BB84 Quantum Key Distribution Protocol

Pritheesh Panchmahalkar
Bobby B. Lyle School of Engineering
Southern Methodist University
Dallas, TX
ppanchmahalkar@smu.edu

***Abstract*—In this paper, we discuss one of the first Quantum Key Distribution Protocols, BB84 protocol proposed by Bennett and Brassard in 1984. Quantum Key Distribution is the mechanism of sharing a secret key among two parties that wish to encrypt their data using the shared secret key and encryption algorithms of their choice. BB84 uses both quantum channel and a public channel for communication of messages to agree upon a common key. We also discuss how an eavesdropper can be detected based on the properties of quantum mechanics and no cloning theorem and also focus on the various approaches for the last phase of the BB84 protocol called as the reconciliation phase which reduces the Quantum Bit Error Rate (QBER).**

***Keywords*—*BB84, Quantum Key Distribution, QKD protocol, Quantum Bit Error Rate (QBER), Reconciliation phase, Qubit*

I. INTRODUCTION

With quantum technology becoming a reality, we need techniques that are secure against the quantum attacks. The cryptographical methods that are resistant to the quantum attacks are referred to as post-quantum or quantum-safe cryptography. BB84 [1] is a quantum key distribution protocol which was coined by Charles Bennett and Gilles Brassard in the year 1984 and hence the name BB84. Quantum key distribution is a mechanism of sharing a private key among two parties. For a cryptographic symmetric encryption algorithm to be used, the end users need to use a shared key. To share the key, techniques like Diffie-Hellman key exchange use mathematical computations. Quantum key distribution uses the quantum physics to ensure security of the key shared among the two parties. It generates a random key for encryption and decryption algorithms. QKD uses two channels for communication; an authenticated classical channel and a quantum channel. Qubits are transmitted in the quantum channel as photons. The Qubits are quantum bits, which result by encoding the bits (0s or 1s) using one of the two orientations. The security of quantum key distribution lies in the Heisenberg uncertainty principle, which states that measuring a qubit interferes with that system, and thus information about the state cannot be collected unless it is measured (Brassard, A Bibliography of Quantum Cryptography, 1993) and also that the quantum information cannot be perfectly cloned.

We will use the classical cryptographical notation to represent the two parties Alice and Bob who want to communicate with each other privately, and an eavesdropper Eve tries to intercept their communication and read their messages. For a secure communication, Alice and Bob need a shared key to encrypt their communication using a symmetric key algorithm, say, AES. Quantum properties allow Alice and Bob to communicate in public channels even if Eve tries to intercept their communication.

II. BB84 QUANTUM KEY DISTRIBUTION PROTOCOL

Without the knowledge of the key, it is difficult to break the encryption algorithm i.e., the data is safe as long as the key is kept safe and private. Quantum key distribution provides a mechanism to distribute the key among the two parties. It takes advantage of the quantum mechanics to ensure that the key being shared is not intercepted. Even if Eve tries to intercept the data, the scheme detects that the breach and discards the key and generates a new key.

In a Quantum key distribution mechanism, quantum channel and an authenticated classical channel are used for communication between Alice and Bob to generate a new private key. Both the channels are public and are subject to eavesdropping. The quantum channel is not used for any meaningful communication but is used to send random bits between users who do not share any kind of secret information initially. Based on the sequence of messages exchanged in the quantum channel and the ordinary classical channel, the end users come to an agreement for a shared private key. If an eavesdropper happens to measure the photons being exchanged in the quantum channel, then the original information that is transmitted by a user is disturbed and cannot transmit the perfectly cloned qubits based on No cloning theorem [2]. If a quantum state is read or measured, then the quantum state is disturbed hence modifying the actual information being sent. There is a high probability to detect the eavesdropping in the quantum transmission. If eavesdropping is detected, then the current process of generating a key is stopped and the whole process is started again to generate a new key, or they could choose to defer the process. After they agree upon a shared private key, they use it as a one-time pad to encrypt their communication or any other required cryptographic purposes which require secret communication.

Alice first generates a random bit string consisting of bits 0s and 1s. The generation of the random bit string should have the following properties.

a) It should be a random process i.e., there shouldn't exist a pattern to guess the next bits after observing a few bits.
b) Should have roughly equal length of 0s and 1s.
c) Should not have 0s and 1s appearing for long length successively.

She generates another random sequence of bases, rectilinear (+) or diagonal (x) which is the same length as the random bit string. The random generation of bases should also have the same properties as for the generation of random 0s and 1s. She encodes the generated random bit string using the random sequence of bases which generates photons in one of the four possible states. A basis is an orientation. It could be one of two:

i. Diagonal Basis and,
ii. Rectilinear Basis.

These photons represent one bit of string based on the generated basis. For a rectilinear basis, one is represented by 90-degree photon and zero is represented by a 0-degree photon. For a diagonal basis, one is represented by 135-degree photon and zero is represented by a 45-degree photon.

TABLE I. BASIS ENCODING

Basis	**Bits**	
	0	1
Rectilinear (+)	→	↑
Diagonal (x)	↗	↖

Table I shows the encoding procedure for the bits based on the basis. Alice transmits these encoded photons to Bob over a quantum channel one by one and records the bit, basis and timestamp before transmission. At this point in time, Alice only shares the photons, but the generated bit sequence is private and only known to Alice. Alice may record the timestamp to detect any unusual delay in transmission which could be due to eavesdropping by Eve. At this point in time, Alice has shared only the polarized state i.e., the encoded bits with Bob.

Bob receives the photons from Alice representing 1s or 0s based on their state represented by the arrows as shown in table 1. Bob also records the timestamps at which he receives the photons. It is also possible that there might be loss in information during the transmission due to imperfect detectors at the Bob's end or a few photons could be lost in the channel during their transit. He generates new sequence of bases to measure each of the photons received from Alice. This sequence is random and independent of what the message received was. He then measures each of the received photons using the random sequence of bases. If the received photon is in rectilinear representation and the randomly generated basis is also rectilinear then the photon is measured in rectilinear as is. If the basis is diagonal, then either 45-degree or 135-degree photon is generated randomly with equal probability. But the received photon is in diagonal representation say 45-degree or 135-degree and the randomly generated basis happens to be a rectilinear basis then the photon is measured as a 0-degree or a 90-degree photon with an equal probability.

TABLE II. MEASURING PHOTON BASED ON RANDOMLY GENERATED BASIS AT BOB'S END

Bob's basis	Received Alice's transmitted photon			
	→	↑	↗	↖
Rectilinear (+)	→	↑	→ or ↑	→ or ↑
Diagonal (x)	↗ or ↖	↗ or ↖	↗	↖

Table II illustrates how Bob measures the received photons from Alice through the quantum channel. When Bob chooses basis that is not the same as the one chosen by Alice, and Bob measures the photon the actual information sent by Alice is lost. The received photon is in diagonal representation and the randomly chosen basis is also diagonal basis then the photon is measured with the same value.

The values measured correspond to a binary 0 or 1. If the photon is either a 0-degree photon or a 45-degree photon, it corresponds to a binary zero, and if the photon is either a 90-degree photon or a 135-degree photon, it corresponds to a binary one which is similar to the way Alice encodes the binary bits using her random bases.

TABLE III. IDENTIFYING BITS

Orientation	Bit Value
→	0
↑	1
↗	0
↖	1

Table III illustrates how bob identifies bit information using the output obtained by measuring the Alice’s photons with his bases. Bob and Alice both have a sequence of bits they can compare and come to an agreement for a private key if no eavesdropping is detected. After measuring all the photons received Bob communicates with Alice now sends the bases, she chose to encode the bits over a public channel to Bob for the first time. She is not concerned about the privacy of bases as the communication of the key has already taken place.

Alice may analyze the timestamps received from Bob against the recorded timestamps when she sent the photons to check for unusual delay in the transmission. If the communication in the quantum channel is not disturbed, Alice and Bob should agree upon the bits encoded the photons. Bob checks the bases Alice has sent against the bases he randomly chose earlier to measure Alice’s photons. He discards the information that didn’t match the bases, Alice has sent and identifies the bits represented by the left-over information and shares it with Alice. A one-time pad is generated and can be used by Alice and Bob to encrypt their communication using one the various available encryption algorithms.

It is possible that the bits that Bob identifies are the same as generated by Alice when he chooses a different basis to measure the photon as the process is random but at the same time it is also possible that it is incorrect. Roughly on an average, more than half the bits produced after measurement of photons match the one’s generated at Alice’s end. In the presence of Eve, if she measures the photons that Alice sends, then when there is a difference in the basis with which she measures the photon and the basis Alice chooses to encode her bit, she generates a photon which is in different orientation and retransmits the measured photons to Bob. Assume that before Alice has shared her randomly generated bases with Bob, Eve intercepts the communication and retransmits her measured photons to Bob. Any measurement resulting in b bits of expected information must induce a disagreement with probability at least b.2 if the measured photon, or an attempted forgery of it, is later re-measured in its original basis. [refer 1984 paper]. Suppose that Eve measures all the photons in diagonal bases and retransmits the measured photons to Bob. In the process of measurement, Eve could be right for about half of the times. Consider the times where Eve goes wrong and if re-measured with the original bases, the retransmitted bits induce disagreements in one-fourth of those. Hence, when Eve attempts to eavesdrop and tries to measure the photons sent by Alice, she introduces more errors in the process as the information is lost whenever she measures incorrectly and can be easily detected.

Hence, to detect eavesdropping Alice and Bob can adopt the following procedure. Bob shares some of the bits with Alice over the public channel. The bits chosen by Bob are random and the size to be chosen could be around one-third to one-fourth of the original size of the key agreed upon. If all the comparisons are right, then it is safe to assume that the communication was not intercepted and hence, they agree upon a key. If eavesdropping is detected, they simply discard the present key and start the whole process again and create a new key. In the process of eavesdropping, it is also possible that Eve could be clever and be disturbing only the subset of information and hence remaining undetected in the process. She could obtain part of information of the key and the whole process appears to be a disturbance in the quantum channel. We now summarize the process of BB84.

- Alice randomly chooses bits (K) and bases (B) to encode the bits which result in qubits which she transmits as photons over a public quantum channel to Bob. The bits Alice chooses are private to her.
- Independent of what Alice has sent, Bob randomly chooses bases (B’) by which he measures the received photons. In the process of measuring a photon, he either chooses the same basis that Alice has used to encode the bit and gets a correlated result or chooses the exact opposite basis and obtains an uncorrelated result. It is also possible that some information is lost during the process of transmission.
- At this time, Bob has bits (K’) resulting from the measurement of received photons and communicates with Alice over public classical Channel
- Alice shares her bases (B) over public channel with Bob.
- Bob compares the received bases (B) and his bases (B’) to check which were the same. About 50% could be incorrect due to randomness and they are discarded. This is called key sifting.

- If they detect eavesdropping due to higher percentage of their bits being incorrect than expected, they discard the key and start the process again.
- Alice and Bob begin reconciliation phase by choose random bits in the rest of the key to correct any possible errors. Errors could occur during transmission or Eve may modify the data during interception. They correct the errors and discard the bits used for error correction as the bits are shared in the public channel. The remaining key is the shared secret key.

TABLE IV. BB84 PROTOCOL EXAMPLE

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Alice bits	0	1	0	1	1	1	1	1	0	1	0	0	0	1	0	1
Alice bases	D	D	D	D	R	D	R	R	R	R	R	D	D	D	R	R
Alice photons	↗	↖	↗	↖	↑	↖	↑	↑	→	↑	→	↗	↗	↖	→	↑
Bob bases	R	R	D	D	R	D	D	R	D	R	R	R	D	D	R	R
Bob photons	↑	→	↗	↖	↑	↖	↖	↑	↗	↑	→	→	↗	↖	→	↑
Bob bits	0	1	0	1	1	1	1	1	0	1	0	0	0	1	0	1
Match			✓	✓	✓	✓		✓		✓	✓		✓	✓	✓	✓
Pre-final Key			0	1	1	1		1		1	0		0	1	0	1

III. RECONCILIATION PHASE

We will now discuss the reconciliation phase of the Quantum key distribution protocol. This phase is essential for the protocol as there may be disturbance during the transmission due to interception or fault in the equipment set up and hence, the information read at Bob's end must be corrected. There are many protocols designed for reconciliation of the key in the quantum key distribution scheme. Most of these protocols use Hamming code or similar error correction schemes that use parity check. This phase could contain steps like error estimation, error reconciliation and privacy amplification based on the protocol used for reconciliation. Error estimation determines the probability of occurrence of error in the key after qubits transmission and key sifting. This is called as Quantum Bit Error Rate (QBER). Error reconciliation is the process of correcting the errors without making any sense of the communication going on between Alice and Bob to Eve. We will discuss Cascade Protocol, Winnow Protocol and PDG in this paper.

A. Cascade Protocol

The Cascade reconciliation protocol [3] is essentially an improvement to earlier protocol known as BBBSS. In the first pass of the protocol, the key is divided into blocks of length k, which is calculated based on the error estimated in the channel. Alice and Bob compute the parity bits for each block and exchange them. If the parity bits match, it means there is no error in the block. If it doesn't match, a binary search is performed, and single bit errors are found and corrected. Now a permutation is applied, and the block size is increased to 2.k, and another pass is performed same as the first. If any errors are corrected in the second pass or later, it means that an error was present in the same block in the previous passes as the error was neither detected nor corrected in the previous passes. If any error has been corrected in the second pass or later passes, a binary search is applied again on the block in which error was corrected. If a new error is identified, it is possible that this error could have masked other potential errors in the block and hence, the binary search is cascaded across the previous passes to check for these masked errors.

On an Average two errors are corrected for every mismatch in the parity bits after the first pass and hence, the amount of errors in the key are decreased exponentially in each pass. Ideally, four passes are generally sufficient to correct all the errors. One of the concerns of the Cascade protocol is that a lot of information is exchanged between Alice and Bob.

B. Winnow Protocol

Unlike the Cascade Protocol that uses the Binary to correct the errors, the Winnow Protocol [4] uses Hamming code for the correction of errors. Hamming code was invented by Richard Hamming, in 1950 when he was working at the Bell Labs and was frustrated by the errors which could be detected but couldn't be corrected until he found a powerful algorithm for this purpose. Hamming codes are a set of linear error-correcting codes. They can detect up to two-bit errors and correct one-bit errors. Hamming code (n, k) use k-bit data and (n-k) bits are reserved for parity check. For example, Hamming code (7, 4) uses 4-bit data and 3 bits are used for parity check. It makes use of Hamming matrices, a Generator matrix, G and a parity check matrix H. For a generator matrix, $G_{m \times n}$ the first m columns form the identity matrix and rest of the columns are filled based on the parity equations matrix, A. The Generator matrix of a linear code C of block length n is an m x n matrix, G for some m, whose row space is C. The matrix G is given by,

$$G = (I_m \, A) \tag{1}$$

where I_m is a m x m Identity matrix

$$I_m = \begin{pmatrix} 1 & 0 & \dots & 0 & 0 \\ 0 & 1 & \dots & 0 & 0 \\ 0 & 0 & \ddots & 0 & 0 \\ 0 & 0 & \dots & 1 & 0 \\ 0 & 0 & \dots & 0 & 1 \end{pmatrix} \tag{2}$$

And A is any arbitrary m x (n-m) matrix. The code obtained using G has a dimension m, as it is row reduced and they are linearly independent.

The parity check matrix, H is obtained using the following equation,

$$H_{n-m\,x\,n} = (-A^T I_{n-m}) \tag{3}$$

In this protocol, Alice and Bob divide their key strings into blocks of length, say l. Alice and Bob compare parities for each block and check if they match same as in Cascade. Alice now computes the dot product of generator matrix and message called as code word and transmits it to Bob. The first m rows of the generator matrix is an identity matrix which implies that the first m bits in the code word forms the message. Bob receives the code word from Alice and now computes the dot product of the code word and the parity check matrix and obtains a syndrome. If the syndrome is a zero, it means there were no errors during the transmission. Otherwise, it means that there was at least one error and this syndrome could be used to correct the error.

$$M \,.\, G = C \tag{4}$$

$$\begin{bmatrix} 1 & 1 & 0 & 0 \end{bmatrix} . \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 0 & 0 & 1 & 1 & 0 \end{bmatrix}$$

Figure 1 shows an example of a (7, 4) Hamming code calculation done by Alice. Alice has a message M with size 1 x 4 and a Generator Matrix (G) with size 4 x 7. She computes the dot product of the matrices, $M \,.\, G$ to calculate the parity bits and obtain the code word of size 1 x 7. Looking at the code word, C it is clear that the first four bits are message M and hence the remaining bits constitute the parity bits.

$$H \,.\, C = S \tag{5}$$

$$\begin{bmatrix} 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{bmatrix} . \begin{bmatrix} 1 & 1 & 0 & 0 & 1 & 1 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}$$

Figure 2 shows how Bob uses the code word sent by Alice to compute the Syndrome, S using the parity check matrix H. Bob has a code word C with size 1 x 7, a parity check matrix 3 x 7. He then computes the dot product of H and C, $H \,.\, C$ to obtain the syndrome (S) of received code word (C).

The detected error is corrected in such a way that the syndrome equals zero for the least amount of changes that could be made to the received code word. But in the QKD, the key needs to be shared in a secure manner, so as to make the communication secure from Eve. It is recommended that Alice compute her syndrome using her string and send it to Bob. If the received syndrome is equal to what Bob has calculated then, it means there are no errors. If Bob's syndrome doesn't equal to the syndrome received from Alice, then he can use his syndrome to identify where the error was. The parity check matrix H results in a syndrome that consists of parity check on the message encoded by the matrix operations.

The hamming code solves most of the errors in the transmission, but it is also possible that it might introduce new errors in the process if the error count in the block happens to be high. For this reason, a permutation is performed so as to evenly distribute the errors among the block, keeping the number of errors below a threshold that hamming code could detect per block.

C. PDG

Prestridge-Dunham Guardian (PDG) codes [5] were introduced by Shawn Prestridge and Dr. Dunham to improve the existing problems of the winnow protocol [4]. It is very similar to the Winnow protocol and maximizes the error detection and correction compared to the Winnow. PDG scheme exploits the Hamming code syndrome to achieve higher efficiency. Similar to the Winnow Protocol, in PDG use the keys k and k' with Alice and Bob respectively to correct the errors. Similar to most of the schemes, the QBER (Quantum Bit Error Rate) of the channel is calculated and known before the reconciliation begins. This QBER helps us calculate the block length in which the keys should be divided into similar to the Winnow Protocol. PDG uses a sequence of steps each round to eliminate the errors. The number of rounds to be used also depend on the QBER of the quantum channel.

The sequence of steps involved in PDG are as follows. The first step is two choose a block length in which the key strings are to be divided. This block length is calculated based on the probability of an undetected error in the hamming code. It is recommended by the authors that the block length should be such that the probability of undetected error is below 1%. The probability of undetected error is calculated as [3]

$$p_{UDE} = \sum_{i=2}^{k} \binom{k}{i} p_{QBER}^{i} (1 - p_{QBER})^{k-i} \qquad (6)$$

where p_{UDE} is the probability of an undetected error. Also, the length of block should be compatible with the Hamming code being used say, (7, 4) Hamming code, (15, 11) Hamming code, (31, 26), (63, 57) and so on.

In the second step, after choosing a block length, Alice and Bob have to fill these blocks with the bits in the key string and pad 0s in the last block if necessary, to fill the block. They randomly permute through the keys and fill the blocks such that each bit goes into only block. The idea is similar to winnow protocol, which is to evenly distribute the errors among the block, to avoid the errors in the successive bits, which would in turn help in burning less bits in the process.

In the third step, Alice and Bob run a parity check scheme called as Longitudinal Redundancy Check (LRC) on each of the blocks. Alice computes LRC for each of the blocks and shares the parity with Bob. Bob also computes LRC at his end compares it with the received LRC for that block to check for any odd number of bit errors. The LRC scheme can detect odd number of bit errors. The possibility of two or more-bit errors occurring in a block is not likely as the block length is calculated based on the QBER of that quantum channel. However, in such cases, the errors can be eliminated by running the scheme for more rounds.

The fourth step involves running a Hamming Code if the parity check detects an error at Bob's end. If Bob detects an error using the LRC check, he asks Alice for her syndrome for that block. Alice computes the parity bits using the Hamming Code (n, k) for that block, obtains a code word which is the concatenated result of bits in the block and the parity bits. Now, she computes the dot product of the code word and the parity check matrix, H to obtain a syndrome, S.

$$c\,.H = S \qquad (7)$$

Alice sends the calculated syndrome, S to Bob. Bob now calculates the syndrome at his end using the parity check matrix, H and the code word received from Alice and compares the result with the syndrome received from Alice. If one-bit error is detected, Bob corrects the error in the block using the syndrome. If two-bit error is detected then, Bob uses the precomputed syndrome tables to check the bit positions where the errors could reside and selects one among the pair and discards one of the bits removing at least one of the errors in the block. This process of discarding the bits not only eliminates the errors but also reduces the number of key bits discarded in the later bits due to exchange of parity bits caused by the error detection.

The steps are repeated for all the blocks and Bob communicates with Alice at the end of the process to let her know what positions of the bits of the key he has discarded. Even if the communication is public and Eve can intercept it, the key is secure as they re-randomize the key again using the permutation process only, they know and have agreed upon. The above process is repeated for a certain number of rounds based on the quantum bit error rate of the quantum channel used for exchange of quantum information. The number of rounds and the Hamming code scheme used based on quantum bit error rate have been defined as author as follows.

TABLE V. PDG SELECTION PARAMETERS BASED ON QBER

QBER	Pass 1	Pass 2	Pass 3	Pass 4
1%	(15, 11)	(31, 26)	(63, 57)	(127, 120)
2%	(15, 11)	(31, 26)	(63, 57)	(127, 120)
3%	(15, 11)	(31, 26)	(63, 57)	(127, 120)
4%	(15, 11)	(31, 26)	(63, 57)	(127, 120)
5%	(15, 11)	(15, 11)	(64, 57)	(127, 120)
6%	(15, 11)	(31, 26)	(31, 26)	(31, 26)
7%	(15, 11)	(15, 11)	(31, 26)	(31, 26)
8%	(15, 11)	(15, 11)	(15, 11)	(31, 26)
9%	(15, 11)	(15, 11)	(15, 11)	(31, 26)
10%	(15, 11)	(15, 11)	(15, 11)	(15, 11)

We can see that the authors recommend using the same Hamming Code (15, 11) for first pass for all the values ranging from 1% to 10% quantum bit error rates. This could make the Hamming Codes to be able to detect and correct most of the single bit errors easily in the first pass and avoid “bursty” errors in the block when the block size increases. The block size is increased up to 127 bits for up to 5% quantum bit error rate. Higher the quantum bit error rate, the lesser the block length should be to cover all the errors and hence the block length remains less for all the passes for quantum channels having 10% quantum bit error rate. This is because of the fact that more errors would occur in the same block and Hamming Code can detect only up to two-bit errors and correct one-bit errors, which would mask the errors if there are more than two in the same block and could remain undetected. The remaining key string is used as the secret key for the encryption algorithm Alice and Bob agree upon.

IV. CONCLUSION

In this paper we have discussed about BB84 quantum key distribution protocol and how it works. We have discussed in detail how all the phases in the protocol work and various protocols for the reconciliation phase. QKD is a pretty secure Key Distribution Protocol and it is implemented widely these days. The reconciliation phase of the protocol has a lot of scope for research on how to optimize the communication between the end parties and increase the length of the key they agree upon. As we have seen PDG is an efficient scheme which discards bits wherever necessary if two or more-bit errors occurs, thus reducing the overall communication for that error and also the number of bits that would be discarded because of the exchange of parity bits.

V. REFERENCES

[1] C. H. Bennett and G. Brassard, “Quantum cryptography: Public key distribution and coin tossing," in Proc. IEEE Int. Conf. Computers, Systems, and Signal Processing. Bangalore, India, December 10 -12,1984. pp. 175-179.

[2] W. K. Wootters and W. H. Zurek, “A single quantum cannot be cloned,” *Nature*, vol. 299, no. 5886, pp. 802–803, 1982.

[3] G. Brassard and L. Salvail, “Secret-key reconciliation by public discussion”, Advances in Cryptography - Eurocrypt '93, Lecture Notes in Computer Science, 410-423, 1993.

[4] W. T. Buttler, S. K. lamoreaux, J. R. Torgerson et al., “Fast, efficient error reconciliation for quantum cryptography”, Phys. Rev. A 67, 052303, 2003.

[5] S. Prestridge and J. Dunham, “Improving Throughput in BB84 Quantum Key Distribution,” Proceedings of the 14th International Joint Conference on e-Business and Telecommunications, 2017