Calculation of the Key Length for Quantum Key Distribution

Miralem Mehic[1], Marcin Niemiec[2], Miroslav Voznak[1]
[1]VSB-Technical University of Ostrava,
17.listopadu 15, 708 00 Ostrava-Poruba, Czech Republic
[2]AGH University of Science and Technology,
al. Mickiewicza 30, 30-059 Krakow, Poland
miralem.mehic.st@vsb.cz

***Abstract*—Quantum key distribution (QKD) is based on the laws of quantum physics and therefore it can guarantee the highest level of security. It is used to establish the key that is used for further symmetrical encryption. Since QKD consists of several phases in which the key is reduced, it is necessary to define the equation by which the length of the raw key is calculated. In this paper, we analyse all QKD phases with an emphasis on the explanation of the process of shortening the initial key. The results are verified with a large number of tests using a quantum cryptography simulator.**

***Index Terms*—Cryptography, security, quantum mechanics, cryptographic protocols.**

I. INTRODUCTION

It seems likely that quantum cryptography (QK) or to be precise, quantum key distribution (QKD), can be used in every day's life to provide unconditionally secure distribution of secret key material. QKD uses quantum physical principles to establish symmetrical binary keys between legitimate users that will use these keys to encrypt theirs communication data. However, QKD link has several limitations: distance limit, point-to-point link behavior, limited key generation rate and dedicated optical infrastructure which significantly affect the final cost.

It is interesting to compare the key rates from QKD networks which were constructed in previous years. In 2002, QKD systems achieved key rate of 1 kbps [1] which was used in the first QKD network in the world, DARPA QKD network. In 2007 in SECOQC, this key rate has increased tenfold [2] while in 2011 in Tokyo QKD network the key rate of 300 kbps have been achieved [3]. Therefore, it is reasonable to expect higher key rates in the coming years. This key rate is very important because it dictates the type of encryption to be used in the network. If the key rate is not high enough, solutions like one-time pad (OTP), which has been proven to be the most secure solution [4], cannot be used due to lack of key material.

An OTP uses a keystream of fully random digits that are combined with plaintext one at a time to form the ciphertext. Consequently, OTP requires the same amount of key material as the amount of data to be encrypted.

QKD employs two distinct channels between communicating parties: quantum channel which is used for transmission of quantum key material encoded in some of photon's properties and public channel which is used for verification of exchanged key material. Key material obtained through the quantum channel is often called raw or initial key material, and this material is further processed over public channel which results with the final key material.

As initially proposed by DARPA [1] and later applied in SECOQC [5] and Tokyo, QKD links in QKD networks are organized on the following way: both endpoints of corresponding QKD link have a storage key reservoir and these reservoirs are gradually filled with the final key material. Later, this key material is used with IPsec protocol suite to encrypt one or more secure virtual private network (VPN) tunnels through the unsecure Internet. Obviously, the type of used encryption determines the speed of emptying reservoir, while the key rate of QKD link determines the charging rate of storage reservoir.

Due to small key rate, previously mentioned QKD networks were focused on IPsec solution with rapid changing of 128-bit or 256-bit Advanced Encryption Standard (AES) keys between the parties. Faster key exchange means higher level of security, so solutions where 12,500 keys per second have been exchanged were implemented [6]. The aim was to get close to lower Data Bits per Key Bit (DPK) limit which is based on *birthday bound* and resemblance to One-Time Pad (OTP) consumption.

In recent years there has been significant development in terms of increasing the key rate, so it is necessary to consider the cases in which key rates are sufficient for OTP encryption. In this paper we present an equation which can be used to calculate the total length of the raw key based on the length of the final key. For testing purposes we used the payload size of the most popular VoIP codecs, and we verified results with the QK simulator [7] with more than 500,000 experiments.

This paper is organized in the following way: in Section II, the BB84 protocol is explained. The results of the

Manuscript received December 27, 2014; accepted April 29, 2015.

The research leading to these results received funding from the grant of SGS reg. no. SP2015/82 and from the European Regional Development Fund in the IT4Innovations Centre of Excellence project reg. no. CZ.1.05/1.1.00/02.0070, both of them were conducted at VSB-Technical University of Ostrava. The research was partially supported by the AGH University of Science and Technology under contract no. 11.11.230.018.

calculation are shown in Section III, and the final conclusions are presented in Section IV.

II. QUANTUM KEY DISTRIBUTION PROTOCOLS

Quantum key distribution (QKD) protocols are used for providing a key to the participant of symmetric system transmission in a safe manner. There are several QKD protocols, such as Ekert's entanglement based E91 protocol [8] and SARG04 protocol by Scarani *et al*. [9] and others. They consist of nearly identical steps at a high level, but differ, among others, in the way the quantum particles or photons are prepared and transmitted. We choose the BB84 protocol, which is the first proposed QKD scheme and still the most widely used in practice. This protocol was presented in 1984 by Bennett and Brassard [10] and it consists of the following five successive stages: secret key exchange, extraction of the raw key, error rate estimation, key reconciliation, and privacy amplification.

A. Secret Key Exchange

At the beginning of communication, the sender hereinafter named Alice and the recipient hereinafter named Bob must agree on the same alphabet. BB84 assumes that the polarization of photons is used as a carrier of information, so Alice defines a random key with a length Q and uses a randomly selected polarization from the alphabet as a carrier of the key. For example, for the bit value 1 she can choose either vertical (x = 90°) or diagonal polarization (x = 45°), and for the bit value 0 she can choose either horizontal polarization (x = 0°) or opposite diagonal polarization (x = -45°). On the receiving side of the quantum channel, Bob chooses a randomly selected basis for detection. Since Bob does not know which basis Alice has used, and he uses a randomly selected basis, he will be able to reliably detect only 50 % of the sent key.

B. Extraction of a Raw Key - Sifting

After exchanging the key values via the quantum channel, all further communication is performed via the public channel. First, Bob informs Alice which polarization basis he has used for each received bit. Second, Alice responds when Bob uses the correct polarization basis and when he uses incompatible bases (these bits must be removed). It must be stressed that Bob only discloses information about the used basis, while the value of the measurement remains secret. After this step, Bob is certain of the sequence of the correct polarization he used for detection. However, we can conclude that the length Q of the raw key is significantly decreased. Bob reliably receives approximately 50 % of the raw key. In the rest of this paper, B represents the length of Bob's reliably received key.

C. Error Rate Estimation

The purpose of error estimation is to determine the percentage of errors in the key after quantum transmission and sifting have occurred. Errors may occur because of a disturbance of the quantum channel, noise in the detectors or an optical misalignment and other reasons. But errors may also occur due to eavesdropping by eavesdropper Eve. However, the threshold of bit error rate for optical channel without presence of Eve is known in forward so Alice and Bob must compare a small portion of their key in order to estimate the quantum bit error rate (QBER). If the error rate is higher than a given threshold, Alice and Bob revealed the presence of Eve and the key distribution process starts all over again.

If we mark n as the total length (Q) of the raw key then the number of bits k that will be used for the QBER estimation depends on the length of the sample block used for error rate estimation which is defined with a parameter we refer to as "level of security" $\hat{S}(k)$. In [11], two levels of security are defined: basic and advanced. Alice and Bob must select the desired level of security $\hat{S}(k)$, and use (1) to calculate k.

However, Alice and Bob must delete the part of the key which they used for estimation of the error rate. It means that the raw key will be shortened even more. We use notation R to mark the length of the key after this phase.

$$\hat{S}(k) = \frac{-\sum_{k=1}^{n} \frac{k}{n} \log \frac{k}{n}}{n}. \quad (1)$$

D. Key Reconciliation

When Alice and Bob are sure that the key distributed via the quantum channel has a low error rate, they must find and correct or delete all errors in the rest of the key. This phase is known to be highly interactive and time-consuming, since the discussion about the location of errors in the key is performed through the public channel. The cascade protocol [12] is the most widely used reconciliation protocol due to its simplicity and efficiency. It is run iteratively in the given number of iterations where random permutation of the key is performed with the objective to evenly disperse errors throughout the key. Next, the permuted key is divided into equal blocks of k_i bits, and after each iteration permutations are performed again and the block size is doubled: $k_i = 2k_{i-1}$. For each block, Alice and Bob will exchange the results of the parity test and perform a binary search to find and correct errors. Instead of going through all the iterations continuously, the Cascade protocol investigates errors in pairs of iterations. The process is recursive. This means that no bits are discarded during the first iteration. It also means that for any error corrected in the second iteration there must be at least one matching error contained in the same block in the previous iteration, since neither error was found or corrected in that iteration. For this reason, for each correction made in any iteration after the first one, a binary search is rerun on the block containing the bit corrected in all previous iterations, in order to identify any potential matching errors. For any new error detected, it follows that another error in a previous iteration was masked, thus the process is repeated so that the error detection and correction process cascades through all previous iterations. This process is illustrated in Figure 1, where the following notation is used: e_i represent identified errors, e_m represent masked errors, and e_c represent errors that have already been corrected.

However, the length of the initial block k_1 is a critical parameter, and should depend on the estimated error rate. An empirical result in [12] indicates that the optimal value of k_1 is 0.73/p, where p is the estimated error rate (QBER) [12]. The Cascade protocol is modified in [13], with the aim

of reaching the theoretical limit for protocol efficiency. From these results, it is clear that four iterations are sufficient for a successful key reconciliation, as was suggested in the [12]. However, since the initial block length depends on the estimated error rate, it is necessary to perform all the iterations. The number of iterations i is increased to the value for which the length of block k_i can be used to split the raw key into two parts ($k_i < \frac{n}{2}$).

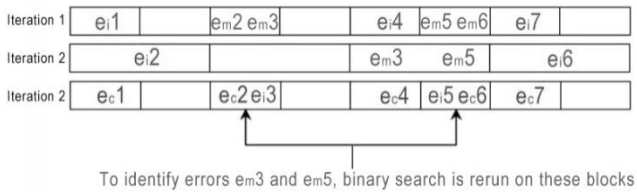

Fig. 1. Error detection process in cascade protocol.

Now, let us go back to the parity check results. If the parity of a block disagrees between Alice and Bob, they perform a binary search on that block with the aim of identifying the single bit error. The binary search consists of dividing the block in half and comparing the parity check results for the divided block until the error is located. This means a maximum of $1 + \lceil \log_2 k_i \rceil$ parity bits are exchanged for each block with an error bit since $\lceil \log_2 k_i \rceil$ is the maximum number of times block k_i can be divided, and one parity bit is exchanged for blocks without errors. In order to minimize the amount of information gained by Eve, it is advisable to discard the last bit of each block and sub-block for which the parity bit was exchanged. Now if we define L_i as the maximum number of leaked bits, and k_i as the length of the block in the i[th] iteration, it is clear that

$$\sum L_i = \sum_i \left(\sum_{\substack{initially \\ even \\ blocks}} 1 + \sum_{\substack{initially \\ odd \\ blocks}} \left(1 + \lceil \log_2 k_i \rceil\right) + \sum_{\substack{other \\ errors \\ corrected}} \lceil \log_2 k_i \rceil \right). \quad (2)$$

This can be shorted [14] to

$$L = \sum L_i = \sum_i \left(\frac{n}{k_i} + \sum_{\substack{errors \\ corrected}} \lceil \log_2 k_i \rceil \right), \quad (3)$$

where $k_i = 2k_{i-1}$, $k_i < \frac{n}{2}$ and n is the length of the initial key. Now it is clear that the number of leaked bits depends on the initial block size and error rate. From the results presented in [13], we can conclude that the majority of errors are corrected in the first two iterations.

TABLE I. PERCENTAGE OF CORRECTED ERRORS PER ITERATION.

Iteration	I	II	III	IV
Percentage of corrected errors	54.5223 %	45.3478 %	0.4517 %	0.002 %

We mark the length of the key after the key reconciliation phase as *F*.

E. Privacy Amplification

Alice and Bob finally have an identical key without errors, but since Eve may have gained significant knowledge of the key, Alice and Bob should strengthen their privacy. This is done by deleting some of the bits of the final key, so the raw key is shortened even more. The number of rejected bits during the privacy amplification process is defined in (4) [15], where S is the number of bits that need to be discarded and *n* is the length of the key (*B*).

$$\frac{n \cdot 2^{-S}}{\log 2} < 1. \quad (4)$$

We mark the length of the final key after this phase as *A*. Now we can compare the length of the key in each previous step

$$Q > B > R > F > A. \quad (5)$$

Finally, this means that the raw key (*Q*) must be significantly longer than the key after being reduced in all phases explained above (*A*). The final key must be long enough to be used for the encryption and decryption of confidential data.

III. CALCULATIONS

It is difficult to predict the precise length of the final key (*A*) since it depends on the error rate in the quantum channel and the Eve's influence. It also depends on the number of leaked bits and the number of iterations needed to discover the errors in the key. A short key cannot be used for strong encryption, while a longer key can be shortened. As such, it is necessary to define the length of the raw key (*Q*) which will result in a usable length of the final key (*A*). For testing purposes we used the payload size of the most popular VoIP codecs listed in Table II.

TABLE II. VOIP CODECS WITH BIT RATE AND PAYLOAD SIZE.

Codec & Bitrate	**Voice Payload Size (bits)**
G.711 (64 Kbps), G722_64k (64 Kbps)	1280
G.726 (32 Kbps)	640
G.726 (24 Kbps), G.728 (16 Kbps)	480
G.723.1 (6.3 Kbps)	192
G.729 (8 Kbps), G.723.1 (5.3 Kbps)	160

To summarize, we present the formula for calculating the length of the raw key (*Q*) from the length of the final key (*A*), error rate and parameter "level of security" $\hat{S}$(k).

$$Q = 2 \times \left\lceil A + \frac{\log \left\lceil \frac{Q}{2} \right\rceil}{\log 2} + S\frac{Q}{2} + L \right\rceil, \quad (6)$$

where S – percentage of raw key (Q) used for calculating

QBER; L – number of bits leaked during the key reconciliation phase; A – length of the final key.

Equation (6) is the Lambert W-function, also known as the Product Logarithm function, and it has two solutions in the real domain. We are interested in the second solution. The value of *L* for different values of error rate is shown in Table III, while the lengths (*Q*) of the raw key based on the length of the final key (*A*) and error rate for different security levels $\hat{S}(k)$ are shown on Fig. 3–Fig. 5.

TABLE III. MAXIMUM NUMBER OF LEAKED BITS (*L*).

QBER / Key length (bits)	0.01	0.05	0.10	0.15
1280	45	182	385	537
640	26	94	194	271
480	22	72	147	204
256	15	41	80	111
192	13	32	61	84
160	13	27	52	71
128	12	23	42	57

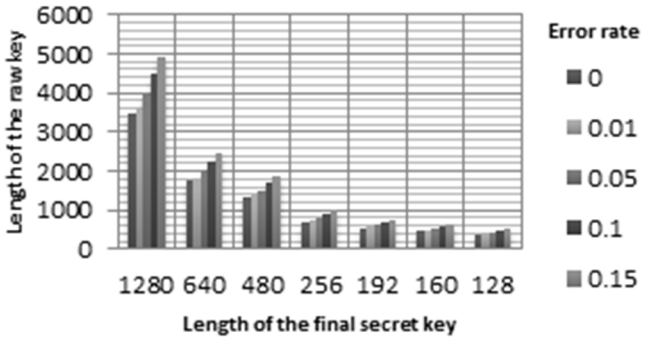

Fig. 3. Length of raw key (Q) based on final secret key (A) and error rate; S(K) = 0.06 => 25.30 %.

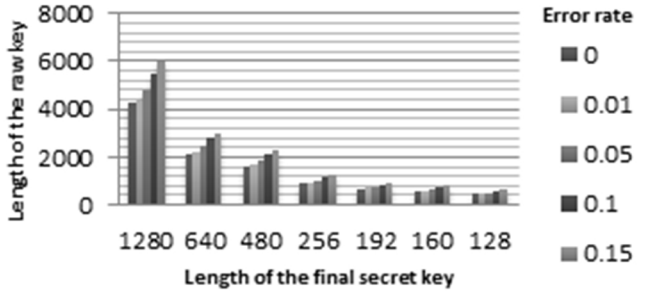

Fig. 4. Length of raw key (Q) based on final secret key (A) and error rate; $\hat{S}(k)$ = 0.11 => 39.10 %.

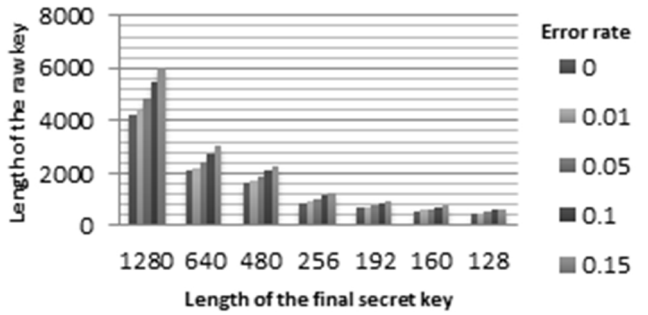

Fig. 5. Length of raw key (Q) based on final secret key (A) and error rate; $\hat{S}(k)$ = 0.15 => 50.25 %.

Table III shows how the maximum number of leaked bits (L) increases with error rate. It is easy to see that the values of leaked bits for error rate 0.01 increases more than double for error rate 0.05.

IV. VERIFICATION OF RESULTS

Following the publication of the Cascade protocol in 1994 [12], different implementations with different initial key size k_1, such as [13] and others, appeared. In this paper, the original concept of the Cascade protocol is used, where Equation 3 is used to calculate the maximum number of leaked bits. The initial block size k_i depends on the estimated error rate, which does not need to be equal to the actual error rate in the quantum channel. Additionally, Table I shows that 99 % of errors are corrected in the first two iterations, thus a further search for errors and leakage of bits is not needed. This means that in practice the values from (3) will not always be achieved. Since (3) is directly included in (6), it follows that (6) is insufficiently precise. This means that the output of (6) must be reduced by a value that will adjust the result to values that are common in practice. Let us consider the following example as a further explanation:

The length (*Q*) of raw key from (6) for input values A = 192, L = 13, p = 0.01 and S = 0.2530 ($\hat{S}(k)$ = 25.30 %) is calculated, and it is 573 bits. We tested this value with a QC simulator [7] and we obtained higher values of the final key length (*A*) than the required 192 bits. It means output from (6) should be reduced so it can match values obtained in practice. This was achieved using (7), and the values for which (7) has been reduced are presented in Fig. 6. To verify this modification we used the QC simulator [7] by generating more than 500,000 keys. We measured their dispersion, dependence on error rate *p* and $\hat{S}(k)$ values.

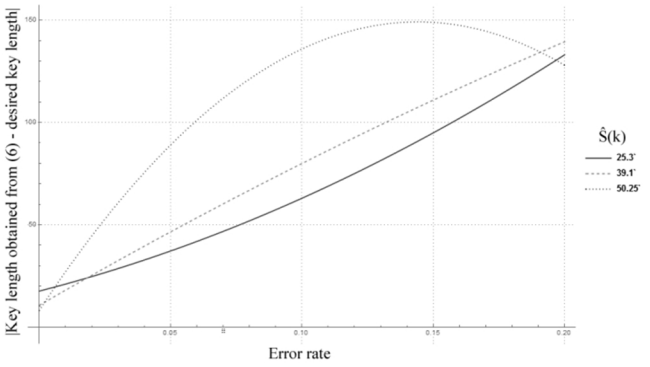

Fig. 6. Dependencies between values obtained from (6) and desired values for the defined error rate and $\hat{S}(k)$ parameter.

Figure 6 shows the difference between values calculated from (6) and the desired values. From this figure we can conclude that with a lower $\hat{S}(k)$ this difference is almost linear (value 25.30); however, as $\hat{S}(k)$ increases, the curve becomes increasingly rounder, finally reaching a parabolic shape for value 50.25.

$$Q = Q - \begin{bmatrix} (-17.6269 \cdot S(k)^2 + 1009.8 \cdot S(k) - 13027) \cdot p^2 \\ +(3.17073 \cdot S(k)^2 - 174.301 \cdot S(k) + 2710.74) \cdot p \\ +(0.0111203 \cdot S(k)^2 - 1.21994 \cdot S(k) + 41.318 \end{bmatrix}. \quad (7)$$

Equation (7) is written such that 90 % of results obtained are greater than the desired final key length (A) while remaining very close to this value, as shown in Fig. 7. The mean value of all measurements is greater than the desired final key length (A) while remaining very close to it.

Figure 7 and Fig. 8 are simple boxplot graphs where the top of the box marks the 75th percentile for the data set; the bottom of the box marks the 25th percentile for the data set; the central line marks the data set median; ends of the whiskers (lines extending from the boxes) mark the highest and lowest values of the data set that are within 1.5 times the interquartile range of the box edges (the fence); and the plus signs mark individual values outside the range of the whiskers.

From Fig. 7 we can conclude that there is less dispersion of values for a greater "level of security" parameter. In our experiment, standard deviation for $\hat{S}(k)$ = 25.30 %, 39.10 % and 50.25 % was 2.3147, 1.7481 and 1.2386, respectively. This behavior is a feature of $\hat{S}(k)$ function.

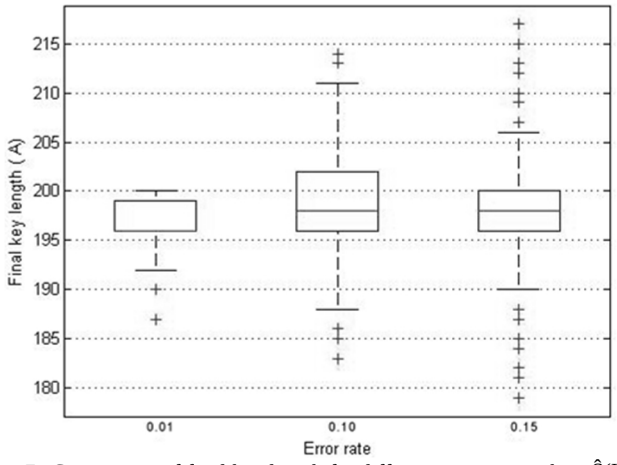

Fig. 7. Comparison of final key length for different error rates where $\hat{S}(K)$ = 0.11 => 39.10 %; the desired final key length is 192 bits.

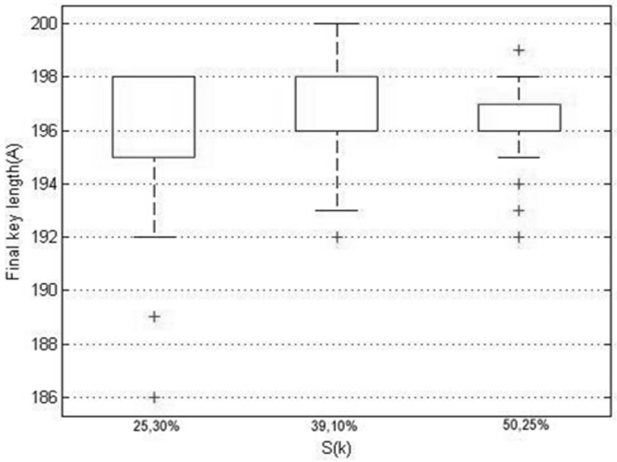

Fig. 8. Comparison of final key length for different values of the $\hat{S}(K)$ parameter; the error rate is 1 % and the desired final key length is 192 bits.

It is important to underline that the maximal tolerated quantum bit error rate (p_{max}) is defined as 12.9 % [5], [16], and for QBER values that are below this threshold ($p \leq p_{max}$) (6), (7) provide very accurate results.

V. CONCLUSIONS

It is difficult to predict the length of the final key (A) since it depends on the error rate in the quantum channel. In this article we present (6), (7) which are used to calculate the length of the raw key (Q) based on the length of the final key (A). For testing purposes we used the payload size of the most popular VoIP codecs, and we verified results with the QK simulator [7] with more than 500,000 experiments.

From (7), it is clear that the length of the final key increases with the parameter error rate (p) and the "level of security" $\hat{S}(k)$. Figures7 and Fig. 8 show the values are spread more widely when the error rate is higher, and when the value of $\hat{S}(k)$ is lower. It is worth noting that the influence of eavesdropping is not included in (6) since the entire QKD process will be repeated if the estimated QBER is higher than maximal tolerated QBER (p_{max}).

The main contribution of this paper lies in providing a way of calculating the raw key length based on the desired final key length used in QKD. Using equations presented in this article it is possible to accurately calculate the amount of raw key material which is subsequently processed and stored in storage reservoirs in order to be used later to encrypt data traffic. Equations presented here can be used to calculate the key consumption of other traffic, regardless of VoIP.

REFERENCES

[1] C. Elliott, H. Yeh, *DARPA Quantum Network Testbed*. BBN Technologies Cambridge: New York, USA, 2007.

[2] R. Alleaume, J. Bouda, C. Branciard, *et al*., "SECOQC white paper on quantum key distribution and cryptography", *arXiv preprint quant-ph/0701168*, p. 28, 2007.

[3] M. Sasaki, "Tokyo QKD network and the evolution to secure photonic network", *CLEO: Science and Innovations. Optical Society of America*, vol. 1, pp. 4–6, 2011. [Online]. Available: http://dx.doi.org/10.1364/cleo_at.2011.jtuc1

[4] C. E. Shannon, "Communication theory of secrecy systems", *Bell system technical journal*, vol. 28, no. 4, pp. 656-715, 1949. [Online]. Available: http://dx.doi.org/10.1002/j.1538-7305.1949.tb00928.x

[5] C. Kollmitzer, M. Pivk, *Applied Quantum Cryptography*. Lectures Notes in Physics, vol. 797, p. 230, 2010. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-04831-9

[6] S. Marksteiner, *An approach to securing IPsec with Quantum Key Distribution (QKD) using the AIT QKD software*. CAMPUS 02 University of Applied Sciences, Graz, Austria, 2014.

[7] M. Niemiec, L. Romanski, M. Swięty, "CCIS 149 - quantum cryptography protocol simulator", *Multimedia Communications, Services and Security Communications in Computer and Information Science*, vol. 149, pp. 286–292, 2011. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-21512-4_34

[8] A. K. Ekert, "Quantum cryptography based on Bell's theorem", *Physical Review Letters*, vol. 67, pp. 661–663, 1991. [Online]. Available: http://dx.doi.org/10.1103/PhysRevLett.67.661

[9] V. Scarani, A. Acin, G. Ribordy, N. Gisin, "Quantum cryptography protocols robust against photon number splitting attacks for weak laser pulse implementations", *Physical review letters*, vol. 92, p. 057901, 2004. [Online]. Available: http://dx.doi.org/10.1103/PhysRevLett.92.057901

[10] C. H. Bennett, G. Brassard, "Quantum cryptography: public key distribution and coin tossing", in Proc. *IEEE Int. Conf. Computers, Systems and Signal Processing*, vol. 175, no. 150, p. 8, 1984.

[11] M. Niemiec, A. R. Pach, "The measure of security in quantum cryptography", 2012 *IEEE Global Communications Conf. (GLOBECOM)*, pp. 967–972, 2012. [Online]. Available: http://dx.doi.org/10.1109/GLOCOM.2012.6503238

[12] G. Brassard, L. Salvail, "Secret-key reconciliation by public discussion", *EUROCRYPT 1993, Lecture Notes In Computers Science*, vol. 765, pp. 410–423, 1994. [Online]. Available: http://dx.doi.org/10.1007/3-540-48285-7_35

[13] T. Sugimoto, K. Yamazaki, "A study on secret key reconciliation protocol", *IEICE Trans. Fundamentals of Electronics, Communications and Computer Sciences*, vol. E83-A, no. 10, 2000.

[14] N. Ruth Li-Yung, "A probabilistic analysis of binary and cascade". [Online] Available: http://math.uchicago.edu

[15] M. Niemiec, *Design, Construction and Verification of a High-Level Security Protocol Allowing to Apply the Quantum Cryptography in Communication Networks*. AGH University of Science and Technology, Krakow, Poland, 2011.

[16] G. Smith, J. Renes, J. Smolin, "Structured codes improve the Bennett-Brassard-84 quantum key rate", *Physical Review Letters*, vol. 100, no. 17, p. 170502, 2008. [Online]. Available: http://dx.doi.org/10.1103/PhysRevLett.100.170502