Commun. Theor. Phys. **64** (2015) 299–304 Vol. 64, No. 3, September 1, 2015

Multi-Bit Quantum Private Query

SHI Wei-Xu (石惟旭),* LIU Xing-Tong (刘星彤), WANG Jian (王剑), and TANG Chao-Jing (唐朝京)
Information Security Laboratory, School of Electronic Science and Engineering, National University of Defense Technology of China, Changsha 410073, China

(Received April 15, 2015; revised manuscript received June 2, 2015)

Abstract *Most of the existing Quantum Private Queries (QPQ) protocols provide only single-bit queries service, thus have to be repeated several times when more bits are retrieved. Wei et al.'s scheme for block queries requires a high-dimension quantum key distribution system to sustain, which is still restricted in the laboratory. Here, based on Markus Jakobi et al.'s single-bit QPQ protocol, we propose a multi-bit quantum private query protocol, in which the user can get access to several bits within one single query. We also extend the proposed protocol to block queries, using a binary matrix to guard database security. Analysis in this paper shows that our protocol has better communication complexity, implementability and can achieve a considerable level of security.*

PACS numbers: 03.67.Dd, 03.67.Hk
Key words: quantum communication, quantum cryptography

1 Introduction

As electric commerce enjoys increasing popularity and diversity, security has become a major concern of both sides of the transaction. A wide range of cryptography techniques are invented to meet the security demands of different occasions. In the transaction between a data seller (Bob) who owns a database and a user (Alice) who is charged for the data read from the seller's database, the two parties may infringe upon each other's individual privacy to obtain extra profit. For example, Alice's retrieval records may leak about her intention on a significant decision-making, which can be exploited by Bob for some benefits. Alice, on the other hand, could peep at other unpaid items in the database during the interaction. Quantum Private Queries (QPQ), one of the many novel quantum protocols designed on quantum cryptographic primitives, serves the above circumstance, in the way that a user is allowed to retrieve an item from a database without knowing more than this item (database security), and the database provider could not learn which item he/she is interested in (user privacy).

The notion of Private Information Retrieval (PIR) was introduced by Chor *et al.*, which allows Alice to retrieve a bit from a data string without disclosing her retrieving address to Bob.[1] Yael Gertner generalized the guarding shield to database privacy and proposed Symmetrical Private Information Retrieval (SPIR) scheme.[2] The above two schemes need more than one database in the query interaction and can tolerate no direct communication between the databases, which is hard to guarantee in the real world. Ten years later, a quantum version SPIR scheme was proposed by Giovannetti *et al.* in Ref. [3], with its security analysis in Ref. [4]. In this scheme, a cheat-sensitive model is used to check whether Bob is honest, and in one query Alice can just read one item from the measurement on Bob's response. The scheme needs a quantum computer at Bob's side. Though quantum computers that can implement efficient database search has been devised,[5−6] the implementability of the scheme is largely reduced because it is not loss tolerant. For instance, Alice could gain more items by sending a superposed query state that can bring back the superposition of multiple query result, then claiming that she has not received the answer because of the loss and requiring Bob to resend it.

A much more practical QPQ model is later created by Jakobi *et al.* in Ref. [7], based on a quantum cryptographic primitive, Quantum Key Distribution (QKD). Some improvements have been made to Jakobi's protocol. Gao *et al.* gave more freedom to the variables (e.g. N, k) to meet the requirement of the final key by adding a parameter θ.[8] Two novel error correction methods[9−10] for quantum private queries made Jakobi's model fault-tolerant, which enhances its implementability. However, none of the above protocols suits the situation when Alice wants to retrieve several bits in one query. The fact that in some occasions, only consecutive data constitutes a meaningful message and that condensing the rounds of single-bit retrieval may bring more efficiency is calling for a multi-bit QPQ protocol.

Let us look through two scenarios that Alice wants to retrieve n (> 1) bits. Scenario 1 is that Alice is charged by the amount of data she wants to retrieve, and these data bits are scattering around the database. Scenario 2 is that Bob sells the data block by block. These blocks are of the same size and each as an integral record. In this scenario, simply repeating a single-bit QPQ protocol cannot guarantee database security, for Alice can violate the

*Corresponding author, E-mail: shiweixu@126.com

block reading rule for the sake of the privacy that the protocol grants her. For the second scenario, a block query scheme, Quantum Private Queries of Blocks (QPQB), is proposed by Wei *et al.* in Ref. [11]. Basing the queries on high-dimension BB84 QKD protocol[12] enforces block retrieval, while the technique of unbalanced-states keeps the effect of cheating strategies of the two parties subtle. However, high-dimension QKD systems applied in QPQB are far less implementable than their 2-dimension counterparts.

Here we propose a multi-bit quantum private query protocol, based on Jakobi's protocol. It allows retrieval of several bits in one query, so can apply to Scenario1 efficiently. It can also be adopted for block queries (Scenario 2) with a minimal modification on the items stored in the database, while retain a considerable security level.

2 Multi-Bit QPQ Protocol

Since our multi-bit protocol is the generalization of Jakobi's model, we review its primitive here. Assume that N items are stored in Bob's database in the form of single bit. The protocol can be divided into the following steps:

(i) Alice and Bob transfer a string of oblivious key according to a modified SARG-04 QKD protocol[13] as follows. Bob sends a long sequence of random bits randomly coded in two bases, $\{|0\rangle, |1\rangle\}$ and $\{|+\rangle, |-\rangle\}$ (say one of the qubits being $|1\rangle$). Alice measures each of these states in 01-basis or $\pm$-basis randomly. The two parties then discard the missing bits according to Alice's announcement. For each left bits, Bob announces a pair of states, consist of the state he sent and an arbitrary state from the other basis (in this case, assume that Bob announces $|1\rangle$ and $|-\rangle$). Through the announced pair of states for one bit, Alice knows whether she gets a conclusive result of that bit. In our example, if she obtains $|1\rangle$ or $|-\rangle$, she gets no knowledge about this qubit (inconclusive result); if $|+\rangle$ is obtained, she can be sure that Bob sent $|1\rangle$ (conclusive result). If the qubit is transmitted correctly, she should have no chance to obtain $|+\rangle$. In this way, Alice and Bob share a kN-length oblivious key (which means that Bob knows the key entirely, whereas Alice only knows a part of the key randomly, but Bob does not know which bits are known to Alice).

(ii) The oblivious key string is cut into k substrings of length N. These substrings are added bitwise at each end to create the final key K^f, in order to reduce Alice's knowledge to roughly one bit in K^f. If Alice losses all knowledge about K^f after bitwise adding, the protocol has to be restarted.

(iii) Alice announces the interval s between the locations of her retrieving bit in the database and her conclusive bit in K^f.

(iv) Bob shifts his K^f by s and encodes the database by bitwise adding the shifted K^f.

(v) Finally, Alice retrieves her bit from the encoded database by bitwise adding her shifted K^f.

From the above description, it is apparent that Jakobi's protocol only provides single-bit retrieval in one query, because the number of conclusive bits restricts the number of Alice's accessible bits. Aiming at increasing the number of retrieving bits in one query, we loosen Alice's knowledge about K^f. One can adopt Gao's scheme[8] to flexibly control the number of conclusive bits. It introduces a variable θ to allow a more "constant" control of the number of bits known to Alice. For simplification, the modification on Jakobi's protocol proposed by Gao has been omitted in the following statement.

We will first look at Scenario 1, which is simpler. Scenario 2 will be later discussion section 4. Alice has bought n bits from Bob's database X, and she wants to retrieve all these bits within one query in some consideration of efficiency. We generalize Jakobi's protocol by adding a permutation that Bob is to operate as Alice announces. The permutation moves Alice's conclusive key bits in her final key to her retrieving addresses in the database; still it conceals Alice's intention from Bob even when Bob does the same to his final key (as will be seen in Subsec. 3.1(ii)). Our protocol consists of the following steps:

(a) As instructed in (i), (ii) of the initial protocol, Alice and Bob share an oblivious key K^f of length N, this time with n bits of K^f known to Alice.

(b) Suppose that Alice wants the x_1-th, x_2-th, ..., x_n-th bits of the database, namely X_{x_1}, X_{x_2}, ..., X_{x_n}, and she knows the k_1-th, k_2-th, ..., k_n-th bits of the key, namely K_{k_1}, K_{k_2}, ..., K_{k_n}. She constructs a permutation matrix P which moves K_{k_1}, K_{k_2}, ..., K_{k_n} to the x_1-th, x_2-th, ..., x_n-th positions of K^f, and announces P.

(c) Bob computes $K_P^f = PK^f$ and then he encrypts X by bitwise adding K_P^f as $C = X \oplus K_P^f$. He sends the encrypted database C.

(d) Alice obtains from C her bits of interest, $X'_{x_i} = C_{x_i} \oplus K_{x_i}$ $(i = 1, \ldots, n)$.

The shift operation in (iii), (iv) of the original protocol is replaced by a permutation in (b), (c), which allows the simultaneous motion of all bits in one interaction. The construction of the permutation matrix is simple. For example, one of Alice's conclusive bits is K_i and she wants to move it to the j-th unit, where stores one of her desired bits. All she has to do is to put the i-th row of an identity matrix to the j-th row of an initially empty P. This step is repeated until she has moved all the corresponding rows into P. Then she puts the remained rows randomly into P to fill it up. Multiplying P by a vector can reorder the elements of the vector as plan. An analogy might clarify the main difference between the ideas of original protocol and ours. The oblivious key is like a punctured paper band with several holes on it, and the user is supervised reading a row of sentence through this band. In Jakobi's

protocol, the shift operation is like moving the entire band to aim one of the hole at one particular word in a sentence. In our protocol, the band is cut into pieces, each of the size of a hole. Permutation means that Alice fits all the pieces together with the punctured ones onto the words she wants to read.

We suggest Alice follow the original Jakobi's protocol when only one bit is retrieved, because it provides better database security as discussed below.

3 Discussion

3.1 *Security Analysis*

As a generalization of Jakobi's protocol, the mere difference in oblivious key producing phase between two protocols is that our protocol requires more bits to be left after bitwise adding. The possible security problems that arise from the stage of producing the oblivious key have been discussed and proved in Ref. [7], thus would not be the chief part here. Our protocol has noteworthy modifications after the distribution of the oblivious key, so instead we will focus on new problems introduced by the modified steps.

(i) *Database Security*

In the single-bit retrieval scheme. Two basic attacks have been analyzed in Ref. [7], which are under the circumstance that Alice possesses a quantum memory and she postpones her measurements until Bob announces his sequence of state pairs. Database security is guarded by the limited performance of unambiguous state discrimination (USD) measurement[14−15] and Helstrom's minimal error-probability measurement.[16−17] However, like in most communication protocols, there exists a compromise between efficiency and security in our protocol. Usually, to increase the success possibility of the oblivious key distribution, the average number of conclusive bits is set larger than n. The number of conclusive bits approximately follows a Poisson distribution, whose variance grows as its mean grows. This may bring Alice surplus conclusive bits that she can dispose arbitrarily through the construction of the permutation matrix. Despite that, in Poisson distribution, the variance turns proportionally minor compared with an enormous mean (recalling that the peak of the distribution shape becomes more concentrated when the mean turns enormous). On the part of efficiency, as will be shown in the next subsection, as the amount of Alice retrieving data becomes larger, more transmitted bits can be saved using our protocol than repeating Jakobi's protocol.

(ii) *User Privacy*

In the phase of oblivious key distribution, user privacy is preserved by no-signaling principle, which grants the impossibility for Bob to know conclusiveness and value of a key bit simultaneously. Bob's knowledge of conclusiveness but not value of a key bit would introduce errors thus would be found cheating, while his mere knowledge of bit value would lead to his confusion about conclusiveness thus keep him away from Alice's privacy. Therefore, the user privacy in oblivious key distribution should not be a problem here.

Our emphasis is whether the permutation matrix constructed by Alice has the possibility of leaking about the querying address. Here, we argue show that this operation leaks no information about Alice's privacy. Suppose Alice constructs a permutation to put the n conclusive bits to the n locations where she wants to query. Let us call the distribution of Alice's retrieving bits the retrieving pattern, and the distribution of Alice's conclusive bits the conclusiveness pattern. If the permutation leaks some information about Alice's retrieving pattern, the retrieving patterns must not be equally possible, which means there must be more conclusiveness patterns that can be turned into one or more particular retrieving patterns by this permutation. However, it is easy to see that there are always equal-number conclusiveness patterns corresponding to each retrieving patterns, regardless of the permutation. Hence the probability that Bob could guess correctly which bits Alice is retrieving does not increase (still being $1/C_N^n$), even though he is announced P.

3.2 *Efficiency*

We now show that when a large number of items are retrieved from a database, our protocol requires less data to be transmitted on the channel. Generally, if she follows the initial Jakobi's protocol, there would be n rounds of communication. Moreover, in each round Bob has to send the entire encrypted database. These two facts not only construct operation burdens to both individuals, but also occupy much time. Our protocol reduces the number of query rounds to one as well as the total number of transmitted bits between the two parties.

More precisely, in our protocol, Alice sends $\log_2 N!$ bits to define the permutation matrix, and Bob sends the N-bit encrypted database, so in total there are $\log_2(N!)+N$ bits needed to be sent. In comparison, each round of Jakobi's protocol requires $\log_2 N$ bits to represent the shift s and N bits to represent Bob's encrypted database. Since Alice has to query n times, the transmitted data is of $n(\log_2 N + N)$ bits in total. Figure 1 has shown the total number of transmitted bits of two protocols. The advantage of the multi-bit protocol over n-turn Jakobi's protocol is evident when the quantity of retrieving bits grows larger. The line where the two planes roughly intersect marks where our protocol begins to exceed Jakobi's protocol.

Table 1 shows the some typical Ns and its smallest tolerable number of retrieving bits in the multi-bit protocol. For example, if the size of the database is 200 bits, only when retrieving more than 8 bits would the parties

transmit less bits using multi-bit protocol than Jakobi's protocol.

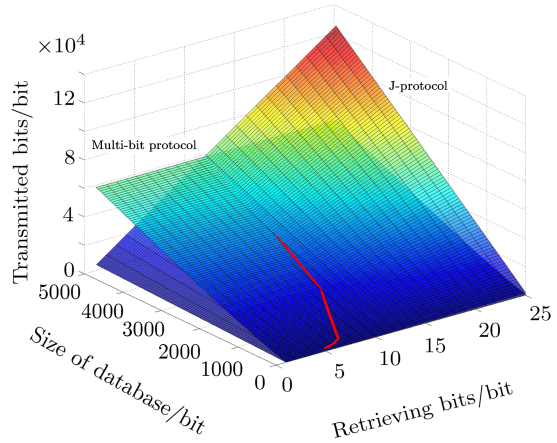

Fig. 1 The performance of the multi-bit protocol and J-protocol in multiple bits queries.

Table 1 Typical Ns and its smallest tolerable number of retrieving bits in multi-bit protocol.

N	10	20	30	60	110	200	400	800	1500
$n_{\min}$	3	4	5	6	7	8	9	10	11

Moreover, our protocol has better computation efficiency than Jakobi's protocol. Generally, in our protocol, the permutation takes N movements, and bitwise adding taking another N addition operations. In Jakobi's protocol, there would be n shift operations and nN bitwise addition operations. As long as more than one bit are being retrieved, there would be fewer operations being taken in our protocol than in Jakobi's. Note that when data is retrieving in blocks, QPQB has the highest computation efficiency comparing to the above two schemes, with N/n shift operations and N addition operations, but it scores less on the aspect of practicality and implementability, as will be discussed at the end of Subsect. 4.2.

4 Application to Block Queries

We will now consider Scenario 2. Suppose there are N/n items in Bob's database, with each item being a n-bit block. Attention must be paid when the proposed method is applied to block queries.

4.1 *Description*

In some cases, the bits that constitute a block weigh different, which need special measures to prevent possible misbehaviour. Imagine a special case where the evaluation of each company is stored in binary format as an item (block), thus bits at higher positions would outweigh those at lower positions. When constructing the permutation matrix, Alice moves her conclusive bits to the high positions of more than one block. In this way, she is able to read the high positions of more than one blocks illegally, thus would know the general evaluation of several companies in one query. Obviously, this infringes database security.

Gertner has mentioned a scheme for block retrieval in Ref. [2], where Alice sends her question only once as in single-bit SPIR schemes and Bob sends back n times, with one bit of the query result each time. Unfortunately, this proposal cannot be adopted here because this would expose the entire database to Alice if the oblivious key is simply reused. Some particular measures based on the structure of the data stored in the block can mitigate the threat from Alice's trick. For example, in the above special case where the highest bit is most crucial to the query result, Bob can replace the highest bit with the parity of the block. In this way, only Alice uses her conclusive bits to read the entire block can she make no absurd conclusion about the result.

Here we pursue a more universal preprocessing on Bob's database items. It must meet that (i) it creates no difficulty for honest Alice to recover the whole data block, and (ii) it should amplify Alice's ignorance of the transmitted message when Alice does not know the message entirely. The circumstance that Alice tries to deduce as much as possible information of the original block from partial knowledge of the block is analogous to the eavesdropping problem in the QKD protocol realization (e.g. BB84 QKD protocol),[18] where Eve wiretaps the correspondence between Alice and Bob, and tries to deduce part of the final key from some known sifted key bits. To solve this problem, we adopt binary matrix family as the hash function, similar to that used in Privacy Amplification[19] of a QKD protocol. In our block queries scheme, Bob is supposed to choose an $n \times l$ binary matrix M ($l \geq n$, because when $l < n$, X_i cannot be recovered by even honest Alice, which does not meet (i)) and make Alice's recovering a process of privacy amplification defined by M. Our strategy is to perform a dilution on the original block data. Let X_i be the column vector composed of the i-th block's bits, Bob constructs X_i^* that satisfies $X_i = MX_i^*$. This task is as simple as randomly picking a solution from the multiple solutions of a set of linear equations. Bob repeats this step for each block using the same M. Then the new database composed by X_i^*s is bitwise adding with the permutated key K_P^f (now the number of the conclusive bits in K^f should be equal to l). At the reception, Alice recovers X_i. Next, we will show how well a binary matrix can guarantee database security.

4.2 *Security*

Here we highly recommend the Toeplitz matrix family as the binary matrix, because of its lower communication complexity, good implementability, and no compromise in security.[20] An $n \times l$ Toeplitz matrix needs only $n + l - 1$

bits to define, much fewer than an arbitrary equal-size binary matrix does (nl bits). Here we take

$$T_1=\begin{pmatrix}1&0&1\\1&1&0\\1&1&1\end{pmatrix}_{3\times3},\quad T_2=\begin{pmatrix}0&1&0&1&1&0\\0&0&1&0&1&1\\1&0&0&1&0&1\end{pmatrix}_{3\times6},$$

$$T_3=\begin{pmatrix}1&0&0&1&0&1&1&1\\1&1&0&0&1&0&1&1\\1&1&1&0&0&1&0&1\end{pmatrix}_{3\times8},$$

as an example to show how well Toeplitz matrices meet requirement (ii) when the size of the block is 3 bits. Tables 2, 3, 4 have shown the possibility that Alice can recover the original block correctly ($\mathrm{Prob_{AliceWin}}$) at the absence of l_{miss} bits of the transmitted block under T_1, T_2, and T_3, respectively. $\mathrm{Prob_{AliceWin}}$ is calculated as

$$\mathrm{Prob_{AliceWin}}(l_{\mathrm{miss}})=\left(\sum_{j=1}^{C_l^{l_{\mathrm{miss}}}}1/A_j\right)\Big/\,C_l^{l_{\mathrm{miss}}},$$

where A_j is the number of eligible X_is under the restriction of the definitive $l-l_{\mathrm{miss}}$ bits of the transmitted block†. When $l=n=3$, we see that $\mathrm{Prob_{AliceWin}}=2^{l_{\mathrm{miss}}}$. When $l=6=2n$, if Alice gets only 3 bits of the transmitted block, possibility she can guess correctly the original data block is 0.15, only slightly higher than that when she guesses without any knowledge about this block (0.125). When $l=8$, she has an even smaller chance of 0.1304 to guess correctly if Alice gets half of the bits (4 bits). The advantage the partial bits bring to her is negligible for her deduction of the block. Note that if she gets less than a half of the transmitted block, she would get no information about X_i. This prevents her from splitting her conclusive bits to read several block, if l is set larger than $2n$.

Table 2 $\mathrm{Prob_{AliceWin}}$ and corresponding l_{miss} under T_1.

l_{miss}	0	1	2	3
$\mathrm{Prob_{AliceWin}}$	1.000	0.500	0.250	0.125

Table 3 $\mathrm{Prob_{AliceWin}}$ and corresponding l_{miss} under T_2.

l_{miss}	0	1	2	3	4	5	6
$\mathrm{Prob_{AliceWin}}$	1.000	0.500	0.250	0.150	0.125	0.125	0.125

The characteristics (size and 01 distribution) of the Toeplitz matrix are essential to the security level that it can provide. We find that there is a limit of how well the Toeplitz matrix of a particular size can provide dilution, and among Toeplitz matrices that have the same n, l, there are always more than one matrices that can reach this limit. T_1, T_2 and T_3 are randomly picked from their equally outstanding counterparts. One can search for the best-performance Toeplitz matrices of particular size in the same way. So far, we have not found a shortcut for constructing such Toeplitz matrix. Despite that, it can be applied to blocks of any structure (not constrained to the data structure mentioned at the beginning of Subsec. 4.1) with equally good performance once the most optimal one is found.

Table 4 $\mathrm{Prob_{AliceWin}}$ and corresponding l_{miss} under T_3.

l_{miss}	0	1	2	3	4	5	6	7	8
$\mathrm{Prob_{AliceWin}}$	1.000	0.500	0.259	0.161	0.130	0.125	0.125	0.125	0.125

Another benefit the dilution operation brings is that it can well eliminate the threat from surplus conclusive bits mentioned in Subsec. 3.1(i) in the case of block queries. As the example of T_1, T_2 and T_3 shows, allocating only a little more than $l/2$ conclusive bits to one block brings no significant advantage to her guessing, and information about this block is totally lost if less than $l/2$ conclusive bits is allocated to it. As long as Alice gets no more than $3l/2$ bits conclusive bits, it is hard for her to obtain significant information of several blocks at one time.

4.3 *Implementability*

We now compare our block queries sheme with QPQB. QPQB is more efficient than our scheme because Bob only has to send the database of size N, while in our protocol, Bob has to send Nl/n bits as the encrypted database. Our protocol also costs more on oblivious key distribution and item preprocessing. However, this should present no problem in classical communication. Also, QPQB forces Alice to deal with the key in blocks by basing the production of the conclusive key on high-dimension oblivious key distribution, which makes the protocol a high-dimension version of Jakobi's protocol. However, QPQB's realization is quite limited. First, once the setup has been built, the size of the data block is fixed. This makes QPQB scheme hard to generalize, because for different block sizes of different database, the users have to own corresponding setups to query, which is almost impossible. In our protocol, the size of data block that can be retrieved in one query depends on the adjustable number of conclusive bits in K^f. Second, the stage of high-dimension QKD systems largely limits the block size of QPQB. So far, the high-

†No worry needs to be taken that some of these eligible X_is have higher possibilities, because all the eligible X_is always have the same possibilities.

est dimension that can be reached is 16,[12] which just provides 4-bit block queries. In this sense, our protocol shows better scalability and flexibility.

Breakthroughs and improvements in 2-dimension QKD protocols and systems provide a solid platform for our protocol. 2-dimension QKD systems that reach high secure bit rate and show considerable robustness at long distance have been built.[21−25] Equipment that serves 2-dimension QKD has been commercially available (such as products from ID Quantique and MagiQ Technologies), and QKD networks have been built (such as DARPA and SECOQC), while high-dimension QKD maintains in the stage of experiment. It is more practical for the parties to base their block quantum private queries on 2-dimension QKD systems.

5 Outlook and Conclusions

In this paper, we propose a multi-bit quantum private query protocol. Some changes have been made on the classical part of Jakobi's single-bit quantum private query protocol, mainly being that instead of sending the shift s to Bob, Alice is allowed to construct and send a permutation matrix to Bob. We show that Bob retains no information about Alice retrieval through this permutation operation, while there exists a problem in database security caused by inaccurate control of the number of conclusive bits. As the size of retrieving data grows larger, the number of conclusive bits becomes less inaccurate. We leave this as an open question, seeking a method that allows more accurate control of the number of conclusive bits, and leaves Alice less chance to obtain extra information. Despite the above defect, our protocol has presented excellent efficiency in terms of communication complexity (exchanging only $O[\log_2(N!)]$ bits) and our block queries scheme presents better implementability (relies on the developed QKD systems).

References

[1] B. Chor, O. Goldreich, E. Kushilevitz, and M. Sudan, J. ACM **45** (1998) 965.

[2] Y. Gentner, Y. Ishai, E. Kushilevitz, and T. Malkin, J. Comput. Syst. Sci., **60** (2000) 592.

[3] V. Giovannetti, S. Lloyd, and L. Maccone, Phys. Rev. Lett. **100** (2008) 230502.

[4] V. Giovannetti, S. Lloyd, and L. Maccone, IEEE T. Inform. Theory, **56** (2010) 3465.

[5] G.L. Long, Commun. Theor. Phys. **45** (2006) 825.

[6] B. Shang, Commun. Theor. Phys. **48** (2007) 264.

[7] M. Jakobi, C. Simon, N. Gisin, J.D. Bancal, C. Branciard, N. Walenta, and H. Zbinden, Phys. Rev. A **83** (2011) 022301.

[8] F. Gao, B. Liu, Q.Y. Wen, and H. Chen, Opt. Express **20(16)** (2012) 17411.

[9] P. Chan, I. Lucio-Martinez, X.F. Mo, C. Simon, and W. Tittel, Sci. Rep. **4** (2014) 5233.

[10] F. Gao, B. Liu, W. Huang, and Q.Y. Wen, IEEE Journal of Selected Topics in Quantum Electronics **21(3)** (2014) 98.

[11] C.Y. Wei, F. Gao, Q.Y. Wen, and T.Y. Wang, Sci. Rep. **4** (2014) 7537.

[12] S. Etcheverry, G. Canas, E.S. Gomez, W.A.T. Nogueira, C. Saavedra, G.B. Xavier, and G. Lima, Sci. Rep. **3** (2013) 2316.

[13] V. Scarani, A. Acín, G. Ribordy, and N. Gisin, Phys. Rev. Lett. **92** (2004) 057901.

[14] U. Herzog and J.A. Bergou, Phys. Rev. A **71** (2005) 050301.

[15] P. Raynal, arXiv:quant-ph/0611133 (2006).

[16] C.W. Helstrom, Journal of Statistical Physics **1** (1969) 231.

[17] C.A. Fuchs, arXiv:quant-ph/9601020 (1996).

[18] C.H. Bennett, and G. Brassard, *Proc. IEEE Int. Conf. on Computers, Systems and Signal Processing*, IEEE, New York (1984) 175.

[19] C.H. Bennett, G. Brassard, and J.M. Robert, SIAM journal on Computing **17** (1988) 210.

[20] H. Krawczyk, *Advances in Cryptology CRYPTO '94*, ed. Y. Desmedt, Springer, Berlin, Heidelberg (1994) p. 129.

[21] A.R. Dixon, Z.L. Yuan, J.F. Dynes, A.W. Sharpe, and A.J. Shields, Appl. Phys. Lett. **96(16)** (2010) 161102.

[22] S. Wang, W. Chen, J.F. Guo, Z.Q. Yin, H.W. Li, Z. Zhou, G.C. Guo, and Z.F. Han, Opt. Lett. **37** (2012) 1008.

[23] A. Tanaka, M. Fujiwara, K. Yoshino, S. Takahashi, Y. Nambu, Akihisa Tomita, S. Miki, T. Yamashita, Zhen Wang, M. Sasaki, and A. Tajima, Quantum Electronics, IEEE Journal of **48** (2012) 542.

[24] K.A. Patel, J. F. Dynes, M. Lucamarini, I. Choi, A.W. Sharpe, Z.L. Yuan, R.V. Penty, and A.J. Shields, Appl. Phys. Lett. **104** (2014) 051123.

[25] A.R. Dixon, J.F. Dynes, M. Lucamarini, B. Fröhlich, A.W. Sharpe, A. Plews, S. Tam, Z.L. Yuan, Y. Tanizawa, H. Sato, S. Kawamura, M. Fujiwara, M. Sasaki, and A.J. Shields, Opt. Express **23** (2015) 7583.