Key Distribution Protocols Based on Extractors Under the Condition of Noisy Channels in the Presence of an Active Adversary

Viktor Yakovlev, Valery Korzhik, Guillermo Morales-Luna and Mihail Bakaev\footnote{V. Yakovlev, V. Korzhik and M. Bakaev are with the Department of Information Security of Telecommunication Systems, State University of Telecommunication, St. Petersburg, Russia e-mail: viyak@bk.ru, korzhik1@bk.ru.}\footnote{G. Morales-Luna is with the Computer Science Department, CINVESTAV-IPN, Mexico City, Mexico e-mail: gmorales@cs.cinvestav.mx. Dr. Morales-Luna acknowledges the partial support of Mexican CONACyT.}

May 19, 2010

Abstract

We consider in this paper the information-theoretic secure key distribution problem over main and wire-tap noise channels with a public discussion in presence of an active adversary. In contrast to the solution proposed by ourselves for a similar problem using hashing for privacy amplification, in the current paper we use a technique of extractors.

We propose modified key distribution protocols for which we prove explicit estimates of key rates without the use of estimates with uncertain coefficients in notations $O, \Omega, \Theta$.

This leads in the new conclusion that the use of extractors is superior to the use of hash functions only with the very large key lengths $\ell$ (of order $\ell > 10^5$ bits).

We suggest hybrid key distribution protocols consisting from two consecutively executed stages. At the fist stage it is generated a short authentication key based on hash function, whereas at the second stage it is generated the final key with the use of extractors. We show that in fact the use of extraction procedure is effective only at the second stage. We get also some constructive estimates of the key rates for such protocols.

Keywords. Authentication, cryptography, extractors, information-theoretic security, key distribution, privacy amplification, wire-tap channel.
1 Introduction

Advances in design and implementation of quantum computers [1] as well as design of super-fast multiprocessor conventional computers threat some conceptually secure cryptosystems. Hence perfect one-time pad ciphers proposed by Shannon [2] are necessary. But the use of perfect ciphers requires key lengths proportional to messages [3]. This inconvenience can be solved with the use of key distribution over communication channels protected from eavesdropping. There are several approaches in order to remove (or at least to control) an eavesdropping on the keys:

- quantum channels [4],
- methods based on fluctuation of radio wave channels [5, 6, 7, 8],
- Wyner’s concept of wire-tap channel,
- key generation by hashing of random string initially distributed over noisy channels [9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22].

In the current paper, we follow the last approach. The most advanced results in this setting, under the condition of an active adversary have been obtained by Maurer and Wolf. They proposed several key distribution protocols [11, 12, 13, 14, 15, 16, 17, 18, 19] and made a performance comparison of asymptotic and non-asymptotic key rates for a given level of key security.

We considered in [22] some modification of the Maurer and Wolf’s MW-protocol consisting in using an authentication algorithm over noisy channels, called by ourselves the $\alpha$-protocol, instead of the request-response algorithm presented in [17]. In the same paper [22], we proposed also the $\beta$-protocol that differs from the $\alpha$-protocol in absence of the hash function transmission over public discrete channel because the hash function can be formed from the string which the users have got just after the execution of the initialization phase. Using the $\beta$-protocol entails an increasing of the key rate in several cases. We proposed also in [22] the so called $\alpha'$ and $\beta'$-protocols in which special initially distributed short keys are used in order to provide authentication procedures over public discussion channels (PDC).

Hybrid protocols comprising pairs of sequentially executed protocols $(\alpha, \alpha')$, $(\alpha, \beta')$, $(\beta, \alpha')$, $(\beta, \beta')$ were investigated in [22]. The first protocol in each pair is used to generate an authentication
key, whereas the second one provides a generation of the main secret key for encryption/decryption given the authentication key. The relation among the key rates and a comparison of protocol performance evaluation were also introduced.

The main feature of the protocols considered in [22] is their strict constructiveness because the parameters determining their efficiency do not contain unknown coefficients typical for $O, \Omega, \Theta$-estimations.

Our contribution and novel content in the current paper are the following:

1. We propose some new (modified) key distribution protocols using extractors. We prove explicit estimates of key rates without the use of estimates of uncertain coefficients in $O, \Omega, \Theta$-estimations. (In [22] we solved the similar problem using hash functions instead of extractors).

In contrast to [17], we consider a scenario where the legal users are able to receive raw bit strings over noisy channels and as a consequence they are pairwise distinct. This entails the need to send check symbols from user A to user B in order to agree the raw bit strings received by legal users. By the same reason, we have changed the authentication algorithm: instead of a request-response algorithm [17], we use a non-interactive one based on the authentication code.

A consideration of the non-asymptotic case leads us in the new conclusion that the use of extractors is superior to the use of hash functions only for very large key lengths ($\ell$) of the order of $10^5$ bits.

2. We suggest hybrid key distribution protocols consisting of two consecutively executed stages.

At the first stage, a short authentication key based on a hash function is generated, whereas at the second stage, the final key using extractors is generated. We show that in fact the use of an extraction procedure is effective only at the second stage. We get also explicit estimates of key rates for such protocols.

3. We prove also an asymptotic behavior of the key rates for all considered protocols that allows to compare the potential efficiency of them with the potential efficiency of protocols considered here and in [22].
The outline of this paper is the following: Section 2 contains the preliminaries and descriptions of the main procedures to be used in key distribution protocols. In Section 2.1 we describe the model of key distribution based on noisy wire-tap channels in the presence of an active adversary and we introduce the main criteria for key distribution protocol efficiency. We introduce main procedures as error correction, authentication and privacy amplification (based both on hashing and extraction). In Section 3 we describe the $\alpha_{\text{ext}}$-protocol, and the new key distribution $\beta_{\text{ext}}$-protocol without transmission of the extractor’s seed on the public discussion channel and we prove their main features. In section 4 we present a modification of the previous $\alpha'_{\text{ext}}$ and $\beta'_{\text{ext}}$-protocols under the condition that initially the legal users share short authentication keys. In section 5 we describe the so called hybrid protocols as combinations of different pairs of single protocols and we estimate their performance evaluation. In section 6 we conclude the paper.

2 Main notions and procedures involved in the key distribution protocol

Here, we repeat mostly the content of the same point as in [22]. It is done in order to provide an independent reading of the current paper.

2.1 Model for key distribution and the main criteria for protocol efficiency

Let us consider the model of key distribution between a legal user, Alice (A), and another user, Bob (B), in the presence of an active adversary, Eve (E), assuming that initially the legal users do not have shared secret keys. The key distribution protocol (KDP) consists of two phases: initialization and key generation.

In the KDP initialization phase, A, B, and E receive random i.i.d. sequences $X = \{x_i\}_{i=1}^k$, $Y = \{y_i\}_{i=1}^k$, $Z = \{z_i\}_{i=1}^k \in \{0,1\}^k$, respectively, such that for each $i$, $\Pr(x_i \neq y_i) = p_m$ and $\min\{\Pr(x_i \neq z_i), \Pr(y_i \neq z_i)\} = p_w$ (see Figure 1). One of the methods to provide legal users A, B with the sequences $X$, $Y$ is to generate the truly random sequence $S = \{s_i\}_{i=1}^k \in \{0,1\}^k$ by some trusted party, say source S, and then to transmit it to the legal users A and B over noisy channels (as in the source model [9], [12]). We will assume that A and B receive the sequences $X$, $Y$.
over binary symmetric channels (BSC) without memory with error probabilities $\pi_A = \Pr(x_i \neq s_i)$, $\pi_B = \Pr(y_i \neq s_i)$, while the adversary $E$ receives the sequence $Z$ over a BSC with error probability $\pi_E = \Pr(z_i \neq s_i)$. It is easy to see that if the original sequence $S$ is truly random then the same property holds for the sequences $X$, $Y$ and $Z$. (Examples of practical implementation of the initialization phase in real world can be found in [22]). In this phase it is natural to assume that the adversary is unable to intervene the transmission from $S$ to $A$ and $B$.

The key generation phase consists in an information exchange over a public discussion channel (PDC) with a goal to share eventually the final key. We note that the use of PDC is necessary in order to send check symbols to test the agreement of the strings $X$ and $Y$ and sometimes for the parameters of the hash function or extractor seed transmission (see details in the following sections). The adversary $E$ can receive all information transmitted over the PDC. We assume also that the PDC’s between legal users and $E$ are binary noiseless channels (if $E$ does not intervene in transmission). However $E$ can change or replace this information as desired and therefore it is necessary to authenticate messages transmitted over a PDC in order to detect any intervention of $E$ and to reject suspicious messages.

Let us define the following parameters of the key distribution protocol characterization:

$\ell$: the key length (the number of bits which are contained in the keys $K_A$ and $K_B$),

$I(K_A, U)$: the amount of Shannon’s information in possession of the adversary $E$ about the final key $K_A$ after receiving all acceptable information $U$, including the sequence $Z$ and the
other messages transmitted over the PDC

\[ P_e = \Pr (K_A \neq K_B) : \] the probability of legal users keys disagreement,

\[ P_f : \] the probability of false rejection of the KDP protocol (when A or B falsely believe that E has intervened the PDC),

\[ P_d : \] the probability of deception false information provided by E during information transmission over PDC (it can result in an opportunity to fix a key between any legal user and E although leaving the legal user on the belief that he (she) has shared a key with his (her) legal partner),

\[ R_k : \] the key distribution rate (the ratio of the key length \( \ell \) to the length of sequences \( X, Y \)),

\[ R_k = \frac{\ell}{k}. \]

It is reasonable to impose the following conditions on the KDP:

\[ \ell = \ell^{req}, \quad (1) \]
\[ I(K_A, U) \leq I^{adm}, \quad (2) \]
\[ P_e \leq P_e^{adm}, \quad (3) \]
\[ P_f \leq P_f^{adm}, \quad (4) \]
\[ P_d \leq P_d^{adm}, \quad (5) \]

where \( \ell^{req} \) denotes the required key length and the superscript \( adm \) stands for admissible parameter value. We will say that the above conditions are requirements of the KDP. The efficiency of the KDP will be estimated by the key rate \( R_k \) and then among all protocols satisfying (1)-(5), we will select the most efficient by making \( R_k \) to attain its largest value. As we will show later, some inequalities (2)-(5) may randomly hold. Then an additional requirement can be stated as

\[ P_{risk} \leq P_{risk}^{adm}, \quad (6) \]

where \( P_{risk}^{adm} \) is the probability that at least one of the inequalities (2)-(5) does not hold.
2.2 Known asymptotic results regarding key rates

Let us denote by \( R^* \), \( R^{**} \) the maximum achievable key rates in a KDP between the legal users under the condition of a passive or active adversary, respectively. In the papers \[12\], \[14\], \[17\], \[19\] the proofs of these values were presented. For the source model of the wire-tap channel with initialization phase in the KDP using BSC with probabilities \( \pi_A \), \( \pi_B \), \( \pi_E \) the following theorem holds:

**Theorem 1 (see [19])** If \( \pi_E > \pi_A \) and \( \pi_E > \pi_B \), then \( R^* = R^{**} \). If either \( \pi_E \leq \pi_A \) or \( \pi_E \leq \pi_B \), then \( R^{**} = 0 \).

We note that under the conditions \( \pi_E > \pi_A \) and \( \pi_E > \pi_B \), the users A and B either share the key or they may detect interception in the case of E’s intervention. This fact cannot be interpreted as a defect of KDP because E can use even a simple strategy: she tries to break off the PDC between legal users in order to impede the completion of the KDP. Let \( p_m \), \( p^A_w \), \( p^B_w \), denote the probabilities of disagreements among the sequence pairs \((X,Y)\), \((X,Z)\), \((Y,Z)\) respectively. Then

\[
\begin{align*}
  p_m &= \pi_A + \pi_B - 2\pi_A\pi_B \\
  p^A_w &= \pi_A + \pi_E - 2\pi_A\pi_E \\
  p^B_w &= \pi_B + \pi_E - 2\pi_B\pi_E
\end{align*}
\]

It is easy to see that if \( \pi_E > \pi_A \) then \( p^A_w > p_m \) and similarly if \( \pi_E > \pi_B \) then \( p^B_w > p_m \). We will consider the worst case for legal users as \( p_w = \min\{p^A_w, p^B_w\} \).

After the execution of the initialization phase the source model is reduced to the channel model where user A sends the sequence \( X^k \) to user B who receives it as \( Y^k \), whereas E receives \( X^k \) as \( Z^k \). Then the probability of error on the main virtual BSC between A and B is \( p_m \) and the probability of the wire-tap virtual channel from A to E is \( p_w \). (The PDC remains the same after such reduction of the source model to the channel model.)

**Theorem 2 (see [19])** In the channel model setup with probabilities \( p_m \), \( p_w \) the maximum key distribution rate is

\[
R^* = g(p_w) - g(p_m),
\]
where \( p \mapsto g(p) = -p \log p - (1 - p) \log(1 - p) \) is the entropy function.

2.3 Error correcting codes

Let \( C \) be a binary linear error-correcting \((k + r, r)\)-code and let \( C^r \) be a string consisting of \( r \) check symbols. It has been proved in [23] that if the information symbols are transmitted on the BSC with the error probability \( p_m \), whereas the check symbols on the noiseless channel, then the average error probability of decoding on the ensemble of all \((k + r, r)\)-codes meets the following modified Gallager’s bound

\[
P_e \leq 2^{-k E(R_c)},
\]

where

\[
E(R_c) = \max_{\rho \in (0,1)} \left[ E_0(\rho) - \frac{\rho(2R_c - 1)}{R_c} \right],
\]

\[
E_0(\rho) = \rho - (1 + \rho) \log_2 \left( \frac{\frac{1}{1 + \rho} (1 - p_m)}{1 + \rho (1 - p_m)} \right)
\]

is Gallager’s function for a BSC with the error probability \( p_m \)

\[
R_c = \frac{k}{k + r}
\]

is the code rate. We note that in the frame of the above model, the code rate \( R_c \) satisfies the inequality

\[
0 \leq \frac{2R_c - 1}{R_c} \leq C^*,
\]

where \( C^* = 1 - g(p_m) \) is the capacity of the BSC with the probability of error \( p_m \).

It follows from [12] that \( \frac{1}{2} \leq R_c \leq \frac{1}{1 + g(p_m)} \). In the asymptotic case \( R_c \to \frac{1}{1 + g(p_m)} \), then

\[
r = k g(p_m).
\]

We see from [13] that an arbitrary small value of the erroneous decoding probability is achieved for large block length if the number \( r \) of check symbols (but not block length) is proportional to the number of information symbols \( k \) with coefficient \( g(p_m) \).
2.4 Authentication based on the class of universal hash functions

In order to execute the authentication procedure, we use the universal hash function which are described below.

For any finite set $A$, let $|A|$ denote its cardinality. For any two finite sets $A, B$, let $H$ be a set of hash-maps $A \to B$. For each $x_0, x_1 \in A$, let $\delta_H(x_0, x_1) = |\{h \in H \mid h(x_0) = h(x_1)\}|$ be the number of hash functions in $H$ that collide in $x_0$ and $x_1$. We recall that $H$ is universal $2$, $U_2$ in short, if for each $x_0, x_1 \in A$, $\delta_H(x_0, x_1) \leq \frac{|H|}{|B|}$.

Let $P_{\text{col}}$ be the so called collision probability, namely the probability that there occurs a pair of elements in $A$ colliding under an uniformly chosen map $h \in H$. Clearly, $P_{\text{col}} \leq \frac{|B| - 1}{|B|}$. The class $H$ is strongly universal $2$, $SU_2$, if

$$\forall x \in A, \forall b \in B : \ |\{h \in H \mid h(x) = y\}| = \frac{|H|}{|B|}$$

and besides for any distinct $x_0, x_1 \in A$, and any $y_0, y_1 \in B$,

$$|\{h \in H \mid h(x_0) = y_0 \& h(x_1) = y_1\}| \leq \frac{|H|}{|B|}.$$

For a given $\epsilon > 0$, the class $H$ is $\epsilon$-almost universal, $\epsilon$-$AU_2$, if for all $x_0, x_1 \in A$: $\delta_H(x_0, x_1) \leq \epsilon |H|$. The class $H$ is $\epsilon$-almost strongly universal, $\epsilon$-$ASU_2$, if (14) holds and for any pairs $x_0, x_1$ and $y_0, y_1$ of distinct points in $A$ and $B$,

$$|\{h \in H \mid h(x_0) = y_0 \& h(x_1) = y_1\}| \leq \epsilon \frac{|H|}{|B|^2}.$$

Naturally, each class $|B|^{-1}$-$ASU_2$ is also $SU_2$.

Examples of hash functions classes: We assume that the sets $A$ and $B$ consist of all binary sequences of lengths $a$ and $b$, respectively: $A = \{0, 1\}^a$, $B = \{0, 1\}^b$, hence $|A| = 2^a$, $|B| = 2^b$.

An $U_2$ class. The set $A$ can be identified with the Galois field $GF(2^a)$. For each $s \in GF(2^a)$, let $h_s : A \to B, x \mapsto [xs]_b$, where the map $z \mapsto [z]_b$ takes the $b$ least significant bits in $z$. The class $\{h_s\}_{s \in A}$ is $U_2$. Such hash functions are described uniquely by binary strings of length $a$. 

9
An \( SU_2 \) class. For each \( s, t \in GF(2^a) \), let \( h_{st} : A \rightarrow A, x \mapsto sx + t \). The class \( \{h_{st}\}_{s,t \in A} \) is \( SU_2 \) and clearly this class can be indexed by sequences of length \( 2a \).

An \( \epsilon \)-ASU\(_2\) class. It has been shown in [24] that the hash functions chosen from an \( \epsilon \)-ASU\(_2\) class are connected with incomplete balanced schemes. The parameters of the \( \epsilon \)-ASU\(_2\) class can be described as

\[
|A| = q^{2i}, \quad |B| = q, \quad |H| = q^{i+2}, \quad \epsilon = \frac{i + 1}{q},
\]

where \( q \) is a power of a prime and \( i > 1 \) is an integer.

Let us analyze the procedure of message authentication. Let \( x \) be the message to be authenticated during its transmission from user A to user B. User A forms the authenticator \( y = h(x) \) of his message \( x \) using the keyed hash function \( h \in H \) known by him (but unknown for adversary the E), then A appends \( y \) to \( x \) and sends the pair \((x, y)\) to the legal user B. In order to check the authenticity of the message \( x \), the user B receives a pair \((\tilde{x}, \tilde{y})\) (which may be forged), B forms the authenticator \( \tilde{y} = h(\tilde{x}) \) with his knowledge of the secret hash function \( h \) and compares \( \tilde{y} \) with \( y \). If they coincide then B accepts \( x \), otherwise he rejects it.

It was shown in [24] that if the hash functions, chosen from the \( \epsilon \)-ASU\(_2\) class, are used in the authentication procedure then for the best adversary’s strategy consisting in an impersonation or substitution of the messages, the following probability bounds hold

\[
P_i \leq |B|^{-1},
\]

\[
P_s \leq \epsilon,
\]

where \( P_i \) is the probability of message impersonation, and \( P_s \) is the probability of message substitution.

Let us define the probability of undetected false message deception by the adversary as \( P = \max\{P_i, P_s\} \). The bounds (16), (17) will hold only if the active adversary ignores completely the used hash function \( h \) in the authentication procedure. But there may be situations when the keyed hash function is partly known by the adversary although authentication procedure is still possible.
In order to clarify this situation let us recall initially from [14] that for a discrete random variable \( \xi \) taking values over a set \( X \) with probability distribution \( P_\xi \) the minimal entropy is

\[
H_\infty(\xi) = -\log \max_{x \in X} P_\xi(x),
\]

and the Renyi entropy of the random variable \( \xi \) is

\[
H_2(\xi) = -\log \sum_{x \in X} P_\xi^2(x). \tag{18}
\]

**Theorem 3 (see [25])** Suppose legal users A and B have the random key \( h \) with length \( \ell_0 \) within an authentication scheme based on \( \epsilon \)-ASU\(_2\) hash functions where \( \epsilon = 2^{-b} \). Denote by \( U \) the total knowledge of E about \( h \). Then, assuming that for any sample \( u \)

\[
H_\infty(h|U=u) \geq t\ell_0, \quad 0 < t < 1, \tag{19}
\]

the probability \( P_d \) of message undetected deception is upper bounded as

\[
P_d \leq 2^{-\left(\frac{b-\ell_0(1-t)}{2} - 1\right)}.\]

### 2.5 Authentication based on noisy channels

The message authentication considered above and based on the use of hash functions from either the class \( SU_2 \) or the class \( \epsilon \)-ASU\(_2\) requires a possession by legal users of the secret or partly secret keys. However such keys cannot be taken directly from the strings \( X^k, Y^k \) shared in the initialization phase because they differ even for legal users. On the other hand it is impossible to concciliate these string by sending from A to B the check symbols strings of \( X^k \) because PDC is get not authenticated and B could “concciliate” formerly the false string \( Z^k \) with E.

In order to avoid this situation it is necessary firstly to design a keyless message authentication based on noisy channels. In [12] a special type of codes has been proposed in order to solve this problem: the so called authentication codes (AC). Let us describe them briefly.

In an initialization phase the users share the strings \( X^k, Y^k \) over a BSC \((\Pr(x_i \neq y_i) = p_m)\) and
they agree an error correcting binary systematic \((n_a,k_a)\)-code \(V\) in order to authenticate a length \(k_a\) message. The authenticator \(w = (w_1,\ldots,w_{n_a})\) of a message \(m\) is formed as follows: for each \(i\) let \(v_i\) be the \(i\)-th bit of the codeword in \(V\) corresponding to \(m\) and \(w_i = v_i\) if \(v_i = 1\), or let it remain undefined otherwise.

After receiving a pair \((\tilde{m},\tilde{w})\), the user B forms his authenticator \(\tilde{\tilde{w}}\) for the message \(\tilde{m}\) using his string \(Y^k\) according to the agreed procedure and compares \(\tilde{w}\) with \(\tilde{\tilde{w}}\). If the number of the coinciding bits in them is less or equal to some given threshold \(\Delta_w\) then the message \(\tilde{m}\) succeeds as authentic, otherwise it is removed as forged. The AC’s were investigated in [12] and can be characterized by two probabilities:

- \(P_f\), the probability of false removal of the message although adversary E does not intervene at all;
- \(P_d\), the probability of the deception of false message, i.e. the probability of the event that E has forged a message and this fact was not detected by B.

\(P_f\) and \(P_d\) do not depend on ordinary minimum code distance of the code \(V\) but on the so called \emph{minimum asymmetric semidistance} \(d_{01}\) that is determined by the minimal number of differences between 0 and 1 symbols in any pair of distinct code words of \(V\).

\textbf{Theorem 4 (see [22, 26])} Let \(V\) be an \((n_a,k_a)\)-AC with constant Hamming weight \(\tau\) for all non-zero codewords and with asymmetric semidistance \(d_{01}\). Then the probabilities \(P_f\) and \(P_d\) for the authentication procedure on noisy wire-tap channel with parameters \(p_m\) and \(p_w\), can be upper bounded as follows:

\[
P_f \leq \sum_{i=\Delta_w+1}^{\tau} \binom{\tau}{i} p_m^i (1-p_m)^{\tau-i},
\]

\[
P_d \leq \sum_{i=0}^{\Delta_w} \binom{d_{01}}{i} p_m^i (1-p_m)^{d_{01}-i} \sum_{j=0}^{\Delta_w-1} \binom{\tau-d_{01}}{j} p_m^j (1-p_m)^{\tau-d_{01}-j}.
\]

It is a very hard problem to find \(d_{01}\) for any linear code. But there exists a very simple method to design the code \(V\) with known \(d_{01}\), given the linear \((n_0,k_0)\)-code \(\tilde{V}\) with known ordinary minimum code distance \(d\) proposed in [12].
Namely, let us substitute the symbol 1 with the symbol pair 10 and the symbol 0 with 01 in \( \tilde{V} \). Then evidently the parameters of the code \( V \) are:

\[
    n_a = 2n_0 \quad , \quad k_a = k_0 \quad , \quad d_{01} = d \quad , \quad \tau = n_0.
\]  

(20)

We have proved in [22] the following theorem with the use of the above code.

**Theorem 5** Let \( V \) be a \((k_0 + r_0, k_0)\)-error correction code with minimum distance \( d \) that is used in the authentication procedure. Then for any \( p, q > 0 \) there exists an integer \( k'_0 \) and an AC, guaranteeing \( \frac{r_0}{k_0} < q, P_f \leq p, P_d \leq p \) for all \( k_0 > k'_0 \).

It follows from this theorem that

\[
    \frac{r_0}{k_0 + r_0} \to 0 \text{ as } k_0 \to +\infty.
\]

This means that the length of the authenticator approaches zero as the block length tends to infinity. Other methods to design constant weight AC were investigated in [27].

### 2.6 Extractors

Let us recall the notion of *extractor* and *strong extractor* [28, 29, 30]. Two probability distributions \( P, Q \), defined on the same set \( X \), are called \( \epsilon \)-close if their statistical difference

\[
    \text{dif}(P, Q) = \frac{1}{2} \sum_{x \in X} |P_X(x) - Q_X(x)|
\]

does not exceed \( \epsilon \). A map \( E : \{0, 1\}^k \times \{0, 1\}^u \to \{0, 1\}^\ell \) is an \((\eta, \epsilon)\)-extractor if for any probability distribution random variable \( X \) on \( \{0, 1\}^k \) such that \( H_\infty(X) \geq \eta \) and any uniformly distributed random variable \( \Gamma \) on \( \{0, 1\}^u \), the statistical difference probability distribution of the extractor output \( E(X, \Gamma) \) with respect to an uniform distribution on \( \{0, 1\}^\ell \) is at most \( \epsilon \). In order words, the extractor maps a random sequence \( X \) of length \( k \) with symbols taken from an ensemble of minimal entropy \( H_\infty(X) \) to a random sequence of length \( \ell \) that is \( \epsilon \)-close to an uniformly distributed sequence with the help of a truly random sequence \( \Gamma \) of length \( u \). The last sequence can be seen as a “seed” of the extractor. The extractor \( E(X, \Gamma) \) has parameters \((k, \eta, u, \ell, \epsilon)\), where \( k \) is the length of
input random sequence, \( \eta \) is the evaluation of minimal entropy \( H_\infty(X) \geq \eta \) on the set of input sequences, \( u \) is the length of the seed \( \Gamma \), \( \ell \) is the length of the output sequence, and \( \epsilon \) is the statistical distance between the output probability distribution and a uniform distribution on the output set.

A mapping \( E : \{0,1\}^k \times \{0,1\}^u \rightarrow \{0,1\}^\ell \) is called a strong extractor \( E(X,\Gamma) \) if for any probability distribution random variable \( X \) on the set \( \{0,1\}^k \) having minimal entropy \( H_\infty(X) \geq \eta \) and for any uniformly distributed random variable \( \Gamma \) on the set \( \{0,1\}^u \) the probability distribution of the concatenated variables \( \Gamma \circ E(X,\Gamma) \) is close to a uniform distribution on \( \{0,1\}^{\ell+u} \). More specifically

\[
\text{dif}\left( \Gamma \circ E(X,\Gamma), U^{\ell+u} \right) \leq \epsilon.
\]

This means that the strong extractor provides the closeness of probability distribution for the concatenation of the output extractor sequence and the seed sequence to a uniform distribution. In the current paper, we will consider only extractors based on the construction \[28, 29\] which is an improvement of the originally proposed by Trevisan \[30\].

**Theorem 6 (see theorem 22 in \[29\])** For every \( k, H_\infty(X), \ell \in \mathbb{N} \) and \( \epsilon > 0 \), such that \( \ell \leq H_\infty(X) \leq k \), there are explicit strong \( (H_\infty(X), \epsilon) \)-extractors \( E : \{0,1\}^k \times \{0,1\}^u \rightarrow \{0,1\}^{\ell-\Delta} \) with

\[
u = O\left( \frac{\log_2 \left( \frac{k}{\ell} \right)}{\log_2 \left( \frac{H_\infty(X)}{\ell} \right)} \right), \tag{21}
\]

or

\[
u = O\left( \log_2 \left( \frac{k}{\ell} \right) \cdot \log_2 \left( \frac{1}{\mu} \right) \right), \tag{22}
\]

where \( 1 + \mu = \frac{k}{\ell-1} \), \( \mu < \frac{1}{2} \) and \( \Delta = O(d) \). The value \( \Delta \) is the loss of extractor output sequence length.

The first extractor \( \text{(21)} \), with \( \frac{H_\infty(X)}{\ell} \) constant, is used for extraction of an arbitrary part of randomness \( (H_\infty(X^k)) \) from the input sequence \( X^k \), whereas the second one \( \text{(22)} \) is needed in order to extract all randomness \( \ell = H_\infty(X^k) \) from the input sequence \( X^k \).

We are not going to use the estimates based on the \( O \)-operator and therefore let us find a more accurate estimate for the length of the seed. For this reason we consider in greater detail the design
In order to design the Trevisan’s extractor it is necessary to realize three components:

1. The linear error code $W$: With parameters $(\tilde{n}, k)$ and minimal code distance $d_w$, where $\tilde{n} = 2^\nu$, $\nu \in \mathbb{N}$. It is proposed to take this code as a concatenation of the Reed-Solomon and the Adamar codes.

2. Combinatorial block design scheme. (Balance incomplete block design, BIBD). This is a family of sets $S = \{S_1, S_2, \ldots, S_\ell\}$ holding the following properties:

   $$S_i \subseteq \{1, 2, \ldots, u\},$$
   $$|S_i| = \nu,$$
   $$i \neq j \implies |S_i \cap S_j| < \log c \quad \text{with } c \geq 1. \quad (23)$$

   This means that the family consists of $\ell$ sets or blocks, each consisting of $\nu$ elements taken from the set of integers $\{1, 2, \ldots, u\}$, while the number of elements contained simultaneously in any pair of blocks is at most $\log c$. Such construction is designated as a $(\nu, c)$-scheme.

3. Boolean function $f$: This map is defined over $\{0, 1\}^\nu$ and for each $a_1, \ldots, a_\nu \in \{0, 1\}$, $f(a_1, \ldots, a_\nu)$ is a codeword of the $(\tilde{n}, k)$-code $W$.

The design of the extractor based on the three components given above is presented in Figure 2.

The algorithm is executed in the following stages:
1. The input sequence $\overline{x}$ is encoded as $\overline{w}$ with the error correcting $(\tilde{n}, k)$-code $W$. The word $\overline{w}$ gives the value of the Boolean function $f : \{0, 1\}^\nu \to \{0, 1\}^\ell$.

2. The random sequence $\gamma$ of length $u$ determines the samples $\gamma|_{S_i}$, consisting of $\nu$ symbols of $\gamma$ with the use of blocks $S_i$ belonging to the $(\nu, c)$-BIBD. This means $\gamma|_{S_i} = [\gamma]_{s \in S_i}$.

3. Output $w = f \left( (\gamma|_{S_i})_{i=1}^\ell \right)$ as the result of the extractor.

In the modified extractor version at [29], it was proposed to use a $(\nu, c)$-weak scheme, in which the condition (23) is changed by the condition

$$\sum_{j<i} 2^{|S_j \cap S_i|} \leq c(\ell - 1),$$

where $c$ is some constant, $c > 1$. The length $\tilde{n}$ of the code $W$ is chosen in [29], p. 106, according to the condition

$$\log(\tilde{n}) = O(\log \frac{k}{\epsilon}).$$

Since $w$ is the output of a Boolean function with $\nu$ arguments, $\tilde{n}$ should be equal to $2^\nu$. Obviously this condition will be fulfilled if

$$\nu = \left\lceil \log \frac{k}{\epsilon} \right\rceil,$$

where $\lceil x \rceil$ is the “ceiling” of $x$ (the least integer greater or equal than $x$).

The characterization of strong extractor is determined by the following statements.

**Theorem 7 (Proposition 10 in [29])** If $S = (S_1, \ldots, S_\ell)$ (with $S_i \subset \gamma$) is a weak $(\nu, c)$-design for

$$c = \frac{1}{\ell} \left[ H_\infty(X^k) - 3 \log \frac{\ell}{\epsilon} - u - 3 \right],$$

then $E : \{0, 1\}^k \times \{0, 1\}^u \to \{0, 1\}^\ell$ is a strong $(H_\infty(X^k), \epsilon)$-extractor.

**Theorem 8 (Lemma 15 in [29])** For every $\nu, \ell \in \mathbb{N}$ and $c > \ell$, there exists a weak $(\nu, c)$-design $S = (S_1, \ldots, S_\ell)$ (with $S_i \subset \gamma$) and

$$u = \left\lceil \frac{\nu}{\ln c} \right\rceil \cdot \nu.$$

Moreover, such a family can be found in polynomial time $\text{poly}(\ell, u)$. 

16
Theorem 9 (Lemma 17 in [29]) For every \( \nu, \ell \in \mathbb{N} \) and \( 0 < \mu < \frac{1}{2} \), there exists a weak \((\nu, 1+\mu)\)-design \( S = (S_1, \ldots, S_\ell) \) (with \( S_i \subset \gamma \)) with \( u = O\left(\nu^2 \cdot \log \frac{1}{\mu}\right) \). Moreover, these families can be found in polynomial time \( \text{poly}(\ell, u) \).

The results of Theorem 6 will be avoided in our further investigation because a presentation of the output sequence length \( u \) in the form \( \ell - \Delta \) is inconvenient in the optimization procedure.

We will get an estimate of \( u \) taken from the results of Theorems 7-9 directly. More specifically, using (24) and (26) one can write the relation for the necessary number of seed symbols for the first extractor (21) in Theorem 6:

\[
u = \left\lceil \frac{\log k}{\ln c} \right\rceil \cdot \left\lceil \log \frac{k}{\epsilon} \right\rceil . \tag{27}
\]

For the second extractor (22), it follows from the proof of lemma 17 in [29] that \( u_0 = \left\lceil \frac{\mu}{\ln 2} \right\rceil \cdot \nu \), where \( u = \tau \cdot u_0 \), \( \tau = \left\lceil \log \frac{4}{\mu} \right\rceil \), \( 0 < \mu < \frac{1}{2} \). Then in terms of (26) we get

\[
u = \left\lceil \frac{\log k}{\ln 2} \right\rceil \cdot \left\lceil \log \frac{k}{\epsilon} \right\rceil \cdot \left\lceil \log \frac{4}{\mu} \right\rceil .
\]

2.7 Privacy amplification

The procedure of privacy amplification (PA) at the final stage of the key generation between users A and B has been investigated in detail in [10, 17]. PA can be implemented either by hashing [10, 17] or by extraction [28]. We will consider in the current paper the second approach.

In order to compare our new results with the results obtained in [22] where hashing has been used as the privacy amplification procedure, let us specify an application of both methods.

The sequence \( X_k \) of length \( k \) bits is mapped by the user A to the sequence \( X^\ell_A \) of length \( \ell \) through a keyless hash function from the class \( U_2 \), or \( SU_2 \). In a similar manner the user B forms his key \( X^\ell_B \) after error-correcting his sequence \( Y^k \).

The most important parameter of the PA procedure is the residual Shannon’s information received by the adversary \( E \), concerning the final key \( K_A = K_B \). The estimates of this information leaking of \( E \) are given in [10, 17] for different settings of wire-tap channels. The more general bound is presented below.
Theorem 10 (see [10]) Let $X^k$ be the sequence of length $k$ transmitted from $A$ to $B$ over a BSC with the error probability $p_m$ and received by $B$ as the sequence $Y^k$. Assume also that $Z^k$ is the result of receiving $X^k$ by the adversary $E$ over a BSC with probability $p_w$, while the Renyi information contained in $Z^k$ about $X^k$ is $t$. Let $H$ be a $U_2$-class of hash functions from $\{0,1\}^k$ into $\{0,1\}^\ell$, known by all participants ($A$, $B$ and $E$) and let $h \in H$ be a truly random hash function chosen by $A$, transmitted to $B$ over a PDC. If the users $A$ and $B$ compute their keys as $K_A = h(X^k)$, $K_B = h(\tilde{Y}^k)$, then the amount of Shannon’s information about the keys $K_A, K_B$ is upper bounded as

$$I(K_A; Z^k, h) \leq \frac{2^{-(k-\ell-t)}}{\ln 2}.$$  \hspace{1cm} (28)

The Renyi information $t$ is connected with the Renyi entropy $H_2(X^k|Z^k)$ as

$$t = k - H_2(X^k|Z^k).$$  \hspace{1cm} (29)

For the BSC used as wire-tap channel we have by (18)

$$H_2(X^k|Z^k) = -k \log (p_w^2 + (1-p_w)^2).$$

If the adversary receives some extra information about $X^k$ aside the information contained in $Z^k$ (for instance the sequence $C^r$ of check symbols of length $r$ eavesdropped by $E$ over the PDC), then the amount of conditional Renyi entropy $H_2(X^k|Z^k, h, C^r)$ that received $E$ can be estimated by the theorem [11] appearing below. (This theorem also gives the estimation of the amount of conditional minimal entropy $H_\infty(X^k|Z^k, h, C^r)$ which we use later).

Theorem 11 (see [17]) Let $X$ and $C$ be two random variables and let $s > 0$. Then the following inequalities hold:

$$H_2(X) - H_2(X|C = c) \leq \log |C| + s$$  \hspace{1cm} (30)

with a probability at least $1 - 2^{-\frac{s}{2} - 1}$,

$$H_\infty(X^k) - H_\infty(X^k|C = c) \leq \log |C| + s$$  \hspace{1cm} (31)

with a probability at least $1 - 2^{-s}$.
We can apply (30) in order to estimate $H_2(X^k|Z^k,h,C^r)$. Then we get

$$H_2(X^k|Z^k,h,C^r) \geq H_2(X^k|Z^k,h) - r - s. \quad (32)$$

It follows from (29) and (32) that

$$\tilde{t} = k - H_2(X^k|Z^k,h,C^r)$$

$$\leq k - H_2(X^k|Z^k,h) + r + s$$

$$= t + r + s. \quad (33)$$

Substituting $\tilde{t}$ from (33) into (28) instead of $t$, we get the upper bound of Shannon’s information leaking to $E$

$$I(K_A;Z^k,h,C^r) \leq 2^{-(k-\ell-t-r-s)} \ln 2,$$

that holds with a probability $P_{risk} \leq 1 - 2^{s-1}$. In order to compare the performance of privacy amplification based on hashing and on extraction, let us prove a new lemma establishing a connection between the Shannon’s information leaking to adversary regarding the key at the output of the extractor, and the statistical difference among distribution of the final key and an uniform distribution.

**Lemma 1** If the statistical distance between the output of the extractor generating the length $\ell$ key and an uniform distribution is at most $\epsilon$, then the amount of the Shannon’s information concerning the key got by any adversary is upperly bounded as

$$I(K_A;Z^k|\Gamma^u) \leq 2\ell\sqrt{\epsilon}.$$

**Proof.** The following inequality holds by definition of strong extractor

$$dif(\Gamma^u \circ \text{Ext}(Z^k,\Gamma^u),V^{\ell+u}) \leq \epsilon. \quad (34)$$

The meaning of this inequality is that the probability distribution of the concatenation of the extractor’s output and the “seed” $\Gamma^u$ is close enough to an uniform distribution. In order to
simplify the notation, let us denote by $D$ the term at the left of (34). We note initially that for an uniform distribution on the space $V^{\ell+u}$, the following equality holds

$$D = M_{\Gamma^u}[\text{dif}(\text{Ext}(Z^k, \gamma^u), V^\ell)].$$

(35)

where $\gamma^u$ is a random sequence and $M_{\Gamma^u}[\cdot]$ is the expectation with respect to the distribution on $\Gamma^u$. In fact, $D$ itself can also be expressed as the term:

$$\frac{1}{2} \sum_{\gamma^u, e^\ell \in E} \left| P_{E|\Gamma^u}(e^\ell) P_{\Gamma^u}(\gamma^u) - P_{V^\ell}(v^\ell) P_{V^u}(v^u) \right|, \quad (36)$$

where $e^\ell$ is the output sequence of the extractor $E$.

Since the distributions $P_{\Gamma^u}(\gamma^u)$ and $P_{V^u}(v^u)$ are both uniform, the term (36) gives:

$$D = \frac{1}{2} \sum_{\gamma^u} P_{\Gamma^u}(\gamma^u) \sum_{e^\ell \in E} \left| P_{E|\Gamma^u}(e^\ell) - P_{V^\ell}(v^\ell) \right|$$

$$= \frac{1}{2} \sum_{\gamma^u} P_{\Gamma^u}(\gamma^u) \text{dif}(\text{Ext}(Z^k, \Gamma^u), V^\ell)$$

$$= M_{\Gamma^u}[\text{dif}(\text{Ext}(Z^k, \gamma^u), V^\ell)]$$

proving (35). By combining (34) and (35) it is obtained

$$M_{\Gamma^u}[\text{dif}(\text{Ext}(Z^k, \gamma^u), V^\ell)] \leq \epsilon. \quad (37)$$

Using the well known Markov’s inequality, (37) implies

$$P_{\Gamma^u} \left[ \text{dif}(\text{Ext}(Z^k, \gamma^u), V^\ell) \leq \rho \epsilon \right] \geq 1 - \frac{1}{\rho},$$

where $\rho \geq 1$ is some arbitrary value.

In the lemma 6 at [17], the following inequality has been proved which put in our own notation states

$$H \left( \text{Ext}(Z^k, \gamma^u) \right) \geq \ell \left( 1 - \text{dif}(\text{Ext}(Z^k, \gamma^u), V^\ell) - 2^{-\ell} \right). \quad (38)$$
Taking into account that the output extractor sequence is just the key, we can write

\[ H \left( \text{Ext}(Z^k, \gamma^u) \right) = H \left( K^\ell | \Gamma^u = \gamma^u \right). \] (39)

Then by substituting (39) into (38), we have that the inequality

\[ H \left( K^\ell | \Gamma^u = \gamma^u \right) \geq \ell \left( 1 - \text{dif}(\text{Ext}(Z^k, \gamma^u), V^\ell) - 2^{-\ell} \right) \] (40)

will hold with probability

\[ P_{\Gamma^u} \text{[inequality (40) holds]} \geq 1 - \frac{1}{\rho}. \] (41)

It follows from (41) a trivial estimate for the averaged value \( H \left( K^\ell | \Gamma^u \right) \) over \( \gamma^u \), namely

\[ H \left( K^\ell | \Gamma^u \right) \geq \left( 1 - \frac{1}{\rho} \right) \ell \left( 1 - \rho \varepsilon - 2^{-\ell} \right). \] (42)

After a simplification on the right side of (42) and by neglecting smaller values than \( 2^{-\ell} \)

\[ H \left( K^\ell | \Gamma^u \right) \geq \ell \left( 1 - \rho \varepsilon - \frac{1}{\rho} + \varepsilon \right). \]

Then for the amount of information leaking of an adversary concerning the key \( K^\ell \), given the knowledge of \( \Gamma^u \), the following bound is obtained

\[ I(K^\ell; Z^k | \Gamma^u) \leq \frac{\ell}{\rho} + \ell \varepsilon \rho \left( \rho - 1 \right). \] (43)

The right side of (43) is minimized under the condition \( \rho = \frac{1}{\sqrt{\varepsilon}} \) giving the final inequality

\[ I(K^\ell; Z^k | \Gamma^u) \leq 2\ell \sqrt{\varepsilon} \]

providing thus the desired result. \( \square \)

It follows from the above lemma that if the value of the statistical difference at the extractor output that forms the length \( \ell \) key does not exceed \( \varepsilon \), then the amount of the residual information regarding the key obtained by the adversary does not exceed \( 2\ell \sqrt{\varepsilon} \).
This means that a requirement, regarding the amount of Shannon’s information on the key leaking to an adversary, of the form \( I(K^\ell; Z^k|\Gamma^u) \leq I^{adm} \) will be fulfilled if \( 2\ell\sqrt{\epsilon} = I^{adm} \). This fact results in the following requirement to the extractor’s statistical distance:

\[
\epsilon = \left( \frac{I^{adm}}{2\ell} \right)^2.
\]  

(44)

3 Key distribution protocols

3.1 Statement of the protocols

Two key distribution protocols in presence of an active adversary have been proposed by Maurer and Wolf in [17]: the **UH-protocol**, in which privacy amplification procedure was executed using hash functions and the **EX-protocol** based on extractions. It has been shown in [17] that the EX-protocol majors the UH-protocol with respect to several conditions.

We want to investigate a performance of these and other new protocols. We will show that our new protocols are superior than those considered in [17] for non-asymptotic cases (e.g. when the sequence lengths are finite).

Initially we consider modified UH- and EX-protocols and denote them as \( \alpha \) and \( \alpha_{ext} \), respectively. A difference between the original and the modified protocols is determined by two factors.

1. We consider protocols under the condition \( \pi_A \neq 0, \pi_B \neq 0, \pi_A, \pi_B < \pi_E \), or equivalently the conditions \( p_m > 0, p_w > p_m \), see Figure [1]. This requires to send the check symbols from A to B in order to conciliate \( X^k \) and \( Y^k \).

2. Instead of the authentication algorithm “request-response” presented in [17], we will use non-interactive the AC-based algorithm (see Section 2) because this allows the users to provide authentication even when the sequences \( X^k \) and \( Y^k \) do not coincide completely. By the same reason, the authentication algorithm and the number of substrings of the original strings \( X^k \) and \( Y^k \) are changed.

Before the execution of the \( \alpha, \alpha_{ext} \)-protocols, the users A and B divide their respective sequences \( X^k, Y^k \), into \( X_1^{k_1}, X_2^{k_2} \) and \( Y_1^{k_1}, Y_2^{k_2} \) of lengths \( k_1, k_2 \). (The first parts \( X_1^{k_1} \) and \( Y_1^{k_1} \) will be used
for key generation in the execution of the PA procedure while the second parts $X_2^{k_2}$ and $Y_2^{k_2}$ will be used in the execution of the authentication procedure). Since the $\alpha$-protocol was already considered in [22], we move on to the $\alpha_{ext}$-protocol [31].

1. The user A forms the string $C_1^{r_1}$ of check symbols of length $r_1$ to the string $X_1^{k_1}$ using a $(k_1 + r_1, k_1)$-error correcting code $C_1$. (This code should be agreed by users in advance.)

2. The user A generates a truly random binary sequence $\gamma$ (which will be used as an extractor seed) of length $u$.

3. The user A forms the authenticator $w$ for the message $(C_1^{r_1}, \gamma)$ using for that an AC based on an error correcting $(n_0, k_0 = r_1 + u, d)$-code and the sequence $X_2^{k_2}$.

4. The user A sends to B the message $(C_1^{r_1}, \gamma)$ over a PDC appended with the authenticator $w$.

5. The user B verifies the authenticity of the message $(C_1^{r_1}, \gamma)$ through the known $(n_0, k_0)$-AC and his string $Y_2^{k_2}$ (see section 2.5). If authenticity is confirmed, then B goes to the next step. Otherwise he rejects the KDP.

6. The user B corrects the error in string $Y_1^{k_1}$ through the check symbols string $C_1^{r_1}$. We denote by $\tilde{Y}_1^{k_1}$ the string $Y_1^{k_1}$ after error correction.

7. In order to get the keys $K_A$ and $K_B$ both users A and B execute a privacy amplification procedure based on an extractor (see section 2.6): $K_A = E_{ext}(X_1, \gamma), K_B = E_{ext}(\tilde{Y}_1, \gamma)$.

Recall that the $\alpha$-protocol differs from the $\alpha_{ext}$-protocol in that it generates a hash function $h$ in step 2. This hash function jointly with the check symbols of $C_1^{r_1}$ and the authenticator $w$ are transmitted to B (steps 3-5). In the seventh step, this hash function is needed for key generation: $K_A = h(X_1), K_B = h(\tilde{Y}_1)$.

It has also been proposed in [22] a new $\beta$-protocol that differs from the $\alpha$-protocol in the following: After the execution of the initialization phase, both users A and B have got the strings that can in fact be used to form the hash functions. In this way, we do not require to send the hash functions over the PDC, hence the length $k_2$ used before for authentication of the hash function can be shortened. Therefore we may expect that the length of the substring $X_1$ is increased (if the
total length of the string $X$ is fixed). But such conclusion is not so apparent because we have to extract the hash function as a segment from the string $X$.

A similar problem appears in the case in which an extractor is used instead of a hash function for privacy amplification. In the $\alpha_{\text{ext}}$-protocol $A$ generates a truly random sequence $\gamma$ and sends it to $B$ jointly with the authenticator of $\gamma$. But the required sequence $\gamma$ can be gotten directly by both users $A$ and $B$ from the initially distributed strings $X$ and $Y$. This results in the following $\beta_{\text{ext}}$-protocol. It is worth to note that although $\gamma$ is not uniformly distributed from the adversary’s point of view this has no relevance for strong extractors.

Within the above setup, the users $A$ and $B$ divide the strings $X^k$, $Y^k$ into three disjoint parts $X_1^k$, $X_2^k$, $X_3^k$ and $Y_1^k$, $Y_2^k$, $Y_3^k$ with $k_1 + k_2 + k_3 = k$. Then they execute the following steps:

1. The user $A$ forms the length $r_1$ string $C_1^{r_1}$ of check symbols of the string $X_1^{k_1}$ using the error correcting $(k_1 + r_1, k_1)$-code $C_1$, agreed in advance.

2. The user $A$ forms the length $r_2$ check string $C_2^{r_2}$ of the string $X_3^{k_3}$ using the error correcting $(k_3 + r_2, k_3)$-code $C_2$, agreed in advance.

3. The user $A$ forms the authenticator $w$ of the message $(C_1^{r_1}, C_2^{r_2})$ using an AC and his substring $X_2^{k_2}$.

4. The user $A$ sends to $B$ the message $(C_1^{r_1}, C_2^{r_2})$ over a PDC appended with $w$.

5. The user $B$ verifies the authenticity of the message $(C_1^{r_1}, C_2^{r_2})$ using a AC and his substring $Y_2^{k_2}$. If it is confirmed then he goes to the next step. Otherwise he rejects the KDP.

6. The user $B$ corrects errors on strings $Y_1^{k_1}$, $Y_3^{k_3}$, using the check strings $C_1^{r_1}$ and $C_2^{r_2}$. Denote by $\tilde{Y}_1^{k_1}$, $\tilde{Y}_3^{k_3}$, the strings $Y_1^{k_1}$, $Y_3^{k_3}$, after error corrections.

7. The users $A$ and $B$ take their substrings $X_3^{k_3}$, $\tilde{Y}_3^{k_3}$, where $k_3 = u$, as the second argument $\gamma^u$ in their extractors.

8. Both users $A$ and $B$ form the keys as $K_A = E_{\text{ext}}(X_1, X_3)$, $K_B = E_{\text{ext}}(\tilde{Y}_1, \tilde{Y}_3)$.
3.2 Performance evaluation of the protocols

A theorem has been proved in [22] determining the optimal parameters for both the $\alpha$, and $\beta$-protocols depending on the posed requirements. Let us prove a generalization of that theorem for the $\alpha$, $\beta$, $\alpha_{ext}$, and $\beta_{ext}$-protocols. We will assume that for the $\alpha$, and $\beta$-protocols a hashing is used as privacy amplification procedure, whereas for $\alpha_{ext}$, and $\beta_{ext}$-protocols an extraction is used. Moreover we assume that the first extraction scheme considered in section [2.6] is used, where the number of random bits $u$ is determined by equation (27).

**Theorem 12** Let us assume that the users $A$, $B$ and the adversary $E$ have binary strings $X^k$, $Y^k$ and $Z^k$, respectively after execution of the initialization phase over the wire-tape channel, $p_w = \Pr(x_i \neq y_i)$, $p_w = \min\{\Pr(x_i \neq z_i), \Pr(y_i \neq z_i)\}$, $p_m \geq 0$, $p_w > p_m$. Then $A$ and $B$ are able to form a common key of length $\ell$ satisfying the requirements (2)-(6) after the execution of any of the $\alpha$, $\beta$, $\alpha_{ext}$, and $\beta_{ext}$-protocols if the parts of lengths $k_1, k_2$ on which were divided the substrings $X^k$, $Y^k$ for the $\alpha$, and $\alpha_{ext}$-protocols or the parts of lengths $k_1, k_2, k_3$ on which were divided the substrings $X^k$, $Y^k$ for the $\beta$, and $\beta_{ext}$-protocols satisfy the equations listed below:

- for all protocols
  \[ k_1 = - \frac{\log P_{adm}^c}{E(R_{c1})}, \quad (45) \]

- for $\alpha$ and $\beta$-protocols
  \[ k_1 = \frac{\ell + r_1 - 2 \log P_{adm}^c - \log(I_{adm} \ln 2) - 2}{- \log(p_w^2 + (1 - p_w)^2)}, \quad (46) \]

- for $\alpha_{ext}$, and $\beta_{ext}$-protocols
  \[ k_1 \cdot H_\infty = \ell c + r_1 - \log P_{risk}^c + u + 3 \log \ell \left( \frac{I_{adm}}{2 \ell} \right)^{-2} + 3, \quad (47) \]
where \( R_{c_1} \) and \( E(R_{c_1}) \) are determined by (8)-(11) and

\[
\begin{align*}
u = & \left\lfloor \log k_1 \left( \frac{t_{adm}}{2\ell} \right)^{-2} \right\rfloor \cdot \log k_1 \left( \frac{t_{adm}}{2\ell} \right)^{-2} \end{align*}
\]

(48)
is the number of the extractor random symbols, \( \ell \) is a parameter under optimization, \( H_\infty = -\log \max(p_\ell, 1 - p_\ell) \),

\[
\begin{align*}
k_2 \left( 1 - g \left( \frac{2d}{k_2} \right) \right) &= 2k_0, \quad \sum_{i=\Delta_w+1}^{k_2} \binom{k_2}{i} p_m^i (1 - p_m)^{k_2 - i} = P_{adm}^f, \\
\sum_{i=0}^{\Delta_w} \binom{d}{i} p_w^i (1 - p_w)^{d - i} &= P_{adm}^d,
\end{align*}
\]

(49) (50)

and \( r_2 \) being the number of check symbols of the error correcting \((k_3 + r_2, k_3)\)-code \( C_2 \) found similarly as in equation (45),

\[
k_0 = \begin{cases} 
  k_1 + r_1 & \text{for the } \alpha\text{-protocol}, \\
  2r_1 & \text{for the } \beta\text{-protocol}, \\
  u + r_1 & \text{for the } \alpha_{ext}\text{-protocol}, \\
  r_1 + r_2 & \text{for the } \beta_{ext}\text{-protocol},
\end{cases}
\]

(52)

where \( \Delta_w \) is the number of the extractor random symbols, \( \ell \) is a parameter under optimization, \( H_\infty = -\log \max(p_\ell, 1 - p_\ell) \),

\[
\begin{align*}
k_2 \left( 1 - g \left( \frac{2d}{k_2} \right) \right) &= 2k_0, \\
\sum_{i=\Delta_w+1}^{k_2} \binom{k_2}{i} p_m^i (1 - p_m)^{k_2 - i} &= P_{adm}^f, \\
\sum_{i=0}^{\Delta_w} \binom{d}{i} p_w^i (1 - p_w)^{d - i} &= P_{adm}^d,
\end{align*}
\]

(49) (50)

where \( k_0 = \begin{cases} 
  k_1 + r_1 & \text{for the } \alpha\text{-protocol}, \\
  2r_1 & \text{for the } \beta\text{-protocol}, \\
  u + r_1 & \text{for the } \alpha_{ext}\text{-protocol}, \\
  r_1 + r_2 & \text{for the } \beta_{ext}\text{-protocol},
\end{cases} \)

(52)

and \( r_2 \) being the number of check symbols of the error correcting \((k_3 + r_2, k_3)\)-code \( C_2 \) found similarly as in equation (45),

\[
k_3 = \begin{cases} 
  0 & \text{for the } \alpha\text{-protocol}, \\
  k_1 & \text{for the } \beta\text{-protocol}, \\
  0 & \text{for the } \alpha_{ext}\text{-protocol}, \\
  u & \text{for the } \beta_{ext}\text{-protocol}.
\end{cases}
\]

(53)

The key rate is then determined as follows:

\[
R_\alpha = \frac{\ell}{k_1 + k_3}, \quad R_\beta = \frac{\ell}{2k_1 + k_3}, \quad R_{\alpha_{ext}} = \max_c \frac{\ell}{k_1 + k_2}, \quad R_{\beta_{ext}} = \max_c \frac{\ell}{u + k_1 + k_2}.
\]

(54)
Proof. For the \( \alpha \) and \( \beta \)-protocols the theorem has been proved in [22]. Let us prove it only for the \( \alpha_{\text{ext}} \) and \( \beta_{\text{ext}} \)-protocols.

Let the bounds of the KDP parameters meet exactly all requirements (2)-(6), e.g. the following equation hold:

\[
P_e = P_{e}^{\text{adm}} = 2^{-k_1 E(R_{c1})}
\]

where \( R_{c1} = \frac{k_1}{k_1+r_1} \) is the code rate, and \( E(R_{c1}) \) is computed by (8)-(10). Under the condition that the adversary gets the sequence \( Z^{k_1} \) over a BSC with error probability \( p_w \) the conditional minimal entropy is

\[
H_\infty(X^{k_1}|Z^{k_1}) = k_1 H_\infty(X|Z) = -k_1 \log \max(p_w, 1-p_w) = k_1 H_\infty.
\]

Since the adversary receives also the check block \( C_{11}^{r_1} \), in line with (31) the following inequality results:

\[
H_\infty(X^{k_1}|Z^{k_1}, C_{11}^{r_1}) \geq k_1 H_\infty - r_1 - s,
\]

which does not comply with the probability \( P_{\text{risk}} \leq 2^{-s} \).

By substituting (55) into (25), we may write \( \epsilon \leq 2^\tau \) where \( \tau = \ell c + 3 \log \ell - k_1 H_\infty + r_1 + s + u + 3 \). Let us assume that \( I^{\text{adm}} \) is chosen in such a way that

\[
2^\frac{\tau}{3} = \log \left( \frac{I^{\text{adm}}}{2\ell} \right)^2,
\]

resulting thus condition (44). Hence we can write

\[
\ell c + 3 \log \ell - k_1 H_\infty + r_1 + s + u + 3 = 3 \log \left( \frac{I^{\text{adm}}}{2\ell} \right)^2.
\]

Assuming \( P_{\text{risk}} = P_{\text{adm risk}} = 2^{-s} \), (56) holds eventually from (47). The value \( u \) in (56) is the number of the extractor random symbols. In order to find it, we can use (27) substituting \( \epsilon \) by \( \left( \frac{I^{\text{adm}}}{2\ell} \right)^2 \) in line with Lemma 1 that results in (48). A solution of the equation system (45) and (47) allows to
find the parameters $k_1, r_1$, given a fixed $c$. It will be shown in the sequel that the key rate can be maximized by a proper selection of the parameter $c$.

In order to find $k_2$ let us assume that the probabilities $P_f$ and $P_d$ have equal values, $P_f = P^{adm}_f$, $P_d = P^{adm}_d$, with

$$P^{adm}_f = \sum_{i=\Delta_w+1}^{2n_0} \binom{2n_0}{i} p^i_m (1 - p_m)^{2n_0 - i},$$

$$P^{adm}_d = \sum_{i=0}^{\Delta_w} \binom{d}{i} p^i_w (1 - p_w)^{d - i}.$$  \hfill (57)

$$\sum_{j=0}^{\Delta_w - i} \binom{2n_0 - d}{j} p^j_m (1 - p_m)^{2n_0 - d - j},$$

\hfill (58)

where $n_0, k_0, d$ are the parameters of error correcting codes used in the AC.

Recall that for the AC we had $k_2 = 2n_0$ where $n_0$ is the length of the error correcting $(n_0, k_0)$-code with minimum distance $d$. For the $\alpha_{ext}$-protocol $k_0 = r_1 + u$, while for the $\beta_{ext}$-protocol $k_0 = r_1 + r_2$, where $r_2$ is the number of check symbols in the $(k_3 + r_2, k_3)$-code $C_2$. This gives relation (52) for the parameter $k_0$.

Using the Varshamov-Gilbert inequality [32] connecting $n_0, k_0, d$ and taking into account that $k_2 = 2n_0$ we get

$$k_2 \left( 1 - g \left( \frac{2d}{k_2} \right) \right) = 2k_0.$$ \hfill (59)

Solving the equation system (57)-(59), equivalent to the equation system (49)-(51), we find the parameters $k_2, d$. The value $r_2$ is calculated by (8)–(10), in which it is necessary to let $k = u r = r_2$, $p = p_m$, $P_e = P^{adm}_e$.

In line with the above protocols, we have that for the $\alpha_{ext}$-protocol, $k_3 = 0$ and for the $\beta_{ext}$-protocol, $k_3 = u$. This fact proves (53). Then relation (54) is apparent from the protocols description.

**Remark 1** If the solution of the system (45)–(53) is not unique then it is reasonable to select any of them maximizing the key rate.

**Remark 2** It is worth to note that the values $k_1, k_2, k_3, c$ found for the same requirements $P^{adm}_e$, $P^{adm}_m$, $P^{adm}_{risk}$, $P^{adm}_f$, $P^{adm}_d$, but for different protocols, may be different.
Remark 3 (Choice of $c$ in (47)-(48)) In figures 3 and 4 the dependence of the key rate for the $\alpha_{ext}$-protocol and the $\beta_{ext}$-protocol is plotted versus the parameter $c$, given fixed values $\ell$ for different error probabilities in the main channels.

We assume that $p_w = 0.2$, $I_{adm} = 10^{-30}$, $P_e^{adm} = P_{risk}^{adm} = P_f^{adm} = P_d^{adm} = 10^{-5}$ in the plotting of these curves. From these curves it is patent that the key rate depends essentially on the choice of the parameter $c$.

Let us compare the $\alpha$, $\beta$, $\alpha_{ext}$, $\beta_{ext}$-protocols on the key rates.

Theorem 13 If the key length $\ell$ is given and the rate $R_c$ of the error correction code satisfies $R_c \geq \frac{2}{3}$, then $R_\beta \geq R_\alpha$, and $R_{\beta_{ext}} \geq R_{\alpha_{ext}}$.

Proof. The first inequality is proved as theorem 10 in [22]. Let us prove the second inequality. Let us write $R_{\alpha_{ext}} = \frac{\ell}{k_1 + k_2}$ and $R_{\beta_{ext}} = \frac{\ell'}{k_1' + u + k_2'}$. For a fixed common length, $\ell = \ell'$, we should prove $k_1' + u + k_2' \leq k_1 + k_2$. 

29
Figure 4: Dependence of key rate versus the parameter extractor \( c \) for the \( \beta_{ext} \)-protocol.

Under the requirements (1), (2), (6) posed to the KDP parameters, \( P_{risk} \), \( I_{adm} \), it follows from (47) that \( \ell' = \ell \) whenever \( k_1 = k'_1 \) and \( r_1 = r'_1 \). Therefore it is necessary just to prove

\[
k_2 \geq u + k'_2. \tag{60}
\]

According to the scheme of the AC code design we can write \( k_2 = 2n_0 = 2(r_1 + u + r_0) \), where \( r_0 \) is the number of check symbols of the \( (n_0, k_0) \)-code. \( k'_2 = 2n'_0 = 2(r_1 + r'_2 + r'_0) \), where \( r'_2 \) is the number of check symbols of the \( (u + r'_2, r'_2) \)-code \( C_2 \), and \( r'_0 \) is the number of check symbols of the \( (n'_0, k'_0) \)-code.

By substituting the expressions for \( k_2 \), \( k'_2 \) presented above into (60), we get the equivalent inequality \( u - 2r'_2 + 2r_0 - 2r'_0 \geq 0 \). In order to prove this inequality it is sufficient to show that \( u \geq 2r'_2 \) and \( r_0 \geq r'_0 \).

The first inequality holds because under the theorem’s condition, \( R_c \geq \frac{2}{3} \) for the \( (u + r'_2, r'_2) \)-code. In order to prove the second inequality, we note that \( r_0 \) is the number of check symbols of the
information block of length $k_0 = u + r_1$ and $r'_0$ is the number of check symbols of the information block of length $k'_0 = r_1 + r'_2$. It is clear that $k_0 \geq k'_0$ and it is followed from Varshamov-Gilbert inequality that $\frac{d}{n_0}$ maintains a constant value as the information block length increases in order to get the required error correction capability. Therefore $r_0 \geq r'_0$ and this completes the proof of the theorem.

With the purpose of comparing the protocols performance with hashing and with extraction, let us find the relation of key rate for sufficiently large $\ell$.

**Theorem 14** As the key length $\ell \to \infty$, then the following relations hold

\[
R_\alpha = \frac{H_2(p_w) - g(p_m)}{3 + 2g(p_m)},
\]

\[
R_\beta = \frac{H_2(p_w) - g(p_m)}{2 + 4g(p_m)},
\]

\[
R_{\alpha_{ext}} = R_{\beta_{ext}} = \frac{H_\infty(p_w) - g(p_m)}{1 + 2g(p_m)}.
\]

Proof. The proofs of (61) and (62) were presented in [22]. In order to prove (63), let us write the relations of the key rates at the $\alpha_{ext}$ and $\beta_{ext}$-protocols taking into account (20), (52), (53), (54):

\[
R_{\alpha_{ext}} = \frac{\ell}{k_1 + 2n_0} = \frac{\ell}{\frac{k_1 + u + 2r_1 + 2r_0}{\ell}}.
\]

\[
R_{\beta_{ext}} = \frac{\ell}{k_1 + u + 2n_0} = \frac{\ell}{\frac{k_1 + u + 2r_1 + 2r_0}{\ell}}.
\]
where \( u \) is the length of the extractor seed, \( r_1 \) is the length of the check string for the \((k_1 + r_1, k_1)\)-code, \( r_2 \) is the length of the check symbols string for the \((u + r_2, u)\)-code, and \( r_0 \) is the length of the check string for the AC-code.

According to (13), \( r = kg(p_m) \) for sufficiently large \( \ell \) (and hence sufficiently large \( k \)). Let us rewrite (64), (65) as

\[
R_{\alpha \text{ext}} = \frac{\ell}{k_1(1 + 2g(p_m)) + 2u + 2r_0}, \tag{66}
\]

\[
R_{\beta \text{ext}} = \frac{\ell}{(k_1 + u)(1 + 2g(p_m)) + 2r_0}. \tag{67}
\]

According with (47),

\[
k_1 = \frac{\ell c - 2 \log P_{\text{risk}} + u + 3 \log \ell \left( \frac{r_{\text{adm}} \ell}{2\ell} \right)^{-2} + 3}{H_\infty - g(p_m)}. \tag{68}
\]

Substituting \( k_1 \) into (66) produces

\[
R_{\alpha \text{ext}} = \frac{H_\infty - g(p_m)}{c(1 + 2g(p_m)) + 2(H_\infty - g(p_m)) \left( \frac{u}{\ell} + \frac{r_0}{\ell} \right)}. \tag{69}
\]

It is easy to show that

\[
\lim_{\ell \to \infty} \frac{u}{\ell} = 0.
\]

Also theorem 5 establishes that \( \lim_{\ell \to \infty} \frac{r_0}{k_0 + r_0} = 0 \), but since \( k_0 \to \infty \) as long as \( \ell \to \infty \), according to (47) and (52) we get \( \lim_{\ell \to \infty} \frac{r_0}{\ell} = 0 \). Now we can write (69), in the limit \( R_{\alpha \text{ext}} = \frac{H_\infty(p_w) - g(p_m)}{c(1 + 2g(p_m))} \), which approaches to a maximum as \( c \to 1 \). This provides a proof of (63) for the \( \alpha_{\text{ext}} \)-protocol.

Similarly, by expressing \( R_{\beta \text{ext}} \) as (67) using \( k_1 \) as in (68), the used arguments in the proof of (63) for the \( \alpha_{\text{ext}} \)-protocol, show that (63) holds also for the \( \beta_{\text{ext}} \)-protocol. \( \square \)

The following trivial corollary results from the above theorem.

**Corollary 1** If the channel parameters \( p_m \) and \( p_w \) are such that

\[
\frac{H_\infty(p_w) - g(p_m)}{1 + 2g(p_m)} \geq \frac{H_2(p_w) - g(p_m)}{3 + 2g(p_m)} \tag{70}
\]
and
\[ 2(H_\infty(p_w) - g(p_m)) \geq H_2(p_w) - g(p_m), \] (71)

then \( R_{\alpha_{ext}} \geq R_\alpha \) and \( R_{\beta_{ext}} \geq R_\beta \) respectively for sufficiently large \( \ell \).

**Corollary 2** If \( p_m = 0 \), then \( R_{\alpha_{ext}} \geq R_\alpha \) and \( R_{\beta_{ext}} \geq R_\beta \).

**Proof.** If \( p_m = 0 \), then the relations (70), (71) can be written as
\[ H_\infty(p_w) \geq \frac{1}{3} H_2(p_w) \quad \text{and} \quad H_\infty(p_w) \geq \frac{1}{2} H_2(p_w). \]

Since \( 2H_\infty(p_w) \geq H_2(p_w) \) \[\Box\], then \( R_{\alpha_{ext}} \geq R_\alpha \) and \( R_{\beta_{ext}} \geq R_\beta \).

Let us exemplify the above results and illustrate that the \( R_{\alpha_{ext}}(\ell) \) and \( R_{\beta_{ext}}(\ell) \)-protocols major the \( R_\alpha(\ell) \) and \( R_\beta(\ell) \)-protocols respectively. Let us select the following natural requirements for the KDP:
\[ I_{adm} = 10^{-30} , \quad P_d = P_e = P_f = P_{risk} = 10^{-5}. \] (72)

In figure 5 we plot the key rates \( R_k \) versus its length \( \ell \) for both \( R_{\alpha_{ext}}(\ell) \) and \( R_{\beta_{ext}}(\ell) \)-protocols with \( p_m = 0.01 \) and \( 0.001, \quad p_w = 0.2 \), and the requirements presented in (72).

The optimization of \( c \) has been performed for every value of \( \ell \). For comparison purposes the dependences \( R_\alpha(\ell) \) and \( R_\beta(\ell) \) are shown also in the figure.

The following conclusions are drawn immediately after an examination of the obtained dependence.

The protocols using extractors have greater key rate than the \( \alpha \) and \( \beta \)-protocols under sufficiently large \( \ell \) and small \( p_m \). It is worth to note that if for the \( \alpha \) and \( \beta \)-protocols the asymptotically possible value key rate calculated by (61), (62) can be achieved even in the considered key length range, it is not true for the \( \alpha_{ext} \) and \( \beta_{ext} \)-protocols, demonstrating a noticeable increasing proliferation of the key rate outside this range.

We can see that the \( \alpha_{ext} \)-protocol is superior than the \( \alpha \)-protocol when \( \ell > 5 \cdot 10^5 \) with \( p_m = 0.01 \) and the \( \beta_{ext} \)-protocol is superior than the \( \beta \)-protocol when \( \ell > 3.5 \cdot 10^6 \) and \( p_m = 0.001 \) (under the stated requirements in our investigations). The key length for which the \( \alpha_{ext} \) and \( \beta_{ext} \)-protocols are superior than the \( \alpha \) and \( \beta \)-protocols essentially depends on the error probabilities in
the communication channels.

The $\beta_{\text{ext}}$-protocol is superior than the $\alpha_{\text{ext}}$-protocol, although these protocols have the same asymptotic key rate. Hence, protocols with extractors are superior than the protocols with hashing, when $p_m = 0$.

4 Key distribution protocols under the condition that legal users shared short authentication key before starting the KDP

The $\alpha'$ and $\beta'$-protocols have been introduced in [22], which differ from the $\alpha$ and $\beta$-protocols in that legal users A and B have got a short key $S_A = S_B$ of length $\ell_0$ before starting the KDP. This key can be used for authentication of messages transmitted over public discussion channels in order to get finally the key of length $\ell >> \ell_0$.

In this section we consider some modification of the $\alpha'$ and $\beta'$-protocols in which instead of hash
functions, extractors are used in order to generate the final key. We call these protocols the $\alpha_{\text{ext}}'$- and $\beta_{\text{ext}}'$- protocols, respectively.

**$\alpha_{\text{ext}}'$-protocol** Let us suppose that the users A and B have binary strings $X^k$, $Y^k$ respectively.

1. The user A calculates the check string $C^r$ of length $r$ for the string $X^k$ using an error correcting $(k + r, k)$-code $C$ that should be agreed between the legal users in advance.

2. The user A generates a random binary string $\gamma^u$ of length $u$.

3. The user A computes the authenticator $w$ for the message $(C^r, \gamma^u)$ using a keyed hash function from the $\varepsilon$-ASU$_2$ class, and the key $S_A$.

4. The user A sends to user B over a PDC the message $(C^r, \gamma^u)$ appending to it the authenticator $w$.

5. The user B verifies the authenticity of $(C^r, \gamma^u)$ using the algorithm presented in section 2. If the authenticity of $(C^r, \gamma^u)$ is confirmed, then B goes to the next step, otherwise he rejects it.

6. The user B corrects errors in the string $Y^k$ using the check string $C^r$. (We denote by $\tilde{Y}^k$ the string $Y^k$ after error correction).

7. Both users A and B compute their keys as $K_A = E_{\text{ext}}(X^k, \gamma^u)$, $K_B = E_{\text{ext}}(\tilde{Y}^k, \gamma^u)$.

**$\beta_{\text{ext}}'$-protocol** In a similar manner there is a modified $\beta$-protocol where the random string $\gamma^u$ is not transmitted over the PDC but it is formed from the random sequences $X^k$, $Y^k$.

The users A and B divide each of the strings $X^k$, $Y^k$ obtained after execution of the initialization phase into two disjoint substrings $X_1^{k_1}$, $X_2^{k_2}$; $Y_1^{k_1}$, $Y_2^{k_2}$, respectively. Then they perform the following steps:

1. The user A calculates the check string $C_1^{r_1}$ of length $r_1$ for the substring $X_1^{k_1}$ using an error correcting $(k_1 + r_1, k_1)$-code $C_1$.

2. The user A calculates the check string $C_2^{r_2}$ of length $r_2$ for the substring $X_2^{k_2}$ using an error correcting $(k_2 + r_2, k_2)$-code $C_2$. 
3. The user A forms the authenticator $w$ for the message $(C_1^{r_1}, C_2^{r_2})$, using a keyed hash function from the class $\epsilon$-ASU_2 and the key $S_A$ with length $\ell_0$.

4. The user A sends to B the message $(C_1^{r_1}, C_2^{r_2})$ appended with the authenticator $w$.

5. The user B verifies the authenticity of the message $(C_1^{r_1}, C_2^{r_2})$ using the authentication algorithm (see section 2) and the key $S_B$. If authenticity is confirmed, then user B goes to the next step, otherwise he rejects the KDP.

6. The user B corrects errors in the strings $Y_1^{k_1}, Y_2^{k_2}$ using the check strings $C_1^{r_1}$ and $C_2^{r_2}$. (We denote by $\hat{Y}_1^{k_1}, \hat{Y}_2^{k_2}$ the strings $Y_1^{k_1}, Y_2^{k_2}$ after error correction.)

7. The user A takes the string $X_2^{k_2}$ as seed $\gamma_u$ and the user B takes the string $\hat{Y}_2^{k_2}$ as seed $\gamma_u$.

8. Both users A and B compute their keys as $K_A = E_{ext}(X_1^{k_1}, X_2^{k_2})$, $K_B = E_{ext}(\hat{Y}_1^{k_1}, \hat{Y}_2^{k_2})$.

Let us estimate the key rate of these protocols.

**Theorem 15** Let us suppose that the users A, B and the adversary E have binary strings $X^k$, $Y^k$ and $Z^k$, respectively after execution of the initialization phase over the wire-tape channel, $p_m = \Pr(x_i \neq y_i), p_w = \min(\Pr(x_i \neq z_i), \Pr(y_i \neq z_i), p_m \geq 0, p_w > p_m$. We assume that the users A and B share initially a short key $S$ of length $\ell_0$ in order to authenticate messages transmitted over the PDC.

Then A and B are able to form a common key of length $\ell$ satisfying the requirements \[(73)\] after the execution of the $\alpha'_ext$ and $\beta'_ext$-protocols if the lengths $k_1, k_2$ of substrings $X^k$, $Y^k$ and $Z^k$ and $\ell_0$ satisfy the equations listed below:

$$k_1 = -\frac{\log P_{adm}}{E(R_{c_1})},$$

$$k_1 \cdot H_{\infty} = \ell c + r_1 - \log P_{adm} + u + 3 \log \frac{I_{adm}}{2^u} - 2 + 3,$$  \[(74)\]

where $R_{c_1}$ and $E(R_{c_1})$ are calculated by \[(10)-12\) and

$$k_2 = \begin{cases} 
0 & \text{for the } \alpha'_ext \text{-protocol,} \\
\ell & \text{for the } \beta'_ext \text{-protocol,} 
\end{cases}$$  \[(75)\]
and

\[
 u = \left\lfloor \log k_1 \left( \frac{I_{adm}}{2 \ell} \right)^{-2} \right\rfloor \cdot \log k_1 \left( \frac{I_{adm}}{2 \ell} \right)^{-2}
\]  

(76)

is the number of the extractor random symbols, \( c \) is the parameter under optimization,

\[
\ell_0 = \frac{a(2 + i)}{2^i},
\]

(77)

\[
\frac{i + 1}{2^i} \leq P_{adm}^d,
\]

(78)

where

\[
a = \begin{cases} 
  r_1 + u & \text{for the } \alpha'_{ext}\text{-protocol,} \\
  r_1 + r_2 & \text{for the } \beta'_{ext}\text{-protocol,}
\end{cases}
\]

(79)

and \( r_2 \) being the number of check symbols of the error correcting \((k_2 + r_2, k_2)\)-code \( C_2 \) found similarly as in eq’s (73)-(74). The key rate is then determined as:

\[
R_{\alpha'_{ext}} = \max_c \frac{\ell}{k_1}, \quad R_{\beta'_{ext}} = \max_c \frac{\ell}{k_1 + k_2}.
\]

(80)

**Proof.** The relations (73), (74) and (76) can be proved similarly as (45), (46) and (47) in theorem 10. The relation (75) is apparent from the protocols description. In order to prove (77), (78) we assume that for authentication of messages of length \( a \) (see relation (79)) an \( \epsilon \)-ASU2-hash-function is used. Relying on (15) we write \( 2^a = q^{2^i}, 2^\ell_0 = q^{i+2}, \epsilon = \frac{i+1}{q}. \) Let us put \( q = 2^b \), then

\[
a = 2^b i, \quad \ell_0 = b(i + 2), \quad \epsilon = \frac{i + 1}{2^b}.
\]

(81)

Let us assume that the probability of false message deception is equal to \( \epsilon = P_{adm}^d. \) Then from (81), the relations (77), (78) are valid. The relation (80) follows from the protocols definition taking into account that the number of the extractor random bits can be optimized with respect to \( c. \)

\[
\Box
\]

By substituting (74) into (80) and using (13) we get that as \( \ell \to \infty: \)

\[
R_{\alpha'_{ext}} = \frac{\ell [H_\infty - g(p_m)]}{Den_1},
\]

(82)

\[
R_{\beta'_{ext}} = \frac{\ell [H_\infty - g(p_m)]}{Den_2},
\]

(83)
where

\[ \text{Den}_1 = \ell c - \log P_{\text{risk}} + u + 3 \log \ell \left( \frac{I_{\text{adm}}}{2\ell} \right)^{-2} + 3, \]

\[ \text{Den}_2 = \text{Den}_1 + u. \]

From (82), (83), we have \( R_{\alpha'} \geq R_{\beta'}\). When \( \ell \to \infty \) both protocols have the same key rates

\[ R_{\alpha'_{\text{ext}}} = R_{\beta'_{\text{ext}}} = H_{\infty} - g(p_m). \quad (84) \]

Let us compare the key rates of the \( \alpha'_{\text{ext}} \) and \( \beta'_{\text{ext}} \)-protocols and the \( \alpha' \) and \( \beta' \)-protocols. In [22] the following relations have been proved:

\[ \begin{align*}
R_{\alpha'} &\to H_2(p_w) - g(p_m) \\
R_{\beta'} &\to \frac{1}{2} [H_2(p_w) - g(p_m)] \end{align*} \]

as \( \ell \to \infty \).

Comparing these relations with (84) we may conclude that \( R_{\alpha'} \geq R_{\alpha'_{\text{ext}}} = R_{\beta'_{\text{ext}}} \) for any values \( p_w \) and \( p_m \). \( R_{\beta'} \) can be either larger or smaller than \( R_{\alpha'_{\text{ext}}} = R_{\beta'_{\text{ext}}} \) depending on the ratio of \( p_w \) and \( p_m \).

In order to illustrate the above assertions we plot in figure [6] the dependence of the key rate versus its length for the \( \alpha' \), \( \beta' \), \( \alpha'_{\text{ext}} \) and \( \beta'_{\text{ext}} \)-protocols, given \( p_w = 0.2, p_m = 0.01 \) and \( I_{\text{adm}} = 10^{-30} \), \( P_e = P_d = P_{\text{risk}} = 10^{-5} \).
Table 1: Groups of hybrid protocols

| Protocol group | Authentication key generation | Privacy generation |
|----------------|-------------------------------|--------------------|
| 1              | $(\alpha,\alpha'), (\alpha,\beta'), (\beta,\alpha'), (\beta,\beta')$ | hashing, hashing |
| 2              | $(\alpha,\alpha'_e), (\alpha,\beta'_e), (\beta,\alpha'_e), (\beta,\beta'_e)$ | hashing, extracting |
| 3              | $(\alpha_e,\alpha'), (\alpha_e,\beta'), (\beta_e,\alpha'), (\beta_e,\beta')$ | extracting, hashing |
| 4              | $(\alpha_e,\alpha'_e), (\alpha_e,\beta'_e), (\beta_e,\alpha'_e), (\beta_e,\beta'_e)$ | extracting, extracting |

5 Two-stage (hybrid) protocols with extractors

We remember that the hybrid protocols [22, 25] are combinations of protocol pairs $(\alpha,\alpha'), (\alpha,\beta'), (\beta,\alpha'), (\beta,\beta')$ where the first protocol in each pair is used in order to generate a relatively short key $S$ of length $\ell_0$ required for hash function and check bits authentication, whereas the second protocol is used in order to form the final key $K$.

The keys $S$ and $K$ can be obtained by execution of the privacy amplification procedure based either on the use of hash functions or extractors. This means that for every above mentioned hybrid protocol pair, there are four variants of hashing or extracting applications. In total, there can be formed 16 protocols, which in turn can be split into four groups as shown in table 1.

The first group of protocols was investigated in [22], and there it has been proved that every such protocol can be the most efficient depending on the additional key requirements imposed to it. It is worth to note that even for large length $\ell$ of the key $K$, the length $\ell_0$ of the authentication key occurs moderate [22] (p. 2543). If $\ell = 32000$ ($p_m = 0.01, p_w = 0.2, P_{adm} = 5 \cdot 10^{-6}$), then $\ell_0 = 678$. But since, as shown in section 3, extractors are superior than hash functions only with large key lengths, their application is useless in the first stage of the hybrid protocols, where a short key is required.

Therefore, the protocols from groups 3 and 4 have not been considered. It is sufficient to investigate protocols from the second group, where the authentication key is generated by hashing and the generation of the final keys is performed by extraction. Thus we consider the following
hybrid protocols: $(\alpha, \alpha'_\text{ext}), (\alpha, \beta'_\text{ext}), (\beta, \alpha'_\text{ext}), (\beta, \beta'_\text{ext})$. For a more detailed description with the design of these protocols and the specification requirements of each protocol component, we refer to [22] (p. 2544).

Let us give a short description of the $(\alpha, \alpha'_\text{ext})$-protocol. It is based on the $(\alpha, \alpha')$-protocol proposed by Korzhik and Morales [25]. In this protocol, the sequences $X^k, Y^k$ of users A and B are divided into three parts $X_1^k, X_2^k, X_3^k$ and $Y_1^k, Y_2^k, Y_3^k$ respectively (see figure 7-a). The subsequences $X_1^{k_1}, X_2^{k_2}, Y_1^{k_1}, Y_2^{k_2}$ are used for the generation of the authentication keys $S_A (S_B)$. The subsequence $X_3^{k_3}, Y_3^{k_3}$ and the key $S_A (S_B)$ are used in the $\alpha'_\text{ext}$-protocol for final $K_A (K_B)$ key generation. The key rate of this protocol can be written as

$$R_{(\alpha, \alpha'_\text{ext})} = \frac{\ell}{k_1 + k_2 + k_3} = \frac{\ell}{k_3 + \ell_0 / R_\alpha},$$

(85)

where $R_\alpha = \frac{\ell}{k_1 + k_2}$ is the key $S$ rate at the length $\ell_0$ in the $\alpha$-protocol.

Let us prove the following lemma.

**Lemma 2** The convergence $\frac{\ell_0}{\ell} \to 0$, as $\ell \to \infty$, holds in the $(\alpha, \alpha'_\text{ext})$-protocol.

**Proof.** Let us consider the $(\alpha, \alpha'_\text{ext})$-protocol. According to [79] the input block length of the

![Figure 7: Different types of hybrid protocol with extractors.](image)
\( \epsilon \)-ASU\(_2\) hash function used in the \( \alpha_{ext} \)-protocol is equal to \( a = r_3 + u \). We show initially that \( r_3 + u \to \infty \) as \( \ell \to \infty \). The relations (76) and (11) give

\[
    u \approx \left( \log k_3 \left( \frac{I_{adm}}{2^\ell} \right)^{-2} \right)^2 / \ln c, \tag{86}
\]

\[
    r_3 = (1 - R_c) k_3. \tag{87}
\]

Thus in order to prove that \( r_3 + u \to \infty \) it is necessary to show that \( k_3 \to \infty \) as \( \ell \to \infty \). Taking into account equation (13), we can present (74) in the form

\[
    k_3 = \frac{\ell c - \log P_{adm} \log P_{adm} + u + 3 \log \ell \left( \frac{I_{adm}}{2^\ell} \right)^{-2} + 3}{H_\infty(p_w) - g(p_m)}. \tag{88}
\]

From the above relation, it follows that \( k_3 \to \infty \) as \( \ell \to \infty \) and both (86), (87) result as \( r_3 \to \infty \) and \( u \to \infty \). Then using (11) for an estimation of the hash function parameters, we may write

\[
    \lim_{i \to \infty} \frac{\ell_0}{a} = \lim_{i \to \infty} \frac{i + 2}{2^i} = 0.
\]

Using (88), the relation (85) can be expressed as

\[
    R_{(\alpha, \alpha'_{ext})} = \frac{H_\infty(p_w) - g(p_m)}{\text{Den}_3}, \tag{89}
\]

where

\[
    \text{Den}_3 = c - \ell^{-1} \left[ \log P_{adm} \log P_{adm} + u + 3 \log \ell \left( \frac{I_{adm}}{2^\ell} \right)^{-2} + 3 \right] + \frac{\ell_0}{\ell} \frac{H_\infty(p_w) - g(p_m)}{R_\alpha}.
\]

Taking into account that \( R_\alpha \) is constant and \( \frac{\ell_0}{\ell} \to 0 \) (see Lemma 2) we can see that the last term in the denominator \( \text{Den}_3 \) of (89) approaches to zero as \( \ell \to \infty \). The other terms in the denominator \( \text{Den}_3 \) also approach to zero because they consist either of values approaching zero or
have a logarithmic dependence on \( \ell \).

Since the right side of (89) approaches a maximum, \( R_{(\alpha, \alpha'_{ext})} \) approaches a maximum as \( c \to 1 \), then the following asymptotic estimation holds for the key rate of the \((\alpha, \alpha'_{ext})\)-protocol

\[
R_{(\alpha, \alpha'_{ext})} = H_\infty(p_w) - g(p_m), \tag{90}
\]

\[
R^*_{(\alpha, \alpha'_{ext})} = H_\infty(p_w) \text{ as } p_m \to 0. \tag{91}
\]

In the \((\beta, \alpha'_{ext})\)-protocol (see figure 7-b) the sequences \( X^k, Y^k \) are divided into four parts \( X^k_1, X^k_2, X^k_3, X^k_4 \) and \( Y^k_1, Y^k_2, Y^k_3, Y^k_4 \), respectively. The subsequences \( X^k_1, X^k_2, X^k_3, Y^k_1, Y^k_2, Y^k_3 \) are used in the \(\beta\)-protocol in order to generate the authentication key \( S \) of length \( \ell_0 \). The subsequence \( X^k_4, Y^k_4 \) is used in the \(\alpha'_{ext}\)-protocol for final keys \( K_A, K_B \) generation. One can write

\[
R_{(\beta, \alpha'_{ext})} = \frac{\ell}{k_4 + \frac{\ell_0}{R_\beta}},
\]

where \( R_\beta = \frac{\ell_0}{k_1 + k_2 + k_3} \) is the authentication key rate.

By comparing this protocol with the previous one, we can conclude that for the same length \( \ell \) of the final key, the equality \( k_4 = k_3 \) should hold. As it was shown in [22], \( R_\beta \geq R_\alpha \) and the length \( \ell_0 \) of authentication key for the \(\alpha\)-protocol is larger than the length \( \ell_0 \) of the authentication key for the \(\beta\)-protocol. Hence

\[
\frac{\ell_0(\alpha\text{-protocol})}{R_\alpha} \geq \frac{\ell_0(\beta\text{-protocol})}{R_\beta}
\]

and \( R_{(\beta, \alpha'_{ext})} \geq R_{(\alpha, \alpha'_{ext})} \).

It is easy to show that

\[
R_{(\beta, \alpha'_{ext})} = H_\infty(p_w) - g(p_m) \text{ as } \ell \to \infty, \tag{92}
\]

that coincides with the key rate of the \((\alpha, \alpha'_{ext})\)-protocol, see [91]. If \( p_m = 0 \) we get by (92)

\[
R^*_{(\beta, \alpha'_{ext})} = H_\infty(p_m).
\]

Next let us consider the \((\alpha, \beta'_{ext})\)-protocol (see figure 7-c) in which each sequence \( X^k, Y^k \) is
divided into four parts \(X_{1}^{k_{1}}, X_{2}^{k_{2}}, X_{3}^{k_{3}}, X_{4}^{k_{4}}\) and \(Y_{1}^{k_{1}}, Y_{2}^{k_{2}}, Y_{3}^{k_{3}}, Y_{4}^{k_{4}}\), respectively. The subsequences \(X_{1}^{k_{1}}, X_{2}^{k_{2}}\) and \(Y_{1}^{k_{1}}, Y_{2}^{k_{2}}\) are used in the \(\alpha\)-protocol for the generation of the key \(S_{A}\), or \(S_{B}\) for the \(\beta'_{ext}\)-protocol, while subsequences \(X_{3}^{k_{3}}, X_{4}^{k_{4}}\) and \(Y_{3}^{k_{3}}, Y_{4}^{k_{4}}\) are used in the \(\beta'_{ext}\)-protocol for the final key \(K_{A}\) \((K_{B})\) generation assuming \(X_{4}^{k_{4}}\) and \(Y_{4}^{k_{4}}\) as random “seeds” \(\gamma^{k_{4}}\) while using in extractor.

Similarly to (85) we can write

\[
R_{(\alpha,\beta'_{ext})} = \frac{\ell}{k_{3} + u + \frac{\ell_{0}}{R_{\alpha}}},
\]

The parameter \(\ell_{0}\) is smaller in this protocol than in the \((\alpha, \alpha'_{ext})\)-protocol, because only to authenticate the check sequences \(X_{3}^{k_{3}}\) and \(X_{4}^{k_{4}}\) of total length \(r_{1} + r_{2}\) there is used an \(\epsilon\)-AS\(U_{2}\) hash-function. However the additional item \((u)\) calculated by (76) increases the denominator and hence

\[
R_{(\alpha,\beta'_{ext})} \leq R_{(\alpha,\alpha'_{ext})}.
\]

In the \((\beta, \beta'_{ext})\)-protocol, each sequence \(X^{k}, Y^{k}\) is divided into five parts \(X_{1}^{k_{1}}, X_{2}^{k_{2}}, X_{3}^{k_{3}}, X_{4}^{k_{4}}, X_{5}^{k_{5}}\) and \(Y_{1}^{k_{1}}, Y_{2}^{k_{2}}, Y_{3}^{k_{3}}, Y_{4}^{k_{4}}, Y_{5}^{k_{5}}\), respectively (see figure 7d). The subsequences \(X_{1}^{k_{1}}, X_{2}^{k_{2}}, X_{3}^{k_{3}}, X_{4}^{k_{4}}, X_{5}^{k_{5}}\) and \(Y_{1}^{k_{1}}, Y_{2}^{k_{2}}, Y_{3}^{k_{3}}, Y_{4}^{k_{4}}, Y_{5}^{k_{5}}\) are used in the \(\beta\)-protocol to generate the keys \(S_{A}, S_{B}\). The subsequences \(X_{4}^{k_{4}}, X_{5}^{k_{5}}\) and \(Y_{4}^{k_{4}}, Y_{5}^{k_{5}}\) are used in the \(\beta'_{ext}\)-protocol to generate the keys \(K_{A}, K_{B}\). Let us write

\[
R_{(\beta,\beta'_{ext})} = \frac{\ell}{k_{4} + u + \frac{\ell_{0}}{R_{\beta}}},
\]

In this relation by the same reason mentioned during the analysis of the \((\alpha, \beta'_{ext})\)-protocol, the value \(\ell_{0}\) will be smaller than in the \((\beta, \alpha'_{ext})\)-protocol and the rate \(R_{\beta}\) is larger. Therefore the third item is slightly decreasing. However the presence of sufficiently large item \(u\) results in a key rate decreasing giving the inequality \(R_{(\beta,\beta'_{ext})} \leq R_{(\beta,\alpha'_{ext})}\).

There may be for the \((\alpha, \beta'_{ext}), (\beta, \alpha'_{ext})\)-protocols some equivalent statement to Lemma 2, e.g. it can be proved that \(\frac{\ell_{0}}{\ell} \rightarrow 0\) as \(\ell \rightarrow \infty\). Furthermore, by writing the relations for \(k_{3}, k_{4}\) and \(u\), from theorem 12 it is very simple to get asymptotically achievable the key rate for the \((\alpha, \beta'_{ext})\), \((\beta, \alpha'_{ext})\)-protocols.

\[
R_{(\alpha,\beta_{ext})} = R_{(\beta,\beta'_{ext})} = H_{\infty}(p_{w}) - g(p_{m}).
\]  

(93)

Comparing (90), (92) and (93) we can see that asymptotically all hybrid protocols have the same
key rates, however the \((\beta, \alpha'_{\text{ext}})\)-protocol has non-asymptotically the largest key rate among all above considered hybrid protocols.

It is worth to compare this protocol with the \((\beta, \alpha')\)-protocol, that (as shown in \cite{22}) has maximum possible key rate for sufficiently large \(\ell\) among all hybrid protocols using hash functions in the privacy amplification procedure.

For the \((\beta, \alpha')\)-protocol one can write \cite{22}

\[
R_{(\beta, \alpha')} = H_2(p_m) - g(p_m) \tag{94}
\]

By comparing \eqref{92} with \eqref{94} and taking into account that \(H_2(p_m) \geq H_{\infty}(p_m)\) we can see that for large \(\ell\), \(R_{(\beta, \alpha'_{\text{ext}})} \geq R_{(\beta, \alpha'_{\text{ext}})}\).

From this inequality, it follows that an implementation of extractors for large key length in hybrid protocols are inefficient.

In figure \[\text{8}\] there are plotted the key rates versus its length for hybrid protocols under the conditions \(p_m = 0.01, p_w = 0.2, I_{\text{adm}} = 10^{-30}, P_{e_{\text{adm}}} = P_{d_{\text{adm}}} = P_{f_{\text{adm}}} = 10^{-5}\).

The curves \(R(\ell)\) were plotted with the use of the technique proposed in \cite{22}. They clearly demonstrate a behavior of the key rate depending on the key length for different protocols. We can see that the \((\alpha, \alpha')\) and \((\beta, \alpha')\)-protocols have the greatest key rates among all hybrid protocols. The \((\alpha, \beta')\), \((\beta, \beta')\), \((\alpha, \alpha'_{\text{ext}})\), and \((\beta, \alpha'_{\text{ext}})\)-protocols have approximately equal key rates for large key
length and the \((\alpha, \beta'_{\text{ext}})\) and \((\beta, \beta'_{\text{ext}})\)-protocols have the least key rate among the above considered hybrid protocols.

6 Conclusions

In the current paper, an investigation of key distribution protocols based on noisy channels started in [22] has been continued with such a difference that extractors are used instead of hash-functions in the privacy amplification procedure. The main goal was to prove extractor-based protocols efficiency by the criterion of key rate maximization. The relations are non-asymptotic and constructive because they do not include some uncertain coefficients in notations, in contrast with other papers.

We use the modified Trevisan’s extractor [30, 29] in our paper. It has been proposed new \(\beta_{\text{ext}}\) and \(\beta^*_{\text{ext}}\)-protocols which differ from those known before [17] because the extractor’s seed is not transmitted over the PDC but, instead, it is generated from random sequences obtained by legal user after the execution of the initialization phase. We proved that the use of extractors in the \(\alpha_{\text{ext}}\) and \(\beta_{\text{ext}}\)-protocols increases the rate, in comparison with hashing-based protocols only for very large key length \(\ell\) (typically \(\ell \in [10^5, 10^6]\)) and for some specified values of the error probabilities both in the main and in the wire-tap channels.

It was investigated a performance evaluation of the so called extractor-based hybrid protocols, consisting of two protocols executed in a serial manner where the first protocol in a pair is used for the generation of a relatively short key \(S\) of length \(\ell_0\). This key is necessary for authentication of check bits, and a random number (seed) of extractor. The second protocol is used for the final key generation. We prove that extractor based protocols should be used only in the second protocol of the pair.

We selected four hybrid protocols for further investigation \((\alpha, \alpha'_{\text{ext}})\), \((\beta, \alpha'_{\text{ext}})\), \((\alpha, \beta'_{\text{ext}})\) and \((\beta, \beta'_{\text{ext}})\)-protocols. The relations for their key rates have been derived for both finite and asymptotically growing key lengths. The greatest key rate is got for the \((\beta, \alpha'_{\text{ext}})\)-protocol. This protocol was compared with the \((\beta, \alpha')\)-protocol considered in [22], which has the greatest key rate among all hybrid hashing-based protocols. The investigations showed that hybrid protocols with the use of extractor-based second stage protocols are less efficient than hashing-based protocol.
We investigated also (but not in a deeper detail) other variants of extractors from [29]. Even with some improvement of their characteristics (in the sense of the seed length), the general conclusion is kept the same: the use of extractors is justified only with very large key length.

We get also asymptotic estimates for the key rates of all proposed protocols that allows to compare the potential efficiency of all considered early protocols. These relations are presented in table 2. We can see that asymptotically all hybrid protocols have the same key rates equal to $H_\infty(p_w) - g(p_m)$, that is larger than the key rates for single $\alpha_{ext}$ and $\beta_{ext}$-protocols that is equal to $\frac{H_\infty(p_w) - g(p_m)}{1 + 2g(p_m)}$.

These relations are similar “on structure” to relations for key capacity $g(p_w) - g(p_m)$, [19], but differ from the last one in changing of Shannon’s entropy $g(p_w)$ to min entropy $H_\infty(p_w)$.

If the main channel is noiseless then all protocols using extractors have the same asymptotic key rates equal to $H_\infty(p_w)$. It is worth to note that asymptotically all hybrid extractor-based protocols are inferior to hash-based protocols. But this conclusion may be considered as a consequence of crude estimate of information leaking to eavesdropper based on the use of min entropy.

We summarize the key rates for different KDP in table 2. It can be seen from this table how closer or farther are the key rates to the secret key capacity given by (7).

References

[1] E. H. Knill, “Building quantum computers/ plenary talk presented at the 2007 ieee international symposium on informational theory,” IEEE Information Theory Society Newsletter, vol. 58, no. 4, pp. 32–35, Dec. 2008.

[2] C. E. Shannon, “Communication theory of secrecy systems.” Bell System Technical Journal, vol. 28, no. 4, pp. 656–715, 1949.

[3] M. E. Hellman, “An extension of the Shannon theory approach to cryptography.” IEEE Transactions on Information Theory, vol. 23, no. 2, pp. 289–294, 1977.

[4] C. H. Bennett and G. Brassard, “Quantum cryptography: Public key distribution and coin tossing,” in Proceedings of International Conference on Computers, Systems and Signal Processing, December 1984.
Table 2: Key rates for different KDP

| Protocol type | (2)                                           | (3)                                           | (4)                                           |
|---------------|-----------------------------------------------|-----------------------------------------------|-----------------------------------------------|
| \(\alpha_{ext}\) | \(\frac{T_2}{T_1} \frac{1}{(1+2g(p_m))} + 2u + 2\tau_0\) | \(\frac{T_1}{1+2g(p_m)}\) | \(T_0\)                                      |
| \(\beta_{ext}\) | \(\frac{T_2}{T_1} + u\) \frac{1}{(1+2g(p_m))} + 2\tau_0\) | \(\frac{T_1}{1+2g(p_m)}\) | \(T_0\)                                      |
| \(\alpha'_{ext}\) | \(\frac{T_1}{T_2} \frac{1}{T_0}\) | \(T_1\) | \(T_0\)                                      |
| \(\beta'_{ext}\) | \(\frac{T_1}{T_2} \frac{1}{T_0}\) | \(T_1\) | \(T_0\)                                      |
| \((\alpha, \alpha'_{ext})\) | \(\frac{T_1}{T_2 + T_0} \frac{1}{T_0}\) | \(T_1\) | \(T_0\)                                      |
| \((\beta, \alpha'_{ext})\) | \(\frac{T_1}{T_2 + T_0} \frac{1}{T_0}\) | \(T_1\) | \(T_0\)                                      |
| \((\alpha, \beta'_{ext})\) | \(\frac{T_1}{T_2 + u + T_0} \frac{1}{T_0}\) | \(T_1\) | \(T_0\)                                      |
| \((\beta, \beta'_{ext})\) | \(\frac{T_1}{T_2 + u + T_0} \frac{1}{T_0}\) | \(T_1\) | \(T_0\)                                      |

(2): Non asymptotic relations for key rates \(R_k\)
(3): Asymptotic relations for \(R_k\), as \(\ell \to \infty\)
(4): Asymptotic relations for \(R_k\), as \(\ell \to \infty, p_m \to 0\)

\[
T_0 = \frac{H_\infty(p_w)}{}
\]
\[
T_1 = T_0 - g(p_m)
\]
\[
T_2 = c\ell + u + a
\]
\[
a = -2P_{risk}^{adm} + 3\log \ell \left(\frac{I_{adm}}{2\ell}\right)^{-2} + 3
\]

\(u\) is given by relation (48) and \(R_\alpha, R_\beta\) were introduced in [22].
[5] T. Aono, K. Higuchi, T. Ohira, B. Komiyama, and H. Sasaoka, “Wireless secret key generation exploiting reactance-domain scalar response of multipath fading channels,” *IEEE Transactions on Antennas and Propagation*, vol. 53, no. 11, pp. 3776–3784, 2005.

[6] A. Kitaura and H. Sasaoka, “A scheme of private key agreement based on the channel characteristics in OFDM land mobile radio.” *Electronics and Communications in Japan (Part III: Fundamental Electronic Science)*, vol. 88, no. 9, pp. 1–10, 2005.

[7] H. Mori, H. Sasaoka, and T. Ohira, “Performance estimation of secret key agreement system exploiting an ESPAR antenna and a received signal strength indicator,” in *ISSSE2004*. IEICE, 2004.

[8] H. Imai, K. Kobar, and K. Morozov, “On the possibility of key agreement using variable directional antenna,” in *Proc. 1-st Joint Workshop on Information Security*, 2006, pp. 1432–1436.

[9] R. Ahlswede and I. Csiszar, “Common randomness in information theory and cryptography – part I: Secret sharing,” *IEEE Transactions on Information Theory*, vol. 39, no. 4, pp. 1121–1132, 1993. [Online]. Available: citeseer.ist.psu.edu/ahlswede93common.html

[10] C. H. Bennett, G. Brassard, C. Crepeau, and U. M. Maurer, “Generalized privacy amplification,” *IEEE Transactions on Information Theory*, vol. 41, no. 6, pp. 1915–1923, 1995.

[11] U. Maurer, “Secret key agreement by public discussion from common information.” *IEEE Transactions on Information Theory*, vol. 39, no. 3, pp. 733–742, 1993.

[12] ———, “Information-theoretically secure secret-key agreement by not authenticated public discussion,” *Lecture Notes in Computer Science*, vol. 1233, pp. 209–223, 1997.

[13] ———, “Protocols for secret key agreement by public discussion based on common information.” in *CRYPTO*, ser. Lecture Notes in Computer Science, E. F. Brickell, Ed., vol. 740. Springer, 1992, pp. 461–470.

[14] U. Maurer and S. Wolf, “Privacy amplification secure against active adversaries,” *Lecture Notes in Computer Science*, vol. 1294, pp. 307–321, 1997.
[15] ——, “Secret-key agreement over unauthenticated public channels I: Definitions and a completeness result.” IEEE Transactions on Information Theory, vol. 49, no. 4, pp. 822–831, 2003.

[16] ——, “Secret-key agreement over unauthenticated public channels II: the simulatability condition.” IEEE Transactions on Information Theory, vol. 49, no. 4, pp. 832–838, 2003.

[17] ——, “Secret-key agreement over unauthenticated public channels III: Privacy amplification.” IEEE Transactions on Information Theory, vol. 49, no. 4, pp. 839–851, 2003.

[18] ——, “Unconditionally secure key agreement and the intrinsic conditional information.” IEEE Transactions on Information Theory, vol. 45, no. 2, pp. 499–514, 1999.

[19] ——, “Towards characterizing when information-theoretic secret key agreement is possible.” in ASIACRYPT, ser. Lecture Notes in Computer Science, K. Kim and T. Matsumoto, Eds., vol. 1163. Springer, 1996, pp. 196–209.

[20] V. Korzhik, V. Yakovlev, and A. Sinuk, “Achievability of the key-capacity in a scenario of key sharing by public discussion in the presence of passive eavesdropper,” in MMM-ACNS. Springer Lecture Notes in Computer Science 2776, 2003, pp. 308–315.

[21] ——, “Key distribution based on noisy channel and error detecting codes,” in Proc. Workshop MMM ACNS 2001. Springer Lecture Notes in Computer Science 2052, 2001, pp. 242–250.

[22] V. Yakovlev, V. I. Korzhik, and G. Morales-Luna, “Key distribution protocols based on noisy channels in presence of an active adversary: Conventional and new versions with parameter optimization,” IEEE Transactions on Information Theory, vol. 54, no. 6, pp. 2535–2549, 2008.

[23] V. Korjik, G. Morales-Luna, and V. Balakirsky, “Privacy amplification theorem for noisy main channel,” Lecture Notes in Computer Science, vol. 2200, pp. 18–26, 2001.

[24] D. Stinson, “Universal hashing and authentication codes,” Lecture Notes in Computer Science, vol. 576, pp. 74–85, 1992.

[25] V. I. Korzhik and G. Morales-Luna, “Hybrid authentication based on noisy channels.” Int. J. Inf. Sec., vol. 1, no. 4, pp. 203–210, 2003.
[26] V. Korzhik, V. Yakovlev, G. Morales-Luna, and R. Chesnokov, “Performance evaluation of keyless authentication based on noisy channel,” in Computer Network Security, ser. Communications in Computer and Information Science, V. Gorodetsky, I. Kotenko, and V. A. Skormin, Eds., vol. 1. Springer, 2007, pp. 115–126.

[27] V. Korjik and M. Bakin, “Information-theoretically secure keyless authentication,” in Proc. IEEE Symp. on IT’2000. IEEE, 2000, p. 20.

[28] R. Raz, O. Reingold, and S. P. Vadhan, “Extracting all the randomness and reducing the error in Trevisan’s extractors.” J. Comput. Syst. Sci., vol. 65, no. 1, pp. 97–128, 2002.

[29] ——, “Error reduction for extractors,” in FOCS ’99: Proceedings of the 40th Annual Symposium on Foundations of Computer Science. Washington, DC, USA: IEEE Computer Society, 1999, p. 191.

[30] L. Trevisan, “Construction of extractors using pseudo-random generators (extended abstract).” in Proceedings of the 31 annual ACM Symposium on Theory of Computing, 1999, pp. 141–148.

[31] V. Korzhik, V. Yakovlev, and G. Morales-Luna, “Non-asymptotic performance evaluation of key distribution protocols based on noisy channels in presence of active adversary,” in Proc. X. Spanish Meeting on Cryptology and Information Security, 2008, pp. 63–68.

[32] F. MacWilliams and N. Sloane, The Theory of Error-Correcting Codes. North-Holland, 1977.