Chapter 3
Quantum Key Distribution

M. Pivk

In this chapter a complete QKD protocol is presented, starting from the transmission via the quantum channel up to the communication over the public channel. The protocol described here is the BB84 protocol, named after Bennett and Brassard [5]. There are other protocols like the B92 protocol [3], the six-state protocol [8], the SARG protocol [19] and the Ekert protocol [10], which are not discussed here. We are focusing on BB84, the most known QKD protocol. In Fig. 3.1 you see an abstract sequence diagram of BB84. k_a is the pre-shared secret needed for authentication and K is the final key generated after BB84 is executed. We begin with the first stage, the transmission of the photons, which is the physical representation of the qubits, from Alice to Bob. This phase of the protocol is discussed in detail in Sect. 3.1. Afterward the communication switches to the public channel (Sect. 3.2).

There, the first phase is the sifting phase, where Alice and Bob negotiate which bits are used and which bits are discarded. To avoid a man-in-the-middle attack by Eve, this message exchange must be authenticated. After agreeing on the bits and being sure that Eve has not modified messages by using an authentication scheme, Alice and Bob go on to the reconciliation phase or error correction phase. Because the quantum channel is not a noiseless channel, Alice and Bob do not share the same identical string. There is a small portion of errors in Bob's string, which is corrected in this phase. Again Eve has the possibility to modify messages during this phase to her interest. Therefore, Alice and Bob must authenticate this phase. Passing reconciliation, Alice and Bob share a string, which is identical with very high probability. But this string cannot be used as a key yet. Eve's information about the string must be considered. She has gained information during the error correction and maybe also during the quantum transmission. Hence, Alice and Bob must map their string via a function to a smaller subset, so that Eve's knowledge decreases almost to zero. This stage is called privacy amplification and afterward Alice and Bob share a secret key only known by them.

M. Pivk (✉)
Safety & Security Department, Quantum Technologies, AIT Austrian Institute of Technology GmbH, Lakeside B01A, 9020 Klagenfurt, Austria, mario.pivk@ait.ac.at: http://www.ait.ac.at

Pivk, M.: *Quantum Key Distribution*. Lect. Notes Phys. **797**, 23–47 (2010)
DOI 10.1007/978-3-642-04831-9_3

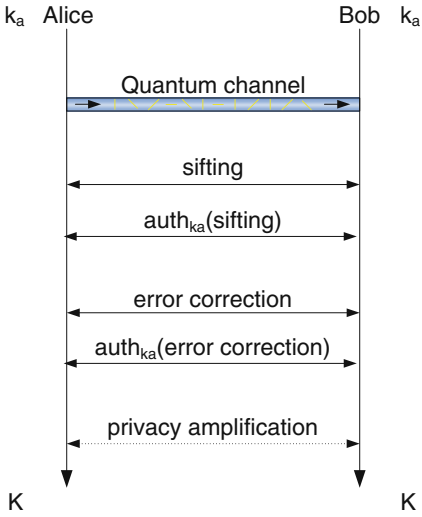

Fig. 3.1 BB84. k_a is the pre-shared key. K is the generated key after protocol execution

3.1 Quantum Channel

As mentioned, the first stage of the BB84 protocol is the quantum transmission. Therefore, we take a closer look at the physical realization: How can qubits be represented and transmitted? In the second part we discuss what Alice and Bob have to do, when they perform the BB84 protocol. Additionally, an equation is given, which represents the theoretical throughput of the quantum channel.

3.1.1 Physical Realization

One way to represent a qubit is by using single photons. They are very suitable for QKD, because photons hardly interact with each other and they can overcome long distances with low loss in optical fibers. Polarization is one of the photon's attributes which can serve as a qubit property. There exists rectilinear polarization (horizontal/0° and vertical/90°) and diagonal polarization (+45° and −45° shifted). We can map horizontal and vertical with the qubits states $|0\rangle$ and $|1\rangle$, and +45° and −45° shifted with the states $|+\rangle$ and $|-\rangle$, respectively. Thus, we have the same statistical probabilities as discussed in Sect. 2.1.3, which we will use next.

There are many different systems for implementing QKD. In Chap. 6 seven systems are presented which are developed for SECOQC.

One possibility to generate single photons is by attenuating the output of a laser. This attenuator is characterized by the mean photon number, which defines the rate of photons passing the attenuator. Usually, the mean photon number is 0.1, meaning that 90% of the time no photon passes the attenuator but in 10% of cases, the probability that a single photon passes is better than 95% (for more information on the computation of the probability see [17]). Nevertheless, the

output of the laser may consist of several path; in BB84 due to the amount of different polarization states four are needed. But after a pulse passes the attenuator it has an arbitrary polarization; therefore, linear polarizers and half-plates are used to create the desired polarization. These elements can be seen as linear operators (Pauli matrices) known from Sect. 2.1.2 performing the bit and phase flip on the qubits.

After the photons have the right polarization they pass through a beam splitter, where they are combined. Now the pulse is transmitted over the quantum channel (optical fiber or free space) and arrives at the receiver. There, a random choice is made by the receiver as to which polarization filter to use (e.g., horizontal or $+45^\circ$). If the receiver chooses differently to the sender we know from Sect. 2.1.3 that there is a chance of 50% for the photon to pass the filter (and change the polarization to horizontal) or to get absorbed (this holds for both $+45^\circ$ and -45° polarized photons). If the receiver chooses the same basis as the sender the photon passes with 100% probability if it was polarized horizontal and with 0% probability if it was polarized vertical. Behind the filter a detector clicks every time a photon passes. (this sending–receiving architecture is taken from [22]).

3.1.2 Photon Transmission and Throughput

Back to our scenario, Alice is on the transmitter and Bob on the receiver site of the above-described quantum channel. Alice chooses now randomly two strings independent of each other with length m. The first string represents the basis for the quantum transmission and the second the proper value of the specific bit. Alice and Bob have the same mapping scheme in common. A sample mapping is defined in Tables 3.1 and 3.2.

Table 3.1 Base mapping

Base	Representation bit
Rectilinear	0
Diagonal	1

Table 3.2 Value mapping

Rectilinear	Diagonal	value bit
Horizontal (0°)	$+45^\circ$	0
Vertical (90°)	-45°	1

Alice starts now to transmit. Therefore, she takes the first bit of the first string, indicating which base to use (e.g., a 0-bit denotes to use the rectilinear basis), and the first bit of the second string indicating which value to take (e.g. a 1 denotes in the rectilinear base to polarize the photon vertical and in the diagonal base to shift the polarization by -45°). She applies this procedure on all m-bits of both strings

and sends them as photons via the quantum channel to Bob. Bob on his part chooses also a random string with length m, for his base choices. With these bits he measures the incoming photons with the corresponding filter. Note that it makes no sense to measure with both filters, because after measurement the original polarization is lost as mentioned in the previous section or in Sect. 2.1.3. Bob keeps the measurement results to himself. Consider that Bob's measurements do not completely match with Alice's ones, due to optical misalignment, disturbance on the quantum channel, noise in Bob's detectors, or the presence of an eavesdropper Eve, although both choose the same basis.

Before describing the next step, we continue by analyzing the throughput of the quantum channel. In [24] a theoretical function is given by which the several losses on the quantum channel can be computed:

$$g_q = \mu \cdot \alpha_f \cdot \alpha_e \cdot \eta_{\text{det}} \cdot k_{\text{dead}}. \tag{3.1}$$

The gain of the quantum channel g_q consists now of the mean photon number μ, which we discussed in the previous section. The next factor α_f represents the fiber loss. This loss is distance dependent and increases with higher distances. For the receiver's detector η_{det} as detection efficiency and k_{dead} as factor accounting for the reduction of the photon detection rate due to the dead time is given. The dead time is the hold-off time following each detection event; during this time the bias voltage of the device is below a certain level such that no photon can be detected. α_e is the additional loss of the system.

To compute the number of Bob's measurement results the data rate f_{data} is required, which is the number of photons Alice's laser can send. This rate is given in hertz and can have range from MHz to GHz. These different rates have impact on the gain of the quantum channel: with higher rates the efficiency shrinks. So the key material Bob receives per second can be computed by $f_{\text{data}} \cdot g_q$. The Eq. 3.1 conforms with experimental test done in [24].

As long as the strings which Alice and Bob need for the choice of bases are randomly chosen (the probability to send a qubit in the rectilinear base or diagonal base is 50%), half of Bob's results, i.e., the raw key, must be discarded because of the independence of both strings (see later in Sect. 3.2.1). To minimize this loss of raw key bits a concept was designed in [13], which increases the efficiency. Instead of having the same probability for 0's and 1's in the string, the authors propose to increase the probability for one of them. In fact the photons will be transmitted more often in one base than in the other. Thus, the rate in which Alice and Bob select the same base will increase. We will discuss this more precisely in Sect. 3.2.1.

Finally, the communication between Alice and Bob on the quantum channel is finished. They now switch to the public channel. So far, Alice holds the two strings of length m containing her choice of base and value and Bob a string of length m containing his choice of bases and his measurement results of length $g_q \cdot m$. The smaller size is due to the loss on the quantum channel.

3.2 Public Channel

The communication via the public channel is necessary for Alice and Bob, since they must negotiate on which bits they perform the next steps. Because they choose their bases for the quantum transmission randomly they must know in which cases they used the same basis. After agreeing on them, the errors must be corrected, since the communication via the quantum channel is noisy. And as a last step Alice and Bob must reduce Eve's knowledge, which she gained during the protocol. By intercepting and resending photons Eve gains knowledge of the raw key (this issue is reflected by the error rate) and also eavesdropping on the messages on the public channel increases her knowledge. If Eve has the possibility to modify messages, she can start a man-in-the-middle attack and then share keys with both disguised as one's peer.

3.2.1 Sifting

The first phase on the public channel is *sifting*. After Alice has sent random bits mapped into randomly chosen quantum bases via photons, further steps are required such that she shares the same bit string with Bob. The first thing Alice must know is which photons Bob has measured. By reason of the fact that only a small fraction of pulses contain photons, and several losses on the quantum channel and at Bob's detector, Alice does not know which bit Bob received. Therefore, Bob sends a message on the public channel to inform Alice which photons he has measured. The easiest way is to send a string as long as the string chosen at the beginning for the bases, which we indicate with length m. Bob sets now the position where he was able to make a measurement to 1 and the other positions to 0. Since we want to save communication traffic, we can benefit from the fact that the gain of the quantum channel g_q (Eq. 3.1) is a very small factor (<0.01). Bob must only tell Alice which position he received. To represent a position $\log_2 m$-bits are necessary. Let $2n$ be the length of the raw key, which is equal to $g_q \cdot m$, then $2n \log_2 m$-bits are necessary to tell Alice the measured positions. This would be more efficient if the following equation is fulfilled:

$$2n \log_2 m = g_q \cdot m \log_2 m < m.$$

We know that the gain of the quantum channel is less than 0.01, so it is sufficient that $m < 2^{100}$. The sending rate is maximal in the range of GHz ($10^9 \approx 2^{30}$), hence adequate for this sifting transmission mode.

With this message Alice knows only which position Bob has measured but not if he has used the same basis, because they have randomly and independently chosen the bases. Thus, Bob sends a second message, in which he publishes the bases he used. An optimized approach sends only those positions where he has successfully measured a photon. Thus, the length of the second message is $2n$. Note that Bob sends only his sequence bases (see Sect. 3.1.2) reduced by the bits where no

measurement was possible, and not the measurement results. After Alice receives both messages she can reduce her secret bit value string by canceling out those positions which Bob did not receive (Bob's first message) and those positions where she uses different bases at transmission (comparison by her base string and Bob's second message). To make sure Bob shares the same string with her. Alice sends a message to Bob containing her choice of bases for those positions where Bob received a photon. Bob himself performs the same procedure as Alice and cancels out those positions with different bases. Finally, the sifting phase is over and Alice and Bob share a secret key: the so-called sifted key. However, error remains due to the noisy quantum channel.

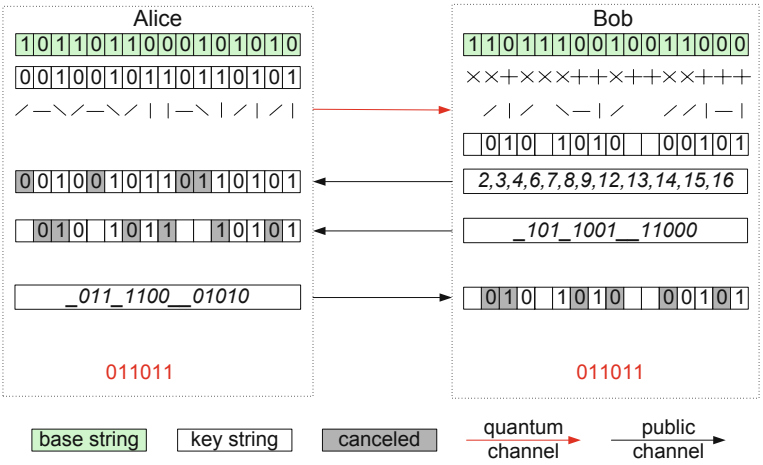

Fig. 3.2 QKD protocol until the end of the sifting phase

But what is the length of this key? Since Alice and Bob have chosen the two bases for transmission and measuring randomly and independently, the probability that they use the same basis is $\frac{1}{2} \cdot \frac{1}{2} + \frac{1}{2} \cdot \frac{1}{2} = \frac{1}{2}$. The length of the sifted key reduced by mismatching bases from the raw key is $\frac{2n}{2} = n$. The complete process until the output of the sifted key is demonstrated in an example in Fig. 3.2. The sending key length is $m = 16$. Assuming that the gain of the quantum channel is given by $g_q = 0.75$ (which is not a realistic value), we use Tables 3.1 and 3.2 for mapping bits to photons. So the sifted key length is $n = m \cdot g_p \cdot \frac{1}{2} = 6$.

As mentioned in Sect. 3.1.2 it is suggested [13] to choose the strings representing the photon bases with different probability. So far, we chose the rectilinear or the diagonal basis with probability $\frac{1}{2}$. Assume now that Alice and Bob choose a basis with probability $\delta \in [0, 1]$, whereby the other basis is used with complementary probability $1-\delta$. The protocol gain for the normal BB84 is 0.5 as we have computed before; the protocol gain g_p for the case that the probability of base usages can be changed is

$$g_p = \delta \cdot \delta + (1 - \delta) \cdot (1 - \delta) = 2\delta^2 - 2\delta + 1. \tag{3.2}$$

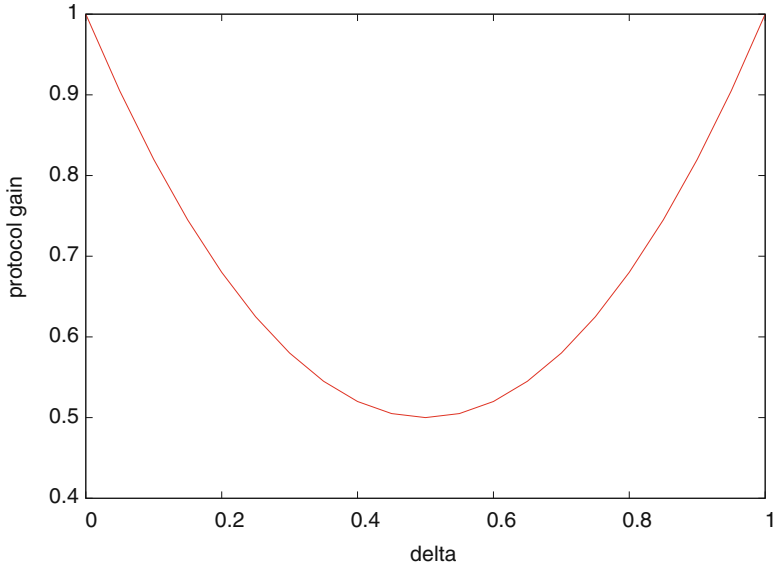

Fig. 3.3 Protocol gain g_p when delta δ is variable (Eq. 3.2)

The fact that δ is variable gives us the probability to increase the protocol gain. As you can see in Fig. 3.3 the gain g_p rises to 1 if δ goes to 1 (to 0 – analog for the other basis). So the case $\delta = 0.5$ would be the worst case.

But what happens with the security if δ goes to 1. In Sect. 3.2.3.1 we will discuss this in detail. To say it right away, the chance for Eve to eavesdrop the photons without being detected increases because Alice and Bob must negotiate at least which bases to use with higher probability, and Eve can use this information. The errors Eve produces decrease in the set where the basis is chosen with higher probability and increase in the set of the other basis. Using the naive error estimation, which calculates the error probability of both sets together, Alice and Bob will fail to detect Eve, because the proportion of the sets changes. Finally, when δ reaches 1, Eve will not introduce any errors. Unless $\delta = 1$, if we use a refined error analysis, which reflects the error probability of both sets (detailed explanation in Sect. 3.2.3.1), Eve can still be detected. In [13] it is described that the set of bits after sifting must have enough samples to make an accurate error estimation. Therefore, some constraints on δ are defined. Assume that m_t photons are transmitted from Alice to Bob, which Bob was able to measure. Then, on average, only $m_t(1-\delta)^2$ photons belong to the case where the basis with the smaller probability is used. The size of this set was analyzed in [13] with the information we allow Eve to obtain on the final key. If we except a fixed but arbitrarily small amount of information on the final key, the number of this set is required only to scale logarithmically with the length of the final key k. Thus, δ must fulfill the following equation:

$$m_t(1-\delta)^2 \geq O(\log k),$$
$$\delta \leq 1 - O\left(\sqrt{\frac{\log k}{m_t}}\right),$$
$$\text{if } m_t \to \infty \text{ then } \delta \to 1 \text{ and } g_p \to 1. \tag{3.3}$$

Note that δ never reaches 1 and the fact that the protocol gain goes to 1 can easily be seen in Fig. 3.3.

3.2.2 Authentication of Sifting

Due to the possibility of a man-in-the-middle attack (combination of intercept/resend attack, see Sect. 5.2.1, and modification of messages on the public channel) by Eve we must ensure that the sifting phase is authenticated by Alice and Bob. We know this phase consists of three messages exchanged between Alice and Bob. Two messages sent by Bob, where he tells Alice which photons he was able to measure and which bases he used for the measurement. The last message completing sifting phase is sent by Alice, where she tells Bob which bases she used for those photons Bob measured. To authenticate those messages we use Wegman–Carter authentication, as explained in Sect. 2.2.2. For Bob's two messages and for Alice's message we generate authentication tags and append them to the respective messages.

Because Wegman–Carter authentication is symmetric, we have to use a secret key shared by Alice and Bob. Since key material is generated in every iteration a small part of it can be used for the authentication of the next round. Unfortunately, when we start the first round a pre-shared key must be available, which is exchanged previously through a secret channel (face-to-face or in another way). In order that key material remains after withdrawal by authentication, the choice of the algorithm is a major concern. Following Eq. 2.35 the authentication cost for Wegman–Carter is $4 \cdot ((b + \log_2 \log_2 a) \cdot \log_2 a)$ for an input size of a and an authentication tag size of b. The key size grows only in the logarithmic scale, so it would be useful for our intended purpose.

Similar to [11] we compute now the authentication cost for the sifting phase. In the previous section we derived that the length of Bob's first message is $2n \log_2 m$ and the length of the second message is $2n$. The sum is $2n(1 + \log_2 m)$ and thus the authentication key length w_1 needed for the tag is

$$w_1 = 4 \cdot \left(g_{\text{auth}} + \log_2 \log_2 \left(2n\left(1 + \log_2 m\right)\right)\right) \cdot \log_2 \left(2n\left(1 + \log_2 m\right)\right), \tag{3.4}$$

where g_{auth} is the length of the resulting tag. Alice and Bob compute their tag using the hash function indexed by the authentication key and Alice compares if the tags match.

If Alice verifies that the message is from Bob she sends her message with her choice of bases, which has length $2n$. We denote the authentication key length for Alice's message as w_2 and the length is computed as

$$w_2 = 4 \cdot \left(g_{\text{auth}} + \log_2 \log_2(2n)\right) \cdot \log_2(2n). \tag{3.5}$$

If Bob determines that the generated tag with cost w_2 matches with Alice's tag both can be sure that the probability that Eve modified the message is at most ε, which is the property of the class of hash function (ε-ASU_2).

The required key material to authenticate the sifting phase is the sum of all costs w_1, w_2 (Eqs. 3.4 and 3.5) and results in the total sifting authentication costs

$$\begin{aligned} t_S = &4 \cdot \left(g_{\text{auth}} + \log_2 \log_2 \left(2n \left(1 + \log_2 m\right)\right)\right) \cdot \log_2 \left(2n \left(1 + \log_2 m\right)\right) \\ &+ 4 \cdot \left(g_{\text{auth}} + \log_2 \log_2(2n)\right) \cdot \log_2(2n). \end{aligned} \tag{3.6}$$

3.2.3 Reconciliation

The reconciliation stage is split into two parts. The first major part is the error estimation. Its aim is to find the correct error rate. To guarantee an optimal error correction, part two of reconciliation. The error correction has the task of correcting all errors in Bob's string via public discussion. Here some information about the string must be disclosed and exchanged between Alice and Bob.

3.2.3.1 Error Estimation

The error estimation is an important step in the QKD protocol to determine the proper error rate of the sifted key. Usually, in BB84, the error rate p is estimated by picking a small random subset of bits with length r from those given in the sifted key. This test string is publicly compared by Alice and Bob and yields in a certain number of errors e. If the length of the test string is chosen adequate to the length of the sifted key n, the error probability is

$$p = \frac{e}{r}. \tag{3.7}$$

Of course, since a part of the sifted key has been announced, those bits must be deleted to avoid information leakage to Eve. This elimination has little effect on the length of the final key if the length of the sifted key is large and if the error estimation is not executed every round. The error rate found for a round can also be used in the following rounds.

If the error rate p turns out to be very large, then either eavesdropping has occurred or the channel is somehow unusually noisy. However, the sifted key is discarded and Alice and Bob may re-start the whole protocol again (on another quantum channel). This threshold $p_{\max}$, for which the rate should not exceed $p \leq p_{\max}$, can be set to, e.g., 11% because at the moment the best error correction code

approaches a maximal tolerated error rate of 12.9% [21]. If the error rate p is reasonably small ($p \leq p_{\max}$), Alice and Bob can continue with error correction.

An improved error estimation was presented in [1] and [13]. In the refined error analysis there does not compute a single error rate but two values are estimated. The idea is not to merge the measurements during the quantum transmission, where Alice and Bob have used the same basis, into one set (more precisely the sifted key) and choose the random test subset out of it. They choose two random test subsets. The first subset consists of all measurements where Alice and Bob have used the first basis (e.g., rectilinear) and the second one where they used the second basis (e.g., diagonal). Let r_1 be the length of the first subset and r_2 be the length of the second subset just before Alice and Bob publicly compare both strings and estimate the number of errors e_1 and e_2, respectively, which leads to the error rates

$$p_1 = \frac{e_1}{r_1}, \tag{3.8}$$

$$p_2 = \frac{e_2}{r_2}, \tag{3.9}$$

where p_1 is the error rate for the photons measured with the first basis and p_2 those with the second basis and

$$p = \frac{p_1 + p_2}{2}. \tag{3.10}$$

The advantage compared to the single error rates is as follows: If Eve starts a specific attack like the biased eavesdropping strategy (a specialization of intercept and resend, see Sect. 5.2.1). Then she chooses the probability q_1 for measuring each photon sent from Alice to Bob in the first basis (e.g., rectilinear) and a second probability q_2 for measuring each photon in the second basis (e.g., diagonal). Hence, with probability $1 - q_1 - q_2$ she does not measure the photon. When Alice and Bob use the same basis errors occur only if Eve uses a different basis. Regarding Bob's side, the photons measured by Eve are randomized and yield an incorrect bit in half of these cases. This introduces an error rate of $p_1 = \frac{q_2}{2}$ for the first basis. Similarly, for the second basis an error rate $p_1 = \frac{q_2}{2}$ is obtained (without respect to the normal noise on the quantum channel). If we apply these two error rates and Eq. 3.10 to the requirement $p \leq p_{\max}$ it results in $(q_1 + q_2) \leq 4p_{\max}$. Eve has the possibility to vary her probability q_1 in a big range. In contrast if we use the constraint $p_1, p_2 \leq p_{\max}$ which has the same property as the single error rate constraint in a random noisy channel, Eve's possibility to choose the probabilities is shortened with $q_1, q_2 \leq 2p_{\max}$. With the refined error analysis the biased eavesdropping strategy is avoided (e.g., $q_1 = 3p_{\max},\ q_2 = 0$ is accepted when we measure only one error rate, but rejected by the detailed error measurement).

In Sect. 3.2.1 we have heard about a strategy to vary the probability of used bases at sending and receiving, to boost the protocol gain g_p (in standard BB84 protocol the probability for sending in one of the bases is $\frac{1}{2}$ as described in Eq. 3.2).

The difficulty now is that with increasing or decreasing δ, a biased eavesdropping strategy where Eve aware of δ is more difficult to detect using the simple error estimation rate. In this case the simple error rate is

$$p = \frac{\delta^2 p_1 + (1-\delta)^2 p_2}{\delta^2 + (1-\delta)^2} = \frac{\delta^2 q_1 + (1-\delta)^2 q_2}{2(\delta^2 + (1-\delta)^2)}. \tag{3.11}$$

If Eve chooses the strategy $q_1 = \delta$ and $q_2 = 1 - \delta$ we see in Fig. 3.4 that when $\delta \to 1$ or $\delta \to 0$ the error probability $p \to 0$ unlike the refined error analysis where $\delta \to 1$ then $p_1 \to 0$ but $p_2 \to 0.5$. Regardless of how δ is chosen ($0 < \delta < 1$) if the measurement set is quite big (as defined in Sect. 3.2.1) then there would be no problem to detect Eve, which is not possible with the simple error analysis.

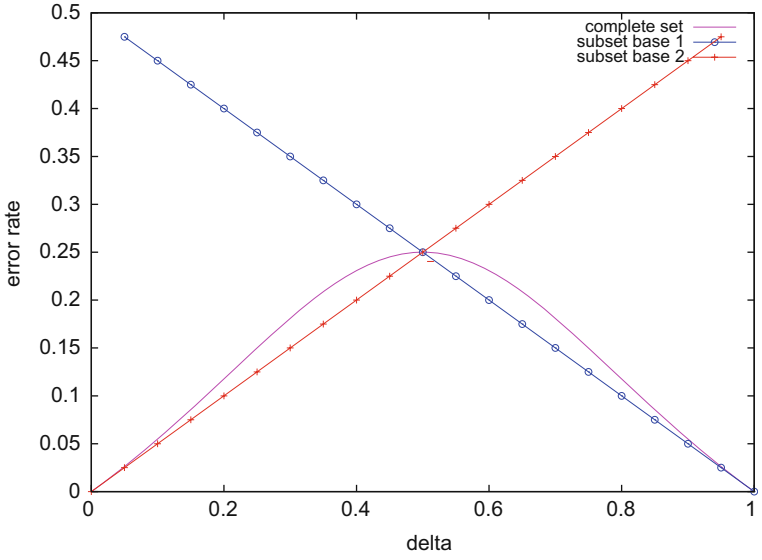

Fig. 3.4 Error rates (variable δ) after a biased eavesdropping attack with $q_1 = \delta$

3.2.3.2 Error Correction

Henceforth we will model the quantum channel as binary symmetric channel (*BSC*). In a $BSC(p)$ we have an alphabet of two symbols (0, 1) which are transmitted over the channel from a sender to a receiver. With probability p noise is added to the transmitted symbol, meaning the receiver gets a 1 instead of a 0 and a 0 instead of a 1, respectively. For every symbol we have a probability $1 - p$ that it will be correctly received (see Fig. 3.5).

After the sifting phase Alice holds a random string sk_a and Bob sk_b with equal length $|sk_a| = |sk_b| = n$. The difference between these two strings depends on the quantum bit error rate (*QBER*) p of the channel. Starting the QKD protocol

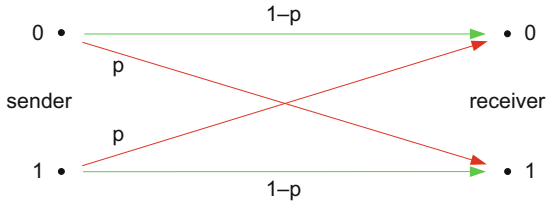

Fig. 3.5 Binary symmetric channel (BSC): symbols are exposed to noise with probability p

this probability p is unknown or rests on empirical values. Further on, the error estimation phase as presented in Sect. 3.2.3.1 gives a good approximation of the error rate. Observe that dist (sk_a, sk_b), the amount of places in which both strings differ (Hamming distance), is nearly np. Setting $sk_a = A$ and $sk_b = B$ (A and B are random variables) the conditional entropy (Eq. 2.38) of A given B is

$$H(A|B) = H(A \oplus B) = nh(p),$$

where $h(p)$ is now short for Shannon entropy $H(X)$ and the random variable X is a Bernoulli trial with parameter p (for each $x \in \mathcal{X}$, $p(x)$ is the same).

The task is to correct Bob's string B without disclosing enough information that gives Eve chances to reconstruct the very same string. Therefore, a reconciliation protocol R^p is defined, which runs on both strings A, B and results in the string S by exchanging some information Q on the public channel. We write $S = \perp$ if the protocol fails to produce S. Since Q must be exchanged on the public channel, an eavesdropper Eve can gain some information on S, which can be expressed by $I_E(S|Q)$. This is the expected amount of bits that an eavesdropper Eve can get on S given Q.

Brassard and Salvail [7] have formulated some definitions to characterize such reconciliation protocols. The first definition deals with the robustness.

Definition 3.1 A reconciliation protocol R^p is ε-robust if

$$(\exists N_0(\varepsilon))(\forall n \geq N_0(\varepsilon)) \sum_{\alpha,\beta \in \{0,1\}^n} prob(A = \alpha, B = \beta)\ prob(R^p(\alpha, \beta) = [\perp, \cdot]) \leq \varepsilon$$

where $0 \leq \varepsilon \leq 1$.

If a protocol is ε-robust the probability to fail is maximal ε. The next theorem is a direct consequence of the noiseless coding theorem.

Theorem 3.1 $(\forall p \leq \frac{1}{2})$ $(\forall$*reconciliation protocol* $R^p)$ *If there exists* $0 \leq \varepsilon \leq 1$ *such that* $R^p = [S, Q]$ *is* ε*-robust then*

$$\lim_{n \to \infty} \frac{I_E(S|Q)}{nh(p)} \geq 1,$$

where n is the length of the transmitted string.

In other words, Eve's information of S is greater than or equal to the information of A given B. Eve cannot have less than this information if the reconciliation protocol R^p is successful. But if these two pieces of information are equal the protocol is optimal. Let us define it by the next definition:

Definition 3.2 A protocol R^p is optimal if

$$(\forall \varepsilon \geq 0)[R^p = [S, Q] \text{ is } \varepsilon\text{-}robust]$$

and

$$\lim_{n\to\infty} \frac{I_E(S|Q)}{nh(p)} = 1,$$

where the public channel is a *BSC(p)*.

In [7] a construction of a optimal protocol is presented. The basic idea is that Alice creates a random label of her string A with length approximately $nh(p)$, when the length of string A is n. If Alice sends this label over the channel, Eve would only have this information and so Definition 3.2 holds, the protocol is optimal. When Bob obtains the label $f(A)$, he computes all possible inputs, resulting in a set $\mathcal{S}\{B'|f(B') = f(A)\}$. The string $B' \in \mathcal{S}$ with minimal Hamming distance from B is the desired S. Brassard and Salvail [7] described this protocol as unpractical, because Alice and Bob require 2^{m2^n} functions. But if the function $f(x)$ is chosen from a $universal_2$ class of hash functions (see Definition 2.5) the protocol remains optimal. The specification of a $universal_2$ class of hash functions can be done in a short and efficient way. The only problem which remains is that there are no known efficient algorithms to compute the set $\mathcal{S}\{B'|h(B') = h(A)\}, h \in \mathcal{H}$, if the hash value of A is known.

The goal is to create an efficient reconciliation protocol, which is defined as follows:

Definition 3.3 A reconciliation protocol R^p is efficient if there is a polynomial $t(n)$ such that $\bar{T}^{R^p}(n) \leq t(n)$ for n sufficiently large, where n is the length of the strings transmitted over the secret channel and $\bar{T}^{R^p}(n)$ represent the expected running time of R^p, given an n-bit long input string.

Definition 3.4 A reconciliation protocol R^p is ideal if it is both optimal and efficient.

Brassard and Salvail determined when their optimal protocol becomes ideal, using the $universal_2$ class of hash function H_3 [9]. They proved that their protocol is ideal if and only if $NP \subseteq BPP$, which is a hypothesis.

With the fact that a ideal reconciliation protocol depends on a open question in complexity, which is unlikely to be true, we have to find another solution. In Sect. 2.2.1 (universal hashing) we had a similar problem. There we define a $strongly\ universal_2$ class of hash functions, where the size of this class becomes too large and unpractical. But just a small change of the probability on the theoretical bound makes these classes useful. Here again if we do not demand optimality and allow the reconciliation protocol to transmit a small amount of leaked information above the theoretical bound the protocol becomes efficient and useful. In [7] this characterization was defined as in Definition 3.5.

Definition 3.5 A reconciliation protocol R_ζ^p is almost ideal if for all $\zeta \geq 0$ we have

1. $(\exists \varepsilon \geq 0)[R_\zeta^p = [S, Q]$ is ε-robust]
2. $\lim_{n\to\infty} \frac{I_E(S|Q)}{nh(p)} \leq 1 + \zeta$
3. $(\exists$ polynomial $t)(\exists N_0(\zeta))(\forall n \geq N_0(\zeta))[\bar{T}^{R_\zeta^p} \leq t(n)]$

for n the length of the string transmitted over the $BSC(p)$.

An *almost-ideal* protocol has an error probability bounded by ε approaching 0 for increasing n (see Definition 3.51). The amount of leaked information is allowed to be slightly greater than the theoretical bound (see Definition 3.52), but the parameter ζ indicating the excess of information is chosen by Alice and Bob before the start of the protocol. After their choice the expected runtime $\bar{T}^{R_\zeta^p}$ of the reconciliation protocol must be bounded by a polynomial (see Definition 3.53).

Such an *almost-ideal* protocol was presented in [7] and an earlier version in [4]. This simple protocol called CASCADE leaks an amount of information close to the theoretical bound, when the error probability is below 15%.

The appropriate part of CASCADE for the error correction is BINARY. When Bob's string contains an odd number of errors, Alice and Bob can perform an interactive binary search on the strings to find and correct one error, respectively. At the beginning Alice sends Bob the parity of the entire string and Bob checks if his parity bit differs from Alice's bit. If not, there is an even number of errors (possibly zero) in the string and nothing is done. If the parity bits differ the following steps are run through to find the error:

1. Alice sends Bob the parity of the first half of the string.
2. Bob determines whether an odd number of errors occurred in the first or in the second half by testing the parity of his string and comparing it to the parity sent by Alice.
3. With the half determined in step 2 we start again at step 1, until the erroneous bit is found.

For a better understanding see the visualization in Fig. 3.6. The leaked information for a string with an even number of errors is one bit and for a string with an odd number $\lceil \log n \rceil + 1$ bits, whereas one error is corrected.

Fig. 3.6 BINARY corrects exact one error of a block with an odd number of errors

CASCADE proceeds in several passes, which are defined by Alice and Bob before execution relative to the error probability p. Alice and Bob hold string $A = A_1, ..., A_n$ and $B = B_1, ..., B_n$ ($A_i, B_i \in \{0, 1\}$), respectively. For the first pass they choose k_1 (determination of parameter k_1 is shown later) and split their strings into blocks of length k_1. Block v for pass 1 is defined by $K_v^1 = \{l|(v-1)k_1 < l \leq vk_1\}$, $v = 1... \left\lceil \frac{n}{k_1} \right\rceil$. On each block BINARY is performed. This can be done parallel for each block to minimize the communication effort. For passes $i > 1$, Alice and Bob choose a k_i and a random function $f_i : [1..n] \rightarrow \left[1.. \left\lceil \frac{n}{k_1} \right\rceil\right]$. Now the block j in pass i has the form $K_j^i = \{l|f_i(l) = j\}$. Again Alice and Bob perform BINARY on these blocks to correct errors. In the previous version of CASCADE [4] the leaked information about the string is eliminated during execution of BINARY by removing the last bit of each subset for which a parity bit is computed. In [7] an improvement is introduced where the removed bits are kept. This allows us to correct more errors in later passes. Because if in pass $i > 1$ an error $B_l \neq A_l$ of block K_j^i is corrected, each block K_v^u from previous passes containing this bit $l \in K_v^u$, $1 \leq u < i$, has now an odd number of errors and can be corrected with less effort. Therefore, a set $\mathcal{K}$ of these blocks is created. The smallest block in $\mathcal{K}$ is chosen and BINARY is executed on it. After correcting the error $B_{l'}$, we create again a new set $\mathcal{B}$ with the previous blocks containing bit $B_{l'}$ from each pass from 1 to i. Now set $\mathcal{K}' = \mathcal{B} \cup \mathcal{K} \backslash \mathcal{B} \cap \mathcal{K}$ is determined. If $\mathcal{K}' \neq \emptyset$ then by choosing again the smallest block further errors can be eliminated. Pass i ends when all blocks K_j^i are checked and when there are no more blocks in $\mathcal{K}'$.

For determining the parameter k_i let $\delta_i(j)$ be the probability that after the pass $i \geq 1, 2j$ errors remain in block K_v^1. Let E_i be the expected number of errors in K_v^1 after pass i. So that after the first pass we have

$$\delta_1(j) = \text{prob}(X = 2j) + \text{prob}(X = 2j + 1),\ (X \approx Bin(k_1, p)),$$

$$E_1 = 2 \sum_{j=1}^{\left\lfloor \frac{k_1}{2} \right\rfloor} j\delta_1(j).$$

Suppose now that k_1 is chosen such that

$$\sum_{l=j+1}^{\left\lfloor \frac{k_1}{2} \right\rfloor} \delta_1(l) \leq \frac{1}{4}\delta_1(j) \tag{3.12}$$

and

$$E_1 \leq -\frac{\ln \frac{1}{2}}{2} \tag{3.13}$$

are satisfied and for the following passes we simply define $k_i = 2k_{i-1}$ for $i > 1$. Brassard and Salvail have shown in [7] that if k_1 satisfies Eqs. 3.12 and 3.13 the amount of information $I(\omega)$ per block of length k_1 leaked after ω passes can be bounded as follows:

$$I(\omega) \leq 2 + \frac{1 - (1 - 2p)^{k_1}}{2} \lceil \log k_1 \rceil + 2 \sum_{l=2}^{\omega} \sum_{j=1}^{\left\lfloor \frac{k_1}{2} \right\rfloor} \frac{j\delta_1(j)}{2^{l-1}} \lceil \log k_1 \rceil. \tag{3.14}$$

For different error probabilities $p \in \{0.01, 0.05, 0.10, 0.15\}$ in four passes the theoretical bound and the leaked information $I(4)$ are computed with Eq. 3.14. In addition empirical tests were performed to find the average amount of leaked information $\hat{I}(4)$ (10 tests with $n = 10,000$). The results of [7] are present in Table 3.3. The column $f(p)\frac{I(4)}{kh(p)}$ gives the percentage of additional leaked information relative to the theoretical bound.

Table 3.3 CASCADE benchmark

p	k_1	$\hat{I}(4)$	$kh(p)$	$I(4)$	$f(p) = \frac{I(4)}{kh(p)}$
0.01	73	6.47	5.89	6.81	1.16
0.05	14	4.60	4.01	4.64	1.16
0.10	7	3.81	3.28	3.99	1.22
0.15	5	3.80	3.05	4.12	1.35

Finally, the reconciliation phase is finished. After getting random strings for Alice and Bob from the sifting phase with length n and error probability p the described reconciliation protocol (CASCADE) has corrected Bob's string and only a very low probability remains that Bob's string has still an error. The leaked information was kept down near the theoretical bound with an additional leakage of 16–35% depending on the error probability. In Chap. 4 a advancement of this protocol is presented, called Adaptive Cascade.

3.2.4 Confirmation/Authentication of Error Correction

The next step in the protocol is on the one hand to confirm the strings in Alice's and Bob's possession and on the other hand to authenticate the reconciliation phase. Fortunately, this can be done in one step based on the fact that Eve could mount a man-in-the-middle attack during the reconciliation phase. Without the authentication step, Alice and Bob could believe that their strings, on which they operate, are identical when they are not. Only one different bit is necessary to produce completely uncorrelated strings at the privacy amplification phase (see Sect. 3.2.5). The best solution for Alice and Bob is to verify that the outputs of the error correction phase are the same. This will authenticate the complete communication during the reconciliation phase, because an intervention of Eve will introduce errors.

The verification can be done by hashing the corrected random strings and compare the resulting tags. We also should keep the size of the tag small compared to the input (corrected shifted key) size. Every additional bit will increase the leaked information and accordingly Eve's knowledge.

As in Sect. 3.2.2 we have to sacrifice a part of already shared key for authentication. Again the best choice to authenticate the corrected shifted key is to use Wegman and Carter authentication as presented in Sect. 2.2.2. Gilbert and Hamrick acknowledge that this is the best choice and analyzed the total authentication costs for the error correction phase [11]. After Eq. 2.35 the authentication cost is $4 \cdot ((b + \log_2 \log_2 a) \cdot \log_2 a)$ for an input size of a and an authentication tag size of b.

The size of the corrected sifted key at the end of the error correction phase is n. The required authentication key length w_1 for the generation of an authentication tag is

$$w_1 = 4 \cdot (g_{EC} + \log_2 \log_2 n) \cdot \log_2 n, \tag{3.15}$$

where g_{EC} is the length of the resulting tag. Alice and Bob compute the tag using the hash function indexed by the authentication key and compare if the tags match. The main problem is if Bob sends this tag to Alice, Eve can modify the tag with arbitrary bits to convince Alice that the strings do not match when, in fact, they do. To avoid this denial-of-service attack Bob must authenticate his message. Therefore, he computes another tag, which authenticates the previous tag as his tag and sends it to Alice. Again he needs a part of the shared key to generate the tag with length g_{auth} which costs

$$w_2 = 4 \cdot (g_{\text{auth}} + \log_2 \log_2 g_{EC}) \cdot \log_2 g_{EC}. \tag{3.16}$$

If Alice determines that her generated tag with cost w_1 matches with Bob's tag and the authentication tag with cost w_2 does as well, she can be sure that the strings are equal. If the tag with cost w_1 does not match but the second does, there must be at least an error in the string of Bob. In the case that both tags do not match, she can assume that either Eve manipulates the messages or the shared keys of Alice and Bob are different.

To indicate Bob that the tags match Alice sends a piece of the key with length

$$w_3 = \tilde{g}_{EC} \tag{3.17}$$

to Bob. Again to avoid the man-in-the-middle attack, Alice must authenticate her message. With the same tag length g_{auth} as Bob's before, yielding an authentication cost of

$$w_4 = 4 \cdot (g_{\text{auth}} + \log_2 \log_2 \tilde{g}_{EC}) \cdot \log_2 \tilde{g}_{EC}. \tag{3.18}$$

After Bob has compared the tags of Alice with his ones they agree the authentication step for the error correction is complete. Alice and Bob can now be sure that with probability of $1 - \varepsilon$ the strings are equal. Because they use an ε-ASU_2 class of hash functions the collision probability for two distinct values is at most ε as the Definition 2.5 for universal hash functions declare. Also the probability for Eve to modify one of this authentication messages is at most ε, even if she suppresses modifications during the error correction phase or make Alice and Bob believe they have different strings when, in fact, they have not.

The required key material to authenticate the reconciliation phase is the sum of all costs w_1, w_2, w_3, w_4 (see Eqs. 3.15, 3.16, 3.17, and 3.18), which results in the total error correction authentication costs

$$\begin{aligned} t_{EC} = {} & 4(g_{EC} + \log_2 \log_2 n) \log_2 n + 4(g_{\text{auth}} + \log_2 \log_2 g_{EC}) \log_2 g_{EC} \\ & + \tilde{g}_{EC} + 4(g_{\text{auth}} + \log_2 \log_2 \tilde{g}_{EC}) \log_2 \tilde{g}_{EC}. \end{aligned} \tag{3.19}$$

3.2.5 Privacy Amplification

The last step to the secret key is the privacy amplification phase. Bennett, Brassard, Crépeau, and Maurer [6] give a good explanation what privacy amplification is:

Privacy amplification is the art of distilling highly secret shared information, perhaps for use as a cryptographic key, from a larger body of shared information that is only partially secret. Let Alice and Bob be given a random variable W, as a random n-bit string, while an eavesdropper Eve learns a correlated random variable V, providing at most $t < n$ bits of information about W, i.e., $H(W|V) \geq n - t$. The details of the distribution P_{VW} are generally unknown to Alice and Bob, except

that it satisfies this constraint as well as possibly some further constraints. They may or may not know P_W. Alice and Bob wish to publicly choose a compression function $g : \{0,1\}^n \to \{0,1\}^r$ such that Eve's partial information on W and her complete information on g gives her arbitrarily little information about $K = g(W)$, except with negligible probability (over possible choices for g). The resulting K is virtually uniformly distributed, given all Eve's information; it can hence be used safely as a cryptographic key.

The size r of the secret that Alice and Bob can distill depends on the kind as well as the amount of information available to Eve. Eve can obtain

- t arbitrary bits of W,
- t arbitrary parity checks of W,
- the result of an arbitrary function mapping n-bit strings to t-bit strings,
- the string W transmitted through a binary symmetric channel with bit error probability p satisfying $h(p) = 1 - \frac{t}{n}$, and hence with capacity $\frac{t}{n}$ where h denotes the binary entropy function.

Assume that we have the scenario where Alice and Bob are connected by an insecure channel to which Eve has passive perfect access. Note that with authentication of the communication active perfect access has the same characteristics, which will not be discussed here for reasons of simplicity. So after Alice's transmission on the quantum channel each of them has knowledge of a correlated random variable. Alice X, Bob Y, and Eve Z. Assume that these variables are distributed according to some joint probability function p_{XYZ}, whereby Eve has partially control over this distribution. Based on the fact that neither Alice nor Bob has an advantage compared to Eve concerning information ($I(X;Y) \not> I(X;Z)$, $I(X;Y) \not> I(Y;Z)$), after the sifting phase which can be denoted as random variable C, Alice can compute a string W from X and C such that Alice's uncertainty about W is 0 and Bob's uncertainty about W is smaller than Eve's one:

$$\begin{aligned} H(W|XC) &= 0, \\ H(W|YC) &< H(W|ZC). \end{aligned} \tag{3.20}$$

The reconciliation phase helps to remove Bob's uncertainty about W; therefore, Alice sends a bit string D with length l is slightly larger than $H(W|YC)$ such that $H(W|YCD) \approx 0$. Eve's uncertainty $H(W|ZCD)$ is lower bounded by $H(W|ZC) - l$, which can be substantially positive, due to Eq. 3.20. We summarize Eve's knowledge ZCD about W to the random variable V.

The aim of privacy amplification is now to generate a secret key K, of which Eve has negligible amount of information. Therefore, Alice and Bob agree on a function g (known by Eve) to generate $K = g(W)$. This g is chosen randomly from a set $\mathcal{G}$ of function to avoid that Eve can decide beforehand which strategy to use. Thus, this function is also a random variable G on the set $\mathcal{G}$. The output length of $G: W \to \{0,1\}^r$ should decrease Eve's information about K to a minimum. Assume $I(W;V) \leq t$ then $I(K;GV) \approx 0$ and $H(K|GV) \approx r$. But how must r be chosen? In

[6] they state $r = n - t - s$ and the security parameter s decreases Eve's information rapidly, because $I(K;GV) \leq \frac{2^{-s}}{\ln 2}$.

The theorem and the corollaries of [6] are presented now, which yields an upper bound to Eve's information. It emerges that for the function set $\mathcal{G}$ the universal classes of hash functions as described in Sect. 2.2.1 become useful, due to their properties.

Theorem 3.2 *Let X be a random variable over the alphabet $\mathcal{X}$ with probability distribution p_X and Rényi entropy $R(X)$, let $Q = G(X)$. Then*

$$H(Q|G) \geq R(Q|G) \geq r - \log_2\left(1 + 2^{r-R(X)}\right) \geq r - \frac{2^{r-R(X)}}{\ln 2},$$

where G is the random variable corresponding to the random choice (with uniform distribution) of a member of a universal class of hash functions $\mathcal{X} \to \{0,1\}^r$.

The first inequality follows from Eq. 2.45. The second inequality can be proven by using Jensen's inequality and using the fact that we use universal hash function. For the complete proof we refer to [6]. The last inequality follows from $\log_2(1+y) \leq \frac{y}{\ln 2}$. The next corollary is derived from the above theorem.

Corollary 3.1 *Let P_{VW} be an arbitrary probability distribution and let v be a particular value of V observed by Eve. If Eve's Rényi entropy $R(W|V = v)$ about W is known to be at least c and Alice and Bob choose $K = G(W)$ as their secret key, then*

$$H(K|G, V = v) \geq r - \log_2\left(1 + 2^{r-c}\right) \geq r - \frac{2^{r-c}}{\ln 2},$$

where G is chosen at random from a universal class of hash functions $W \to \{0,1\}^r$.

Corollary 3.2 *Let W be a random n-bit string with uniform distribution over $\{0,1\}^n$, let $V = e(W)$ for an arbitrary eavesdropping function $e : \{0,1\}^n \to \{0,1\}^t$ for some $t < n$, let $s < n - t$ be a positive safety parameter, and let $r = n - t - s$. If Alice and Bob choose $K = G(W)$ as their secret key, then Eve's expected information about the secret key K, given G and V, satisfies*

$$I(K;GV) \leq \frac{2^{-s}}{\ln 2},$$

where G is chosen at random from a universal class of hash functions $\{0,1\}^n \to \{0,1\}^r$.

Finally, corollary 3.2 can be proven with usage of Corollary 3.1 (see [6]). Note that the security parameter does not depend on the error probability p or else variables, if the upper bound for Eve's knowledge t is known.

Let us recapture the entire protocol to identify the knowledge t Eve has obtained. The easiest knowledge to identify is in the reconciliation phase and the authentication of the reconciliation. We know that during error correction at least $nh(p)$ bits (p is the error probability) are necessary to correct all errors. In Sect. 3.2.3.2 a reconciliation protocol has been presented which reaches nearly this limit. It differs only by a factor $f(p)$ given as in Table 3.3. Thus, the knowledge Eve accumulates during this phase is $nf(p)h(p)$. If we formulate this as a fraction τ_1 by which the reconciled key is to shorten then

$$\tau_1(p) = \frac{nf(p)h(p)}{n} = f(p)h(p) = -f(p)\left(p\log_2 p + (1-p)\log_2(1-p)\right). \tag{3.21}$$

In the confirmation/authentication of the error correction phase a hash value of the key is transmitted. This tag has length g_{EC}, giving Eve additional information. The last phase where Eve can gain knowledge is the sifting phase, if she eavesdrops several photons during quantum transmission. How much information Eve gets, depends on the error probability p. Lütkenhaus derived in [14] and [15] a fraction τ_2, by which we need to shorten the reconciliation key in addition to the other parameters to get a secure key. Therefore, he used a trivial extension of Corollary 3.1:

$$I(K;GV) = H(K) - H(K|GV) = r - H(K|GV) \leq r - R(K|GV). \tag{3.22}$$

This inequality can be applied due to Eq. 2.45. Furthermore, in [14] it is shown that the collision probability of a key k is bounded above by the collision probability of his sifted key for averaged g and v, hence

$$I(K;GV) \leq r - R(K|GV) \leq r - R(W|V). \tag{3.23}$$

We set the information $I(K;GV)$ to zero and thus the shortening is

$$\tau_2 = \frac{n-r}{n} = 1 - \frac{R(W|V)}{n} = 1 + \log_2\langle p_c^w(v)\rangle_v, \tag{3.24}$$

where $\langle\rangle_v$ means averaged with respect to v and $p_c^w(v)$ is the collision probability $\sum_{w\in\mathcal{W}} p(w|y)^2$. We can express the collision probability for the sifted key as the product of the collision probability for single bits of this sifted key $\langle p_c^w(v)\rangle_v = (p_c^{(1)}(p))^n$ with respect to the error rate of the quantum channel p. In [14] this collision probability was derived as

$$p_c^{(1)}(p) \leq \frac{1}{2} + 2p - 2p^2 \qquad \text{for } p \leq 1/2, \tag{3.25}$$

which gives, finally,

$$\tau_2(p) \leq \log_2\left(1 + 4p - 4p^2\right) \qquad \text{for } p \leq 1/2. \tag{3.26}$$

The gain for the privacy amplification g_{pa} is composed of τ_1 from Eq. 3.21, τ_2 from Eq. 3.26, $\frac{s}{n}$ (security parameter length) and $\frac{g_{EC}}{n}$ (authentication tag length)

$$\begin{aligned} g_{pa} &= 1 - \tau_1(p) - \tau_2(p) - \frac{s}{n} - \frac{g_{EC}}{n} \\ &= 1 + f(p)(p \log_2 p + (1-p)\log_2(1-p)) \\ &\quad - \log_2\left(1 + 4p - 4p^2\right) - \frac{s}{n} - \frac{g_{EC}}{n} \qquad \text{for } p \leq 1/2. \end{aligned} \tag{3.27}$$

Note that the two fractions $\frac{s}{n}$ and $\frac{g_{EC}}{n}$ are not crucial factors for the final key, because they are almost fixed values and if the length of the key increases they are comparatively small. Figure 3.7 shows that the maximal acceptable error rate is around 10% for CASCADE and around 11% if we reach Shannon's limit. The factors $\frac{s}{n}$ and $\frac{g_{EC}}{n}$ are not considered.

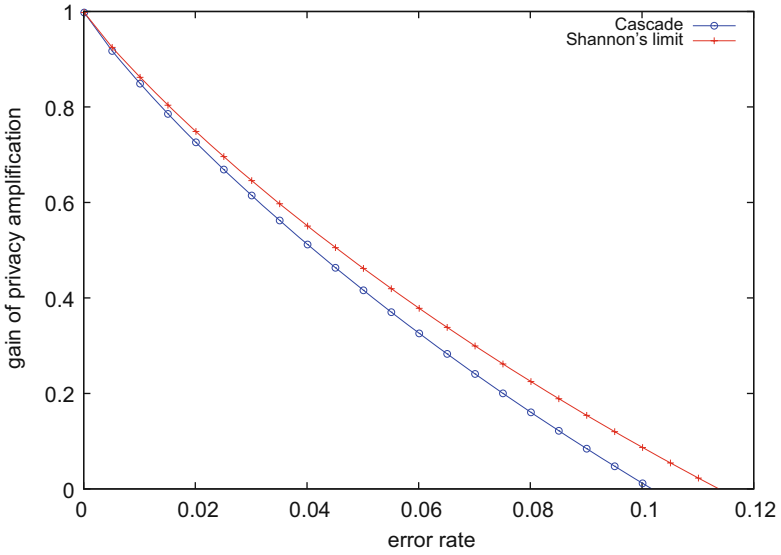

Fig. 3.7 The gain of privacy amplification g_{pa} for increasing quantum error rate p

Concluding the QKD protocol we give a universal class of hash function, which can be used to reduce the reconciliation key of length n to a final key of length k. Therefore, the multiplication in finite fields is defined as

Definition 3.6 Let $\mathcal{A} = GF(2^n)$ and $\mathcal{B} = \{0,1\}^k$. Let $h_c(x)$ be defined as the last k bits of the product $c \cdot x$ in a polynomial representation of $GF(2^n)$. The set

$$\mathcal{H}_{GF(2^n)\to\{0,1\}^k} = \{h_c : c \in GF(2^n)\}$$

is a universal family of hash functions [23].

This family requires only n bits to identify the particular function (the value of c). In particular it is chosen for privacy amplification because the multiplication of large blocks can be done efficiently. We know that by using the traditional shift-and-add algorithm, the multiplication of two elements of $GF(2^n)$ can be achieved in quadratic time. In [2] a simple algorithm is presented, which performs a multiplication in $a \cdot l \log l$ steps for some small constant a.

The value c, which chooses a function of the family of hash functions, must be random. Alice or Bob could randomly choose such a value c and announces it publictly to its peer. But we must consider that this data exchange must be authenticated, not to be exposed to an attack by Eve. Recalling the sifting phase, where Bob has sent Alice an authenticated message, in which he told her, which photons he was able to measure and what bases he has used. A convention was that Bob chooses his bases randomly, so Alice and Bob share a $2n$ authenticated random string (the length is $2n$ because normal sifting has a gain of 0.5). They can now extract the first n bits of this string and use it as c. For this reason the communication during the privacy amplification phase and thus an additional authentication is not necessary.

As an example let the string for the random choice of base, which Bob sent authenticated to Alice, be 0010011101011000101. After performing sifting and reconciliation, the reconciliation key, shared by Alice and Bob, is 1101001110. Due to the error rate and the information leaked during reconciliation we have computed that Eve has 4 bits of information, so we must shorten the key to the length of 6 bits. We extract the value of c from Bob's random choice of bases, which is 0010011101 and multiply c with the reconciliation key.

$$0010011101 \cdot 1101001110 = 100000011011010110$$

The final key consists now of the last six bits of the result (010110).

3.3 QKD Gain

In Sect. 3.1 and 3.2 discussing the communication over the classical and the quantum channel we heard how a QKD system works and some equations are given to compute the output of the several stages. The gain of the quantum channel g_q in Eq. (3.1), the protocol gain g_p in Eq. (3.2), and the gain of privacy amplification g_{pa} in Eq. (3.27) form the complete gain. Additionally we must subtract the required authentication key (t_S (3.6) and t_{EC} (3.19)) which are needed for the next round. So the length k of the final authenticated key for one round is

$$k = m \cdot g_q \cdot g_p \cdot g_{pa} - t_S - t_{EC}, \tag{3.28}$$

where m is the length of the random sequence Alice generates at the beginning and sends via the quantum channel to Bob. The major problem with this protocol is that for the first round we need a pre-shared secret between Alice and Bob. Thus, the key

output length would be limited to the length of the secret. After a certain "warm-up" phase the protocol would reach the expected output on average.

3.4 Finite Resources

All assumptions regarding the postprocessing (sifting, error estimation and correction, privacy amplification) and hence the security of quantum key distribution protocols have been proven in the asymptotic limit. However, the actual implementations of the protocols can only use finite resources, such as limited computational power and keys of finite length. This fact has already been addressed for special protocols like the BB84 protocol [12] or the six-state protocol [16]. In both scenarios it has been shown that the values for the sifted key rate in the finite scenario differ significantly from the expected values from the asymptotic limit.

As described in [20] also the definition of security has to be altered when considering finite resources. In the asymptotic limit a key of length l is said to be secure if its deviation ε from a perfect key tends to zero as l increases. Another property most security definitions lack is composability. Composability assures that a key coming from a quantum cryptographic protocol can safely be used in classical applications (e.g., for encryption with a one-time pad). For the nonasymptotic scenario Scarani and Renner presented a security definition, which satisfies also composability [20]. Here a key is considered to be secure if the difference between the generated key and a perfect key is smaller than ε, which means that ε is the maximum probability that the generated key differs from a perfect key.

The main goal Scarani and Renner pursue in their paper is to obtain a sifted key rate r based on a certain number of signals, a security parameter ε, and some losses from the error correction. As starting point they take the relation from the asymptotic limit, where the sifted key rate r' is

$$r' = S\big(X|E\big) - H\big(X|Y\big), \tag{3.29}$$

where S is the von Neumann and H the Shannon entropy. The goal is achieved using tools from Renner's PhD thesis [18]. The exact formulas and proofs are described in [20] and will not be further discussed here. The major result is that the key rate becomes positive, e.g., for a little more than 10^4 signals and an error rate of 0.5% or at least 10^5 signals and an error rate of 5%.

References

1. Ardehali, M., Chau, H.F., Lo, H.K.: Efficient quantum key distribution (1998). URL http://www.citebase.org/abstract?id=oai:arXiv.org:quant-ph/98% 03007 32
2. Assche, G.V.: Quantum Cryptography and Secret-Key Distillation. Cambridge University Press, New York, USA (2006) 45

3. Bennett, C.H.: Quantum cryptography using any two nonorthogonal states. Phys. Rev. Lett. **68**(21), 3121–3124 (1992). DOI 10.1103/PhysRevLett.68.3121 23
4. Bennett, C.H., Bessette, F., Brassard, G., Salvail, L., Smolin, J.A.: Experimental quantum cryptography. J. Cryptology **5**(1), 3–28 (1992) 36, 37
5. Bennett, C.H., Brassard, G.: Quantum cryptography : Public key distribution and coin tossing. In: Proceedings of the IEEE International Conference on Computers, Systems, and Signal Processing, pp. 175–179 (1984) 23
6. Bennett, C.H., Brassard, G., Crépeau, C., Maurer, U.M.: Generalized privacy amplification. IEEE Trans. Inf. Theory **41**(6), 1915–1923 (1995) 40, 42
7. Brassard, G., Salvail, L.: Secret-key reconciliation by public discussion. In: EUROCRYPT, pp. 410–423 (1993) 34, 35, 36, 37, 38
8. Bruss, D.: Optimal eavesdropping in quantum cryptography with six states. Phys. Rev. Lett **81**(14), 3018–3021 (1998) 23
9. Carter, L., Wegman, M.N.: Universal classes of hash functions. J. Comput. Syst. Sci. **18**(2), 143–154 (1979) 35
10. Ekert, A.K.: Quantum cryptography based on bell's theorem. Phys. Rev. Lett. **67**(6), 661–663 (1991). DOI 10.1103/PhysRevLett.67.661 23
11. Gilbert, G., Hamrick, M.: Practical quantum cryptography: A comprehensive analysis (part one) (2000). URL http://www.citebase.org/abstract?id=oai:arXiv.org:quant-ph/00%09027 30, 39
12. Inamori, H., Lütkenhaus, N., Mayers, D.: Unconditional security of practical quantum key distribution. Eur. Phys. J. D **41**(3), 599–627 (2007) 46
13. Lo, H.K., Chau, H.F., Ardehali, M.: Efficient quantum key distribution scheme and proof of its unconditional security. Journal of Cryptology **18**, 133 (2005). URL http://www.citebase.org/abstract?id=oai:arXiv.org:quant-ph/0011056 26, 28, 29, 32
14. Lütkenhaus, N.: Estimates for practical quantum cryptography. Phys. Rev. A **59**, 3301 (1999). URL http://www.citebase.org/abstract?id=oai:arXiv.org:quant-ph/9806008 43
15. Lütkenhaus, N.: Security against individual attacks for realistic quantum key distribution. Phys. Rev. A **61**(5), 052,304 (2000). DOI 10.1103/PhysRevA.61.052304 43
16. Meyer, T., Kampermann, H., Kleinmann, M., Bru, D.: Finite key analysis for symmetric attacks in quantum key distribution. Phys. Rev. A **74**(4), 042,340 (2006) 46
17. Nielsen, M.A., Chuang, I.L.: Quantum Computation and Quantum Information. Cambridge University Press, Cambridge (2000). URL http://www.amazon.ca/exec/obidos/redirect?tag=citeulike09-20% &path=ASIN/0521635039 24
18. Renner, R.: Security of Quantum Key Distribution. Ph.D. thesis, Swiss Federal Institute of Technology 46
19. Scarani, V., Acin, A., Ribordy, G., Gisin, N.: Quantum cryptography protocols robust against photon number splitting attacks for weak laser pulses implementations. Phy. Rev. Lett. **92**(5), 057,901 (2004) 23
20. Scarani, V., Renner, R.: Quantum cryptography with finite resources: Unconditional security bound for discrete-variable protocols with one-way postprocessing. Phys. Rev. Lett. **100**(20), 200,501 (2008) 46
21. Smith, G., Renes, J.M., Smolin, J.A.: Better codes for BB84 with one-way post-processing (2006). URL http://www.citebase.org/abstract?id=oai:arXiv.org:quant-ph/0607018 32
22. Tang, X., Ma, L., Mink, A., Nakassis, A., Xu, H., Hershmanand J. Bienfang, B., Su, D., Boisvert, R.F., Clark, C., Williams, C.: Quantum key distribution system operating at sifted-key rate over 4 Mbit/s. In: Quantum Information and Computation IV., *Presented at the Society of Photo-Optical Instrumentation Engineers (SPIE) Conference*, Vol. 6244 (2006). DOI 10.1117/12.664455 25
23. Wegman, M.N., Carter, L.: New hash functions and their use in authentication and set equality. J. Comput. Syst. Sci. **22**(3), 265–279 (1981) 44
24. Xu, H., Ma, L., Mink, A., Hershman, B., Tang, X.: 1310-nm quantum key distribution system with up-conversion pump wavelength at 1550 nm. Optics Express **15**, 7247–7260 (2007) 26