the complexity of modular exponentiation. The number of modular exponentiation executed in each phase is summarised in Table 1.

One of the most efficient methods to compute general modular exponentiation with $n-$bit numbers is through a square-and-multiply algorithm along with Karatsuba multiplication. The former method takes $\mathcal{O}(n)$ multiplications, and the latter has complexity $\mathcal{O}(n^{1.58})$. Thus, the overall method takes $\mathcal{O}(n^{2.58})$ $n-$ bit operations [23]. To set an overestimation on the OT generation rate, let us only consider the time (in CPU cycles) required to compute the modular exponentiation operations. We can use the following expression:

$$\left(\frac{C_{mexp}}{C_{cycles}} \times N_{mexp}\right)^{-1} \tag{1}$$

where C_{mexp} is the number of CPU cycles required to compute modular exponentiation, C_{cycles} is the CPU frequency (number of cycles per second), and N_{mexp} is the number of modular exponentiations performed in the OT implementation. This

TABLE 1 Number of modular exponentiations for each phase

	Sender	Receiver
Precomputation phase	2	1
Transfer phase	2	1

expression only renders an overestimation because it depends on both the implementation of the modular exponentiation operation and the CPU frequency used.

Considering a standard CPU operating around 2.5 GHz ($C_{cycles} = 2.5 \times 10^9$ cycles per second) and a very efficient implementation of modular exponentiation [24] ($C_{mexp} \sim$ 400,000 CPU cycles), the BM OT protocol would be able to perform at most ~1041 BM OTs in 1 s as represented in Figure 3. Note that this is a very loose overestimation of the number of OTs per second. Here, we just considered the computational complexity of modular exponentiation, and we assumed that all the other operations do not have a big impact on the computation time. So we can conclude that the real OT rate must be well below this threshold. As reported in [12], it takes around 18 ms to generate a similar OT protocol, Naor–Pinkas (NP) OT [15], that requires five modular exponentiations. This corresponds to a rate of just 56 OTs per second.

The OT rates presented above lead to serious constraints on the execution of SMC protocols that rely on OT. The Yao SMC protocol [5] uses Boolean circuits to privately compute the desired functionality and requires as many OTs as half the number of input wires. Thus, the execution time of the OT phase of the Yao protocol with a 32,000 input Boolean circuit would take at least 16 s using our rough OT rate estimation and around 2 min 23 s using NP OT rate. In a deployment environment where it is required to evaluate several rounds of the same circuit, this approach becomes impractical, and higher rates must be achieved.

2.3 | OT extension protocols

Because most of the required computation to achieve OT comes from asymmetric cryptographic primitives that use modular exponentiation, it would be desirable to substitute it with more efficient methods. Symmetric cryptography has the advantage of being more efficient than asymmetric cryptography. In addition, the known quantum attacks based on Grover's algorithm only provide a quadratic advantage over classical approaches, which can be mitigated by doubling the size of the symmetric keys [21]. Unfortunately, it was proved by Impagliazzo and Rudich that OT protocols require asymmetric cryptographic assumptions [25]. This means OT cannot be performed by symmetric cryptographic tools alone.

Nonetheless, researchers developed some OT schemes to circumvent Impagliazzo and Rudich's result using hybrid protocols mixing symmetric and asymmetric cryptography. This idea was introduced by Beaver [26], where he showed that it is possible to extend the number of OT using symmetric cryptography when a small number of base OT is created using asymmetric cryptography. Although Beaver's protocol was very inefficient, it paved the way for more efficient implementations [12, 27–30]. Currently, one of the most efficient protocols can generate around 10 million OTs in 2.62 s in a back-to-back scenario [12]. Because these protocols use a small number of base OTs and quantum secure symmetric tools, the security of the extended protocol is mainly dependent on the security of the base OT protocol. Moreover, the protocol that we analyse in Section 4.2 [12] is not secure against malicious parties and must only be deployed in a semi-honest environment. Protocols that are secure against malicious parties need an extra consistency check phase, which increases their complexity [30, 31].

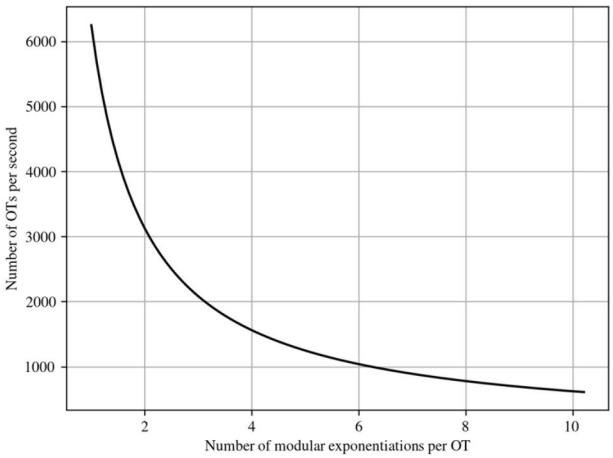


FIGURE 3 Plot of expression (1) on the overestimation of the rate against the number of modular exponentiation operations required per OT. OT, oblivious transfer

3 | HYBRID QUANTUM OBLIVIOUS TRANSFER

The intrinsic randomness provided by quantum phenomena is an ideal resource that allows the first-ever type of OT invented by Rabin to be achieved [1]. This flavour is commonly known as 1/2 OT because the receiver only receives with probability $\frac{1}{2}$ the unique message m sent by the sender. In principle, this could be performed using a single photon and a BB84 setting [8]. However, as pointed in [8], this direct application suffers from several flaws. In practice, the probability of success would be deeply influenced by the efficiency detection of the receiver. In addition, a receiver with access to quantum memory can receive the message with full probability. Furthermore, the angle of measurement could be changed to increase the amount of partial information of the message.

From this inherent randomness that can create insecure Rabin OT, Bennet [8] and Crépeau [9] presented the first quantum oblivious transfer (QOT) protocol. The argument used to construct QOT out of the quantum randomness is very similar to the classical equivalent used in [32] by Crépeau. However, the authors could not show their protocol to be

secure against the most general class of measurements. In 1993, Brassard et al. came up with proof of security of an interesting primitive, known as quantum bit commitment (QBC), that would allow the building of secure QOT. Two years later, Yao proved the unconditional security of [8] the initial proposal based on the Brassard et al. proof. Interestingly, Mayers [33] discovered that the Brassard et al. proof was flawed and went further by proving that it is not possible to have unconditionally secure QBC under non-relativistic physics [34]. Independently, Lo and Chau [35] reached the same result. This led to the unfortunate conclusion that QOT could not be securely built from QBC.

Regardless of the ideal scenario, several protocols have been developed that prove practical security under some assumptions on the technological power of the adversary: low level of quantum memories (noisy-storage model) and technological limitation to perform generalised measurements [36–40]. In addition, Unruh proved that the Bennet and Crépeau protocol with ideal commitments is secure in the stand-alone and universal composable models [41]. Thus, one could think of using some classical BC protocol in the QOT protocol instead of its quantum version. In fact, this hybrid implementation is still secure against quantum computers in the quantum random oracle model; that is, in practice, it is secure as long as we use quantum-resistant hash functions [42, 43].

This idea was explored by Lemus et al. [11] who used a provably secure cryptographic hash-based commitment scheme [44] to build a practical OT protocol with stand-alone and composable security. Beyond being resistant to quantum computer attacks, this protocol provides a practical way to perform OTs in an SMC environment. It is divided into two independent phases (oblivious key phase and transfer phase), where the first phase corresponds to a precomputation phase that uses quantum technologies, and the second phase classically executes the OT based on the precomputed elements. Consequently, we do not need the quantum equipment to be running concurrently with the SMC execution.

3.1 | Quantum oblivious keys

In order to understand the HQOT protocol presented below, we start by introducing the concept of *oblivious keys*. These are asymmetric keys shared between two agents, usually called sender and receiver, such that the sender knows all the elements of the key (k^S) while the receiver only knows some fraction of the key (k^R) and knows which part of the key is correlated with the sender's key (x^R). So an oblivious key is a tuple (k^S, (k^R, x^R)) where k^S is owned by the sender and (k^R, x^R) is owned by the receiver such that $x_i^R = 0$ when k_i^S and k_i^R are correlated and $x_i^R = 1$ otherwise. When two indexes, k_i^S and k_i^R, are correlated, we have that $k_i^S = k_i^R$. It is important to note that these oblivious keys are primarily generated through quantum processes [11, 45–60], and to the best of our knowledge, there are no classical protocols to generate them. For this reason, we may also call them QOKs. Conceptually, the idea of QOK is similar to that of quantum keys (QKs). In the same way as QK is the resource that allows encryption of a specific message, QOK is the resource that enables the performance of OT with messages. In other words, encryption methods consume QK, while OT methods consume QOK.

The term *oblivious key* was coined for the first time in the context of OT protocols by Jakobi et al. ([61], 2011). They developed this concept to implement a 1-out-of-N OT, also known as private database queries. They avoid the assumptions of the no-go theorems [33, 35] because their protocol is not perfectly concealing; that is, the sender may obtain some (limited) information about the receiver's choice. This protocol inspired several works that tried to improve on Jakobi's initial proposal [45–59]. Moreover, this concept was introduced earlier under a different name (*weak string erasure*) and with its security based on the noisy-storage model [60].

3.2 | Hybrid quantum oblivious transfer

Lemus et al.'s protocol [11] follows a different line of research and can achieve OT along with the concealing property and without assuming any constraint in the quantum memory of the adversary. As already mentioned, this hybrid approach starts from generating oblivious keys where the receiver knows half of the sender's key.

This first phase, also known as the *oblivious keys phase*, follows the quantum approach developed by Bennet [8] with Halevi and Micali classical bit commitments provided by universal and cryptographic hashing [44]. Here, we give a discrete variable description of the protocol with photon polarisation encoding [11]. The generation of oblivious keys is divided into three steps:

1. Setup phase:

In this phase, the sender generates two random binary strings s, a of the same size, where s represents the binary encoded elements, and a represents the set of bases in which s elements are encoded. For $a_i = 0$ (1), s_i is encoded in the computational (Hadamard) basis. For short, we denote $|0\rangle = |(0,0)$, $|1\rangle = |(1,0)\rangle$, $|+\rangle = |(0,1)\rangle$ and $|-\rangle = |(1,1)\rangle$. The sender sends $|\phi\rangle = |\phi_1 \ldots \phi_{2l+t}\rangle$ to the receiver, where $|\phi_i\rangle = |(s_i, a_i)\rangle$, l is the length of the OT messages, and t is the testing qubits.

The receiver measures in a random basis given by string $\tilde{a}$ and computes the string $\tilde{s}$ accordingly.

2. Commitment and testing phase:

This phase ensures that the receiver measures the received state during the *setup phase* and does not delay these measurements after the *revealing phase* below. The receiver starts by committing to pairs $(\tilde{s}_i, \tilde{a}_i)$ using the hash-based commitments proposed in [44] and sends them to the sender. The sender asks the receiver to open some subset of the commitments, T, of size t. Then, the sender checks if $s_i = \tilde{s}_i$ whenever $a_i = \tilde{a}_i$. If these tests pass, they proceed to the next phase.

3. Revealing phase:

At this point, the sender reveals to the receiver the bases $a|_{\bar{T}}$ he used during the *setup phase* and sets his oblivious key to $k^S = s|_{\bar{T}}$. The receiver defines $x^R = \tilde{a}|_{\bar{T}} \oplus a|_{\bar{T}}$ and defines his corresponding oblivious key to be $k^R = \tilde{s}|_{\bar{T}}$.

The security of this phase relies on the laws of quantum theory (quantum uncertainty principle) and on the fact that there is no significant quantum speed-up to find collisions on the hash-based bit commitments [11, 44]. In addition, it is important to stress that *intercept now–decipher later* attacks are not effective. This is because the collisions to the hash-based commitments have to be found during the execution of the *commitment and testing phase*, and there is no impact on the protocol's security if they are found later. The use of bit commitments is also important because it makes this protocol secure against malicious adversaries. Finally, note that the oblivious key phase is a precomputation phase that can be executed before executing the actual OT. This approach enables the execution of a quantum-safe Yao GC protocol [5] without the need to use quantum technology simultaneously with its execution.

The oblivious key protocol could then be used to realise OT using a protocol similar to the reduction of $\frac{1}{2}$ OT to 1-out-of-2 OT. This step is described in Figure 4. Here, we consider the sender to have two l–bit messages, the receiver has one bit choice, and both share an oblivious key. The receiver starts by defining two sets, I_0 and I_1, that correspond to the indexes where the oblivious keys are correlated and to the indexes where they are not, respectively. In addition, note that the sets I_0 and I_1 have length l because both are used to mask the l–bit messages m_0 and m_1 according to b as described in step 2. This hybrid protocol only requires $2l$ comparisons, three l–bit XOR operations, three l–bit truncations, and two communication rounds.

OK → OT

Sender input: m_0, m_1 (l–bit strings).
Receiver input: b.
Oblivious key pair: $(k^S, (k^R, x^R))$.

(Transfer phase)

1. R defines $I_0 = \{i : x_i^R = 0\}$ and $I_1 = \{i : x_i^R = 1\}$ and sends the pair $(I_b, I_{b\oplus 1})$ to S.
2. S sets the pair (e_0, e_1) as $e_i = m_i \oplus k^S|_{I_{b\oplus i}}$ and sends it to R.
3. R computes $m_b = e_b \oplus k^R|_{I_0}$.

Sender output: $\perp$.
Receiver output: m_b.

FIGURE 4 Hybrid quantum oblivious transfer (OT) phase

OK → O-OT

Sender input: m_0, m_1 (l–bit strings).
Receiver input: b.
Oblivious key pair: $(k^S, (k^R, x^R))$.

(Transfer phase)

1. R defines $I_0 = \{i : x_i^R = 0\}$ and $I_1 = \{i : x_i^R = 1\}$ and **sends only** I_b to S.
2. S sets the pair (e_0, e_1) as $e_i = m_i \oplus k^S|_{I_{b\oplus i}}$ and sends it to R.
3. R computes $m_b = e_b \oplus k^R|_{I_0}$.

Sender output: $\perp$.
Receiver output: m_b.

FIGURE 5 Optimised hybrid quantum oblivious transfer (O-OT) phase

3.3 | Optimised hybrid quantum OT

Now, observe that the transfer phase presented before uses $4l$ bits during the two communication rounds. This can be reduced as it is redundant to send both set of indexes $(I_b, I_{b\oplus 1})$ during the first communication round. In fact, with only one set (I_b) received, the sender is able to know its complement $(I_b = I_{b\oplus 1})$. Thus, we end up with the optimised protocol (Optimised HQOT [O-OT]) presented in Figure 5.

This O-OT version requires less bandwidth compared with the initially proposed HQOT transfer phase. In this case, the receiver only needs to send l bits to the sender, leading to an overall reduction of one-fourth in the number of bits sent.

3.4 | Correlated hybrid quantum OT

Some optimisations of the Yao GC protocol (e.g. free-XOR and point-and-permute) allow the use of a particular type of OT where the keys corresponding to 0 and 1 are correlated. In this setting, the sender defines a random value $\Delta \in \{0,1\}^l$ that is used during the Yao GC OT phase. Then, the parties proceed as described in Figure 6.

This correlated HQOT (C-OT) approach decreases the bandwidth of the transfer phase from $3l$ to $2l$, compared with the O-OT version presented before. This comes from the fact that the sender only sends one value e to the receiver instead of a pair (e_0, e_1). This reduction is similar to the one presented in [12], but in C-OT, we make use of oblivious keys as the only resource to generate this random correlated OT version. Note that this version does not increase the computational complexity when compared with O-OT. The sender computes twotwo l–bit XOR operations ($m_1 = \Delta \oplus k^S|_{I_b}$ and $e = m_1 \oplus k^S|_{I_{b\oplus 1}}$), and the receiver only computes one l–bit XOR operation if $b = 1$. In case $b = 0$, the receiver does not need to compute any l–bit XOR operation to recover m_b, reducing the computational complexity compared with O-OT. Finally, the security of C-OT is based on only the sender knowing Δ and $k^R|_{I_1} \neq k^S|_{I_1}$.

OK → C-OT

Sender input: Δ ($l-$bit strings).
Receiver input: b.
Oblivious key pair: $(k^S, (k^R, x^R))$.

(Transfer phase)

1. R defines $I_0 = \{i : x_i^R = 0\}$ and $I_1 = \{i : x_i^R = 1\}$ and sends I_b to S.
2. S sets $m_0 := k^S|_{I_b}$ and $m_1 := \Delta \oplus m_0 = \Delta \oplus k^S|_{I_b}$. Defines $e = \Delta \oplus k^S|_{I_0} \oplus k^S|_{I_1}$ and sends e to R.
3. R proceeds as follows:
(a) If $b = 0$: $k^R|_{I_0} = k^S|_{I_0} = m_0$.
(b) If $b = 1$: computes $e \oplus k^R|_{I_0} = \Delta \oplus k^S|_{I_1} = m_1$

Sender output: $m_0 := k^S|_{I_b}$ and $m_1 := \Delta \oplus m_0$.
Receiver output: m_b.

FIGURE 6 Correlated hybrid quantum oblivious transfer (C-OT) phase

3.5 | Sender random hybrid quantum OT

Even without using the Yao GC optimisations that required a special type of OT (C-OT), we can further optimise the O-OT and tailor it to the plain Yao GC protocol. This SMC protocol uses OT to provide the evaluator with corresponding input keys to evaluate the circuit. Therefore, because the input wire keys are randomly generated by the garbler, it is more efficient to directly use the oblivious keys as the input wire keys. So we do not need to transfer any messages, and we can use a randomised version of the HQOT protocol. In this case, the messages m_0 and m_1 are not chosen by the sender while the bit choice b is still chosen by the receiver.

As presented in Figure 7, this sender random HQOT (SR-OT) only requires $2l$ comparisons, three $l-$bit truncations, and one communication round during the *transfer phase*. All these operations are linear in size (l) of the key generated, $\mathcal{O}(l)$, and the number of communication rounds is optimal.

3.6 | Receiver random hybrid quantum OT

The oblivious keys can also generate a different type of oblivious transfer: receiver random OT (Figure 8). As explained in [62], in this protocol, the receiver does not input the bit choice b but receives it as an output along with the corresponding message, m_b. In this setting, we need to define $I^j_{\min}$ to be the set with the smallest index, $I^j_{\min} := I_j$, where min $I_j \leq$ min I_i, $i \in \{0, 1\}$. Note that from the definition of oblivious keys, $I^j_{\min}$, is either I_0 or I_1 with equal probability. This fact masks the set received by the sender in step 1. From the sender perspective, $I^j_{\min}$ does not reveal any information about the j value. In addition, its value is random and not defined by the receiver. Because $I^j_{\min}$ defines the key used to encrypt m_0, the receiver has no power to decide which message he wants to receive, and the sender is not able to know which message is known by the receiver.

OK → SR-OT

Receiver input: b
Oblivious key pair: $(k^S, (k^R, x^R))$

(Transfer phase)

1. R defines $I_0 = \{i : x_i^R = 0\}$ and $I_1 = \{i : x_i^R = 1\}$ and sends I_b to S.

Sender output: $m_0 = k^S|_{I_b}$, $m_1 = k^S|_{I_{b\oplus 1}}$.
Receiver output: $m_b = k^R|_{I_0}$.

FIGURE 7 Sender random hybrid quantum oblivious transfer (SR-OT) phase

OK → RR-OT

Sender input: m_0, m_1 ($l-$bit strings).
Oblivious key pair: $(k^S, (k^R, x^R))$.

(Transfer phase)

1. R defines $I_0 = \{i : x_i^R = 0\}$ and $I_1 = \{i : x_i^R = 1\}$ and sends $I^j_{\min}$ to S.
2. S sets the pair (e_0, e_1) as $e_0 = m_0 \oplus k^S|_{I^j_{\min}}$, $e_1 = m_1 \oplus k^S|_{\overline{I^j_{\min}}}$ and sends it to R.
3. R computes $m_j = e_j \oplus k^R|_{I_0}$.

Sender output: $\perp$.
Receiver output: bit choice $b = j$ and message m_b.

FIGURE 8 Receiver random hybrid quantum oblivious transfer (RR-OT) phase

3.7 | Random hybrid quantum OT

We could join both SR-OT and RR-OT protocols and use oblivious keys to generate a completely random OT (R-OT) for both parties. In this random version, the sender does not choose his messages, and the receiver does not choose the message. The sender outputs both messages m_0 and m_1, and the receiver outputs the bit choice b and the corresponding message m_b. This protocol is described in Figure 9, and extensions and transfers are addressed in Figure 10.

3.8 | Hybrid quantum OT flavour comparison

In Sections 3.3–3.7, we presented several versions of OT protocols, all based on oblivious keys. It is important to note that C-OT and the randomised protocols are tailored to

OK → R-OT

Oblivious key pair: $(k^S, (k^R, x^R))$.

(Transfer phase)

1. R defines $I_0 = \{i : x_i^R = 0\}$ and $I_1 = \{i : x_i^R = 1\}$ and sends $I_{\min}^j$ to S.

Sender output: $m_0 = k^S|_{I_{\min}^j}$, $m_1 = k^S|_{\overline{I_{\min}^j}}$.
Receiver output: bit choice $b = j$ and message $m_b = k^R|_{I_0}$.

FIGURE 9 Random hybrid quantum oblivious transfer (R-OT) phase

General OT extensions protocol

Sender input: m pairs (x_j^0, x_j^1), $\forall 1 \leq j \leq m$ of l-bit strings.
Receiver input: m selection bits $\mathbf{r} = (r_1, ..., r_m)$.

Initial OT phase (Precomputation phase)

1. S randomly generates a string $\mathbf{s} = (s_1, ..., s_\kappa)$.
2. R randomly chooses κ pairs of κ-bit strings $\{(\mathbf{k}_i^0, \mathbf{k}_i^1)\}_{i=1}^{\kappa}$.
3. R and S execute κ base OTs, where S plays the role of the receiver with input $\mathbf{s}$ and R plays the role of the sender with messages $(\mathbf{k}_i^0, \mathbf{k}_i^1)$ $\forall 1 \leq i \leq \kappa$.

OT extension phase (Transfer phase)

4. R applies a pseudorandom number generator G to $\mathbf{k}_i^0$, i.e. $\mathbf{t}^i = G(\mathbf{k}_i^0)$. Computes $\mathbf{u}^i = \mathbf{t}^i \oplus G(\mathbf{k}_i^1) \oplus \mathbf{r}$ and sends $\mathbf{u}^i$ to S for every $1 \leq i \leq \kappa$.
5. S computes $\mathbf{q}^i = (s_i \cdot \mathbf{u}^i) \oplus G(\mathbf{k}_i^{s_i})$.
6. S sends (y_j^0, y_j^1) for every $1 \leq j \leq m$, where $y_j^0 = x_j^0 \oplus H(j, \mathbf{q}_j)$, $y_j^1 = x_j^1 \oplus H(j, \mathbf{q}_j \oplus \mathbf{s})$ and $\mathbf{q}_j$ is the j-th row of the matrix $Q = [\mathbf{q}^1|...|\mathbf{q}^\kappa]$. Note that, in practice, it is required to transpose Q to access its j-th row.
7. R computes $x_j^{r_j} = y_j^{r_j} \oplus H(j, \mathbf{t}_j)$.

Sender output: $\perp$.
Receiver output: $(x_1^{r_1}, ..., x_m^{r_m})$.

FIGURE 10 Precomputation and transfer phases of oblivious transfer (OT) extension protocol presented in [12]

specific SMC protocols and do not fit a general scenario. In the Yao GC framework, both C-OT and SR-OT versions can only be used if the *OT* phase is executed before the *wire encryption* phase. In addition, the R-OT version can be used in the GMW protocol [6] to precompute multiplication triples [62]. Let us compare the computational and communicational complexity of each version for m executions of OT. This is presented in Tables 2 and 3, respectively.

These reductions were also achieved by the optimisations proposed in [12, 62], but here we use oblivious keys as our main resource. Note that we assume that b bit choice is uniformly distributed in the C-OT protocol. Thus, with probability 1/2, the receiver performs one l−bit XOR operation, leading to $5/2\ ml$ bit XOR operations in m executions of the protocol. In addition, from both Tables 2 and 3, we conclude that SR-OT is the OT version with the least computational and communicational complexity, as it only requires 5 ml bit operations and ml bits during communication. The following section will compare the general O-OT protocol with classical base OT and extended OT protocols.

4 | OBLIVIOUS TRANSFER COMPLEXITY ANALYSIS

In this section, we compare the complexity of the optimised O-OT version of HQOT presented before with well-known classical protocols. The O-OT protocol is divided into two phases: the oblivious key distribution phase (we also call it a *precomputation* phase) and the transfer phase. The first phase of O-OT is similar to the HQOT oblivious key phase [11] presented in Section 3.2, but the second phase follows the protocol shown in Section 3.3. To fairly compare the O-OT protocol with other protocols, we must divide them into these two phases. We apply the following rule: all the steps used in the protocol that are independent of the messages (m_0 and m_1) and of the bit choice (b) are considered to be part of the precomputation phase; otherwise, they are included in the transfer phase. Since the precomputation phase can be implemented before executing the Yao GC protocol, it is more important to guarantee that the transfer phase has small complexity. Furthermore, here we will only compare the complexity among the different protocols' transfer phases because their precomputation phases rely on different technologies. Because quantum technologies are still in their infancy and constantly evolving, it is difficult to compare the efficiency with classical approaches. Nevertheless, it is worth stressing that the oblivious key phase of O-OT protocol is linear in all its security parameters. In fact, as presented in [11], its time complexity is of order $\mathcal{O}(k(2l + t))$, where k is the security parameter of the hash-based commitments, $2l$ is the number of qubits sent used to generate the oblivious keys directly, and t is the number of testing qubits.

4.1 | Classical OT

In Section 2, we divided the well-known Bellare–Micali protocol into two phases and concluded that it requires three exponentiations during the transfer phase. In Table 4, we present the number of required modular exponentiations and communication rounds during the transfer phase of four well-known classical protocols that have their security based on the computational hardness of the discrete logarithm problem.

From Table 4, we see that the most efficient protocol (SimpleOT [16]) still requires one exponentiation operations and two communication rounds. From the above Equation (1) and setting $C_{mcyles} = 2.5 \times 10^9$, $C_{mexp} = 400\ 000$, and $N_{mexp} = 1$, we obtain an overestimation of around 6000 OTs per second. Comparing with the rate achieved by OT extension protocols (10 million OT in 2.62), it is still very inefficient. In

TABLE 2 Computational complexity of several OT flavours based on oblivious keys

Operation	OT	O-OT	C-OT	SR-OT	RR-OT	R-OT
Bit-wise comparison	2 ml	2 ml	2 ml	2 ml	2 ml + m	2 ml + m
Bit-wise truncation	3 ml	3 ml	3 ml	3 ml	3 ml	3 ml
Bit-wise XOR	3 ml	3 ml	$\frac{5}{2}$ ml	0	3 ml	0

Abbreviations: C-OT, correlated HQOT; HQOT, hybrid quantum OT; OT, oblivious transfer; O-OT, optimised HQOT; R-OT, random HQOT; RR-OT, receiver random HQOT; SR-OT, sender random HQOT.

TABLE 3 Communicational complexity of several OT flavours based on oblivious keys

	OT	O-OT	C-OT	SR-OT	RR-OT	R-OT
Bits sent	4 ml	3 ml	2 ml	ml	3 ml	ml

Abbreviations: C-OT, correlated HQOT; HQOT, hybrid quantum OT; OT, oblivious transfer; O-OT, optimised HQOT; R-OT, random HQOT; RR-OT, receiver random HQOT; SR-OT, sender random HQOT.

TABLE 4 Number of modular exponentiation operations and communication rounds executed during the transfer phase of several classical protocols

Protocol	Exponentiation	Comm. rounds
EGL [13]	3	2
BM [14]	3	2
NP [15]	2	2
SimpleOT [16]	1	2

Abbreviations: BM, Bellare–Micali; EGL, Even, Goldreich and Lempel; NP, Naor and Pinkas; OT, oblivious transfer.

addition, note that the number of modular exponentiations does not decrease in their random versions.

This means that the current classical OT protocols have a computational complexity limited by $\mathcal{O}(n^{2.58})$ bit operations due to modular exponentiation. The O-OT protocols only depends on simple bit operations (XOR, truncation, and comparison), meaning its computational complexity is linear in the length of the messages $\mathcal{O}(n)$.

In addition, it is important to stress that none of the above protocols are secure against quantum computer attacks. To have classical OT protocols with this level of security, we need to follow post-quantum approaches that may lead to more demanding operations [63]. As reported in [64], using module learning with errors-based Kyber key encapsulation [65], it takes 24 ms to generate one OT in a LAN network. This leads to a rate of just 41 OT per second, which is even lower than the rate reported by [12] for the NP [15]: 56 OT per second. In [66, 67], the authors present a 1-out-of-n OT based on the NTRU post-quantum encryption system [68] and compare it with the SimpleOT [16] version. In this case, although the sender and receiver phases are more individually efficient, the overall NTRU OT protocol is still less efficient. For the highest security level, it takes around 1.372 ms to generate one OT with the post-quantum approach, whereas it takes 0.727 ms using the original simpleOT protocol. These timings lead to rates of 728 and 1375 OTs per second, respectively. It is important to note that these protocols are still prone to *intercept now–decipher later* attacks because they are based on computational assumptions that are only *believed* (and not proved) to be secure against quantum computer attacks.

4.2 | OT extension

As we explained in Section 2.3, several techniques based on a hybrid symmetric-asymmetric approach were developed as a way to increase the OT execution rate. These techniques use a small number κ (=128) of base OT protocols and extend this resource to m (=10,000,000) OT executions, where $m \gg \kappa$.

Again, to compare the OT extension approach with the O-OT, the OT extension protocols must be decomposed into two phases: precomputation phase and transfer phase. Here we make a bit-wise comparison of the communication and computational complexity of m O-OT and one OT extension because OT extension protocols execute a predetermined number (m) of OTs at once.

Let us consider the OT extension protocol proposed in [12] (ALSZ13), which, at the time of writing, reports the fastest implementation: 10 million OT in 2.68 s. This protocol is originally divided into two phases: initial OT phase and OT extension phase. Note that these two phases correspond exactly to our division of precomputation and transfer phases. Thus, for comparative purposes, we are only interested in the second phase. In Tables 5 and 6, we show the computational and communicational complexity of both protocols, respectively.

In Table 5, PRG stands for pseudorandom generator, κ represents the number of base OTs executed in the OT extension precomputation phase, m is the number of final OTs, and l is the length of the OT strings. In addition, we consider that $l \sim \kappa$ have the same order of magnitude, meaning they represent the same cost of bits. This is so because $\kappa = 128$ in [12] and the key length used in the garbled circuits are $l = 128$, 192, or 256. The red coloured text in the table represents the operations omitted in a sender randomised version. So we easily see that the Sender Random OT extension protocol do not gain any advantage over the SR-OT because they correspond to the same reduction in the number of bit operation and communication.

We have that the communicational complexity is exactly the same in both protocols: $\sim 3\,ml$. So the OT extension does

TABLE 5 Computational complexity comparison between ALSZ13 [12] OT extension protocol and O-OT from Section 3.3

Operation	ALSZ13	O-OT
PRG (AES)	3κ	-
Hash (SHA-1)	$3m$	-
Bit-wise XOR	$3\kappa m + 3\ ml + \kappa$	$3ml$
Bit-wise AND	κm	-
Matrix transposition	$m \log\ m$	-
Bit-wise comparison	-	$2ml$
Bit-wise truncation	-	$3ml$

Abbreviations: OT, oblivious transfer; O-OT, optimised hybrid quantum OT; PRG, pseudorandom generator.

TABLE 6 Communicational complexity comparison between ALSZ13 [12] OT extension protocol and O-OT from Section 3.3

	ALSZ13	O-OT
Bits sent	$2ml + m\kappa$	$2ml + ml$

Abbreviations: OT, oblivious transfer; O-OT, optimised hybrid quantum OT.

not have any advantage over O-OT during the communication phase. Regarding their computational complexity, we have to compare the cost of ml bit operations in O-OT against 3κ AES, $3m$ SHA-1 and $\kappa + m \log\ m$ bit operations in ALZS13 OT extension.

Firstly, we can conclude that O-OT transfer phase is asymptotically more efficient than OT extension transfer phase. The computational complexity of OT extension is not linear in the number of OT executions, $\mathcal{O}(m\log m)$, whereas it is linear in the case of O-OT, $\mathcal{O}(m)$. Thus, for a bounded value l, $m \log\ m > ml$ asymptotically. However, in practice, for $l \sim 128$, it would take around $m > 10^{50}$ OTs to actually have $m \log\ m > ml$. Secondly, when comparing ml bit operations with $3m$ SHA−1 computations, we conclude that O-OT uses less bit operations. Each execution of SHA−1 [69] requires 80 rounds of several bit operations. If we underestimate the cost of each round and we assume it only requires two bit operations, we determine that ALSZ13 OT extension protocol requires at least $3 \times 80 \times 2\ \mathrm{m} = 480\ \mathrm{m} > ml, \forall m$.

From this, we conclude that O-OT transfer phase competes with the ALSZ13 corresponding phase and has the potential to be more efficient. It is important to stress that O-OT efficiency performance of the transfer phase comes along with a drastic increase in the security of the protocol: while ALSZ13 protocol relies on computational assumptions of the base OT, O-OT is proved to be secure against quantum computers; while ALSZ13 is a semi-honest protocol (assumes well-behaved parties that follow the protocol), O-OT is secure against any corrupted party. Indeed, to obtain a fair comparison, we should consider OT extension protocols that are secure against malicious parties. The work developed in [27] presented for the first time a protocol in the malicious scenario, which was latter optimized by KOS15 [31] and ALSZ15 [30]. Both optimisations carry out one run of the semi-honest OT extension presented in ALSZ13 plus some consistency checks. The protocol presented in [31] adds to ALSZ13 a *check correlation* phase after the transfer phase, and the protocol presented in [30] adds a *consistency check* phase during the transfer phase. This means that both malicious protocols' transfer phases have greater computational and communicational complexity when compared with ALSZ13. Therefore, we can easily deduce that the O-OT transfer phase has less computational and communicational complexity than its classical equivalents with respect to the adversary model used.

5 | CONCLUSION

The importance of secure and efficient implementations of OT is crucial for several secure computations. In particular, the Yao GC protocol security and efficiency are deeply connected with OT security and efficiency. However, Impagliazzo and Rudich's result enforces classical OT protocols founded on asymmetric cryptographic primitives that are known to be insecure against quantum attacks, or their resistance is derived from weak conjectures. On the other hand, in [11], the authors used the laws of physics to prove the HQOT protocol to be secure against malicious adversaries with access to a quantum computer.

Here, we compared an O-OT of HQOT protocol transfer phase with the transfer phase of the protocol that to the best of our knowledge, is the fastest implementation of OT [12]. We concluded that the O-OT transfer phase has the potential to be faster than the ALSZ13 OT extension transfer phase while preserving much higher security. The ALSZ13 protocol has only been proved to be secure in the semi-honest model, while O-OT is secure in the malicious setting. In addition, we concluded that the transfer phase of current maliciously secure implementations [30, 31] has greater computational and communicational complexity than the O-OT transfer phase.

Furthermore, we hold that oblivious keys allow separation of the quantum technological burden from the execution of the OT and can be used to efficiently implement several types of OTs: general OT, correlated OT, receiver random OT, sender random OT, and random OT. It is important to note that to the best of our knowledge, there are no classical methods that can securely generate oblivious keys, leaving its distribution to quantum methods.

ACKNOWLEDGEMENT

This project is supported by the European Regional Development Fund (FEDER), through the Competitiveness and Internationalization Operational Programme (COMPETE 2020) of the Portugal 2020 framework (Project Q.DOT with No. 039728 [POCI-01-0247-FEDER-039728]), and by FCT/MCTES through national funds and when applicable co-funded EU funds under UIDB/50008/2020-UIDP/50008/2020 (action QUESTS) and SFRH/BD/144806/2019.

ORCID

Manuel B. Santos https://orcid.org/0000-0003-4156-8607

REFERENCES

1. Rabin, M.: How to Exchange Secrets with Oblivious Transfer. IACR Cryptology ePrint Archive (1981)
2. Pinkas, B., Schneider, T., Zohner, M.: Scalable private set intersection based on ot extension. ACM Trans. Priv. Secur. 21(2) (2018)
3. Kolesnikov, V., et al.: Scalable private set union from symmetric-key techniques. In: Galbraith, S. D., Moriai, S. (eds.) Advances in Cryptology – ASIACRYPT 2019. Springer International Publishing, Cham, pp. 636–666 (2019)
4. Naor, M., Pinkas, B.: Oblivious transfer and polynomial evaluation. In: Proceedings of the Thirty-First Annual ACM Symposium on Theory of Computing (STOC 99), pp. 245–254. Association for Computing Machinery, New York (1999)
5. Yao, A.C.: Protocols for secure computations. In: Proceedings of the 23rd Annual Symposium on Foundations of Computer Science (SFCS), pp. 160–164. IEEE, Chicago (1982)
6. Goldreich, O., Micali, S., Wigderson, A.: How to play any mental game. In: Proceedings of the Nineteenth Annual ACM Symposium on Theory of Computing, STOC 87, pp. 218–229. Association for Computing Machinery, New York (1987)
7. Damgård, I., et al.: Multiparty computation from somewhat homomorphic encryption. In: Safavi-Naini, R., Ran, C. (eds.) Advances in Cryptology – CRYPTO 2012, pp. 643–662. Springer, Berlin (2012)
8. Bennett, C., et al.: Practical quantum oblivious transfer. 576, 351–366 (1991)
9. Claude, C.: Quantum oblivious transfer. J. Mod. Opt. 41(12), 2445–2454 (1994)
10. Shor, P.W.: Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. SIAM J. Comput. 26(5), 1484–1509 (1997)
11. Lemus, M., et al.: Generation and distribution of quantum oblivious keys for secure multiparty computation. Appl. Sci. 10(12), 4080 (2020). http://doi.org/10.3390/app10124080
12. Asharov, G., et al.: More efficient oblivious transfer and extensions for faster secure computation. In: Proceedings of the 2013 ACM SIGSAC Conference on Computer & Communications Security (CCS 13), pp. 535–548. Association for Computing Machinery, New York (2020)
13. Even, S., Goldreich, O., Lempel, A.: A randomized protocol for signing contracts. Commun. ACM. 28(6), 637–647 (1985)
14. Bellare, M., Micali, S.: Non-interactive oblivious transfer and applications. In: Proceedings on Advances in Cryptology, CRYPTO -89, pp. 547–557. Springer-Verlag, Berlin (1989)
15. Naor, M., Pinkas, B.: Efficient oblivious transfer protocols. In: Proceedings of the twelfth Annual ACM-SIAM Symposium on Discrete Algorithms SODA 01. Washington, D.C (2001)
16. Chou, T., Orlandi, C.: The simplest protocol for oblivious transfer. In: Proceedings of the 4th International Conference on Progress in Cryptology – LATINCRYPT 2015 - Volume 9230, page 40–58. Springer-Verlag, Berlin (2015)
17. Pohlig, S., Hellman, M.: An improved algorithm for computing logarithms overGF(p) and its cryptographic significance (Corresp.). IEEE Trans. Inf. Theor. 24(1), 106–110 (1978)
18. Shanks, D.: Class number, a theory of factorization and genera. In: Proceedings of the Symposium on Pure Mathematics, pp. 415–440. American Mathematical Society, Providence (2019)
19. Pollard, J.M.: Monte Carlo methods for index computation (mod p). Math. Comput. 32, 918–924 (1978)
20. Adrian, D., et al.: Imperfect forward secrecy: How diffie-hellman fails in practice. In: Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, CCS 15, pp. 5–17. Association for Computing Machinery, New York (2015)
21. Bernstein, D.J., Lange, T.: Post-quantum cryptography. Nature. 549(7671), 188–194 (2015)
22. Pirandola, S., et al.: Advances in quantum cryptography. Adv. Opt. Photon. 12(4), 1012–1236 (2020)
23. Menezes, A.J., Van Oorschot, P.C., Vanstone S.A.: Handbook of Applied Cryptography. CRC Press, Boca Raton (2001). https://www.routledge.com/Handbook-of-Applied-Cryptography/Menezes-van-Oorschot-Vanstone/p/book/9780849385230
24. Gueron, S.: Efficient software implementations of modular exponentiation. IACR Cryptology ePrint Archive. 239, 01 (2011)
25. Russell, I., Rudich, S.: Limits on the Provable Consequences of One-way Permutations. [*No source information available*], 03 (1999)
26. Beaver, D.: Correlated pseudorandomness and the complexity of private computations. In: Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing STOC 96, pp. 479–488. Association for Computing Machinery, New York (1996)
27. Ishai, Y., et al.: Extending oblivious transfers efficiently. In: Boneh, Dan (eds.), Advances in Cryptology - CRYPTO 2003. Lecture Notes in Computer Science, vol. 2729, pp. 145–161. Springer, Berlin (2003). https://link.springer.com/chapter/10.1007/978-3-540-45146-4_9
28. Nielsen, J.: Extending oblivious transfers efficiently - how to get robustness almost for free. IACR Cryptology ePrint Archive. 215, 01 (2007)
29. Nielsen, J.B., et al.: A new approach to practical active-secure two-party computation. In: Safavi-Naini, R. (eds.) Advances in Cryptology - CRYPTO 2012. Lecture Notes in Computer Science, vol. 7417, pp. 681–700. Springer, Berlin (2012). https://link.springer.com/chapter/10.1007/978-3-642-32009-5_40
30. Asharov, G., et al.: More efficient oblivious transfer extensions with security for malicious adversaries. In: Oswald, E., Fischlin, M. Advances in Cryptology – EUROCRYPT 2015, pp. 673–701. Springer, Berlin (2015)
31. Keller, M., Orsini, E., Scholl, P.: Actively secure OT extension with optimal overhead. In: *Advances in Cryptology – CRYPTO 2015*, volume 9215 of *Lecture Notes in Computer Science*, pp. 724–741. Springer (2015)
32. Claude, C.: Equivalence between two flavours of oblivious transfers. In: Pomerance, C. (ed.) *Advances in Cryptology – CRYPTO '87*, pp. 350–354.Springer Berlin Heidelberg, Berlin (1988)
33. Mayers, D.: The Trouble with Quantum Bit Commitment. arXiv *preprint quant-ph/9603015* (1996)
34. Mayers, D.: Unconditionally secure quantum bit commitment is impossible. Phys. Rev. Lett. 78(17), 3414–3417 (1997)
35. Lo H.-K., Chau H. F.: Is Quantum Bit Commitment Really Possible? Phys. Rev. Lett. 78(17), 3410–3413 (1997). https://doi.org/10.1103/physrevlett.78.3410
36. Almeida, Á., et al.: Implementation of a two-state quantum bit commitment protocol in optical fibers. J. Opt. 18, 015202 (2015)
37. Koenig, R., Wehner, S., Wullschleger, J.: Unconditional security from noisy quantum storage. IEEE Trans. Inf. Theor. 58 (2009)
38. Loura, R., et al.: Noise and measurement errors in a practical two-state quantum bit commitment protocol. Phys. Rev. A. 89, 052336 (2015)
39. Souto, A., et al.: Bit-string oblivious transfer based on quantum state computational distinguishability. Phys. Rev. A. 91, 042306 (2015)
40. Rodrigues, J., et al.: Oblivious transfer based on single-qubit rotations. J. Phys. Math. Theor. 50(20), 205301 (2017)
41. Unruh, D.: Universally composable quantum multi-party computation. In: Gilbert, H. (ed.) Advances in Cryptology – EUROCRYPT 2010, pp. 486–505. Springer, Berlin (2010)
42. Aaronson, S., Shi, Y.: Quantum lower bounds for the collision and the element distinctness problems. J ACM 51(4). 595–605 (2004). http://doi.org/10.1145/1008731.1008735
43. Aggarwal, D., et al.: Quantum Attacks on Bitcoin, and How to Protect Against Them. arXiv *e-prints*, page arXiv: 1710.10377 (2017)
44. Halevi, S., Micali, S.: Practical and provably-secure commitment schemes from collision-free hashing. In: Proceedings of the 16th Annual International Cryptology Conference on Advances in Cryptology, CRYPTO 96, pp. 201–215. Springer-Verlag, Berlin (1996)
45. Gao, F., et al.: Flexible quantum private queries based on quantum key distribution. Opt. Express. 20(16), 17411–17420 (2012)

46. Rao, M.P., Jakobi, M.: Towards communication-efficient quantum oblivious key distribution. Phys. Rev. A. 87(1), (2013). http://doi.org/10.1103/physreva.87.012331
47. Zhang, J.-L., et al.: Private database queries based on counterfactual quantum key distribution. Phys. Rev. 88(2), 022334 (2012)
48. Wei, C.-Y., et al.: Practical quantum private query of blocks based on unbalanced-state bennett-brassard-1984 quantum-key-distribution protocol. Sci. Rep. 4, 7537 (2012)
49. Chan, P., et al.: Performing private database queries in a real-world environment using a quantum protocol. Sci. Rep. 4(1), (2015). http://doi.org/10.1038/srep05233
50. Gao, F., et al.: Postprocessing of the oblivious key in quantum private query. IEEE J. Sel. Top. Quant. Electron. 21(3), 98–108 (2015)
51. Liu, B., et al.: Qkd-based quantum private query without a failure probability. Sci. China Phys. Mech. Astron. 58, 10 (2015)
52. Fang, Y., et al.: Enhancing user privacy in SARG04-based private database query protocols. Quant. Inf. Process. 14(11), 4201–4210 (2015)
53. Zhou, Y.-H., et al.: A quantum private query protocol for enhancing both user and database privacy. Commun. Theor. Phy. 69(1), 31 (2018)
54. Wei, C.-Y., et al.: A generic construction of quantum-oblivious-key-transfer-based private query with ideal database security and zero failure. IEEE Trans. Comput. 67(1), 2–8 (2018)
55. Wei, C.-Y., Wang, T.-Y., Gao, F.: Practical quantum private query with better performance in resisting joint-measurement attack. Phys. Rev. 93(4), 042318 (2016)
56. Yang, Y.-G., et al.: Quantum private query with perfect user privacy against a joint-measurement attack. Phys. Lett. 380(48), 4033–4038 (2016)
57. Chang, Y., et al.: Practical two-way qkd-based quantum private query with better performance in user privacy. Int. J. Theor. Phys. 58, 05 (2016)
58. Du, Z., Li, X.: Robust high capability QKD-based database private query. Int. J. Theor. Phys. 58(2), 391–398 (2019)
59. Ye, T.-Y., Li, H.-K., Hu, J.-L.: Semi-quantum private query protocol without invoking the measurement capability of classical user. Int. J. Theor. Phys. 59(7), 2044–2051 (2020). http://doi.org/10.1007/s10773-020-04476-3
60. Koenig, R., Wehner, S., Wullschleger, J.: Unconditional security from noisy quantum storage. IEEE Trans. Inf. Theor. 58, 1962–1984 (2012)
61. Jakobi, M., et al.: Practical private database queries based on a quantum-key-distribution protocol. Phys. Rev. A. 83, 022301 (2011)
62. Asharov, G., et al.: More efficient oblivious transfer extensions. J. Cryptol. 30(3): 805–858 (2017)
63. Paquin, C., Stebila, D., Tamvada, G.: Benchmarking post-quantum cryptography in tls. In: Proceedings of the International Conference on Post-Quantum Cryptography, pp. 72–91. Springer (2020)
64. Mansy, D., Rindal, P.: Endemic Oblivious Transfer. CCS 19, pp. 309–326. Association for Computing Machinery, New York (2019)
65. Avanzi, R., et al.: Crystals-kyber. *NIST*. Technical Report (2017)
66. Mi, B., et al.: Oblivious transfer based on ntruencrypt. IEEE Access. 6, 35283–35291 (2018)
67. Mi, B., et al.: A post-quantum light weight 1-out-n oblivious transfer protocol. Comput. Electr. Eng. 75, 90–100 (2019)
68. Hirschhorn, P.S., et al.: Choosing ntruencrypt parameters in light of combined lattice reduction and mitm approaches. In: Abdalla, M., et al. (eds.) Applied Cryptography and Network Security, pp. 437–455. Springer, Berlin (2009)
69. National Institute of Standards and Technology: Secure Hash Standard (shs), pp. 180–184. Federal Information Processing Standards Publications (FIPS PUBS), Washington, D.C. U.S. Department of Commerce (2015)

How to cite this article: Santos MB, Pinto AN, Mateus P. Quantum and classical oblivious transfer: A comparative analysis. *IET Quant. Comm*. 2021;2:42–53. https://doi.org/10.1049/qtc2.12010