Published in IET Information Security
Received on 20th December 2011
Revised on 27th October 2012
Accepted on 19th November 2012
doi: 10.1049/iet-ifs.2011.0360

ISSN 1751-8709

Channel level crossing-based security for communications over fading channels

Dimitrios S. Karas[1], *George K. Karagiannidis*[1], *Robert Schober*[2]

[1]*Department of Electrical and Computer Engineering, Aristotle University of Thessaloniki, Panepistimioulolis, GR-54124 Thessaloniki, Greece*
[2]*Department of Electrical and Computer Engineering, University of British Columbia, 2356 Main Mall, Vancouver, BC, CAN V6T 1Z4, Canada*
E-mail: dkaras@auth.gr

Abstract: Several key exchange methods for wireless channels have been proposed in the literature. They are referred to as physical-layer security techniques and are usually based on the channel's fading characteristics and the principle of channel reciprocity. In this study, the authors present key exchange algorithms for wireless fading channels whose operation is based on channel estimation. Specifically, the authors present a complete key exchange scheme that includes channel sampling, thresholding and error reconciliation. Two error reconciliation methods are proposed. The first one is based on neural networks and the second one is based on linear block coding. Simulations of the proposed methods' performances and levels of security are presented and conclusions are drawn regarding their overall utility.

1 Introduction

Cryptographic algorithms require a method to securely exchange cryptographic keys which are subsequently used to encrypt and decrypt the information messages. For example symmetric-key encryption algorithms use the same key or trivially related keys for both encryption and decryption. In most of these methods, it is required that both communicating parties are privy to the cryptographic key. This is achieved by using a secure key exchange algorithm which must be designed in such a way that the information received by an eavesdropper, who is wiretapping the communication channel, cannot be used to deduce the key or it would at least take the eavesdropper an extremely large amount of time to obtain the key.

1.1 Related literature

Numerous key exchange schemes have been developed for wired and wireless communications systems, including the Diffie–Hellman technique [1], which was the first published method for establishing a secret key over a communications channel. Over the past few years, a new class of key exchange methods over wireless channels has been proposed, which exploits the channel's properties and time response characteristics in order to facilitate the key exchange process [2–19]. These methods are referred to as physical (PHY)-layer key exchange algorithms in the literature. Traditional key exchange schemes provide security through computational complexity, meaning that if the computational capability of adversaries increases or if an algorithm for cracking such encryption methods is discovered in the future this kind of security is compromised. PHY-later security is a concept that is not affected by these shortcomings. Moreover, it has been shown [2] that, in theory, suitably long codes can come exponentially close to perfect secrecy. Perfect secrecy is achieved when the conditional entropy of the message M given the codeword X, $H(M|X)$, is equal to the entropy of the message $H(M)$. In fact, PHY-layer security may become a part of a multilayered security scheme where each layer achieves a specific security goal.

The concept of combining key exchange with physical layer characteristics was first presented in [3] in 1995. Since then, several methods have been proposed in this field. In [4], factors such as noise and interference were taken into consideration and a key exchange protocol was presented and tested in a real-world setting. Key generation by using Secure Fuzzy Information Reconciliators (SFIR), which are a class of randomness extractors that can perform error reconciliation, is also examined in [4]. Similar to the schemes in [4], the methods proposed in this work are also based on channel estimation and thresholding. Another channel thresholding based PHY-layer key exchange method was presented in [5]. Specifically, an algorithm that uses a quantiser formed by using the statistics of the wireless channel is proposed. The issue of user authentication – the process of validating the legitimacy of a communicating node and the prevention of a spoofing attack – is also addressed in [5].

Transmitter authentication in wireless channels was also explored in [6, 7], whereas in [8, 9] several key exchange methods were proposed with emphasis on the design of a high bit rate implementation. In [10], the concept of using

multiple-antenna diversity for key generation purposes was investigated.

Other research papers in the field deal with PHY-layer security in orthogonal frequency-division multiplexing (OFDM) systems [11], applications in static environments such as indoor networks [12], use of an adaptive quantisation algorithm for key exchange [13] and error reconciliation based on randomness extractors [14]. A practical implementation of a key sharing platform at 60 GHz was presented in [15], whereas the performance of several key exchange methods and their practical feasibility are discussed in [16–18]. Furthermore, in [19] PHY-layer security was studied in the presence of an adversary who is performing a jamming (denial-of-service) attack, reducing the efficiency of the channel. In contrast, Martinovic *et al.* [20] propose a method where jamming is used by the legitimate communicating nodes to hinder the adversary's activity.

Further information about PHY-layer security can be found in [2, 21] and references therein.

1.2 Contributions

In this work, a PHY-layer key exchange protocol for a wireless link between two transceivers is presented. Thereby, the principle of channel reciprocity is utilised so that the transceivers extract two highly correlated channel magnitude envelopes. The motivation for this approach is that the legitimate transceivers can utilise information that is only known to them and not the eavesdropper. As pointed out in [18], the channel magnitude envelopes will normally not be known to the eavesdropper. Furthermore, a novel thresholding process is proposed, where the channel magnitude envelope is sampled by the transceivers over a predetermined time duration, and a least-square curve is calculated by using a number of these samples as a data set. Each transceiver generates a bit string by comparing each sample of the magnitude envelope with the value of the least-square curve at the corresponding position. Thus, the transceivers generate two similar bit strings that provide the basis for the cryptographic key.

Owing to the existence of noise and various sources of interference, there will generally be discrepancies between the two generated bit strings. However, symmetric key cryptography requires that both transceivers possess identical cryptographic keys. In this work, two error reconciliation methods are proposed that enable both transceivers to generate identical bit strings by using the two similar bit strings and a process secure from eavesdropper activity. The first method's operation is based on a two-layer neural network. Specifically, one transceiver creates a neural network that is trained in such a way that it will output a randomly selected cryptographic key for inputs similar to the transceiver's bit string. Then, the neural network's parameters are transmitted to the other transceiver, who uses it in order to obtain the cryptographic key as an output by using its bit string as an input. The second method is based on a linear block code, where a bit masking method is applied for increased security. Specifically, an error correction code is used in order to correct the discrepancies between the two bit strings, and the average fade duration of the wireless channel is used in order to generate a bit mask that increases the security level of the method. In both cases, the produced key is known only to the legitimate transceivers and is secure against eavesdropper activity.

1.3 Organisation of the paper

The rest of the paper is organised as follows. In Section 2, the system and channel model is described and an overview of the proposed key exchange protocol is given. A new channel thresholding method is proposed in Section 3. In Section 4, we describe the proposed neural network based error reconciliation method. In Section 5, the proposed linear block coding-based error reconciliation method is presented. Simulation results are provided in Section 6, and some conclusions are drawn in Section 7.

2 System and channel model

A cryptographic key exchange scheme for a wireless communication setting has to adhere to some basic principles. First, it is assumed that the two transceiver nodes are communicating over a wiretapped channel. That being said, the key exchange scheme should (a) produce information-theoretically secure keys and (b) perform error reconciliation between the private keys generated by the transmitter and the receiver, such that they become identical.

It is assumed that the transceivers communicate over a slow fading additive white Gaussian noise (AWGN) channel. Nevertheless, the method presented in this paper is also applicable to other channel models such as Rician or Nakagami-m fading. Our approach utilises the reciprocity principle of wireless channels [2], according to which two transceivers operating in the same frequency band and communicating with each other experience the same channel characteristics at the same time. Also, we assume that the eavesdropper is at such a distance from both communicating nodes that the envelope of the signal received by the eavesdropper is uncorrelated with the signal received by the legitimate receivers. This assumption is valid for most practical scenarios. Specifically, an eavesdropper who is more than half a wavelength away from both legitimate transceivers will experience two independent fading channels to the two transceivers. According to [18], these channels are uncorrelated with the channel between the two legitimate nodes. We further assume that the number of deep fades in a specific time interval might be known to the eavesdropper but not their duration or their time location. It is also assumed that the hardware is such that the principle of reciprocity holds, as it has been documented in the open literature [3, 4].

The channel sampling procedure is that proposed in [21], which also takes into account the principle of reciprocity. Specifically, assuming that a transmitter sends a signal to a receiver, because of the principle of reciprocity, if it sends a signal back to the original transmitter, it will experience the same realisation of that fading at that instant. In order to take advantage of this property we assume that the legitimate transceivers A and B exchange L pilot signals. Specifically, when A transmits a signal, B measures the received signal strength, sends a signal back to A, who also measures the received signal strength and so forth. Note, that the time between two consecutive transmissions from the same transceiver, T_s, must be less than the channel coherence time in order to assume an identical impulse response between two consecutive channel samplings by both transceivers. Following this process, the transceivers can construct an estimate of the channel magnitude envelope, respectively, that each consists of L samples. Then, the transceivers apply a thresholding process to their respective sampled magnitude envelopes, so that they each

doi: 10.1049/iet-ifs.2011.0360

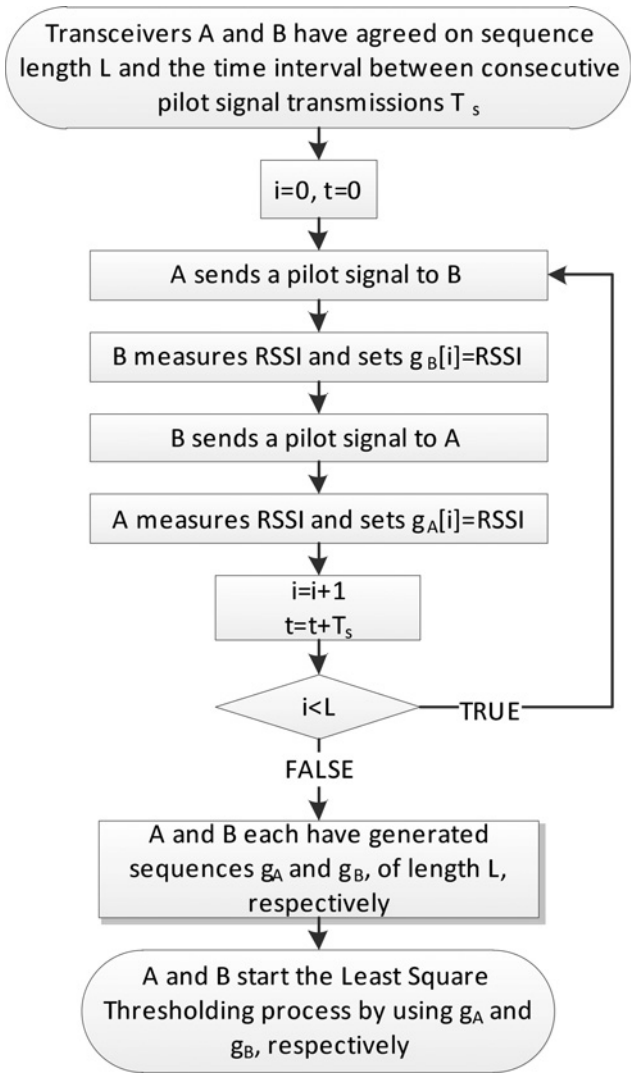

Fig. 1 *Flowchart of the channel sampling method*

generate a bit string of equal length. Fig. 1 depicts a flowchart representation of the channel sampling method. However, in practice, the principle of reciprocity holds only approximately, because of the presence of noise, various sources of interference and the characteristics of the specific hardware and synchronisation issues between the two transceivers in the sampling process. Thus, there are discrepancies between the bit strings generated by each transceiver, so they cannot be immediately used as a cryptographic key. In order to resolve this problem, a novel neural network-based error reconciliation scheme is proposed and described in Section 4, and a linear block coding-based scheme is presented in Section 5.

3 Least-square thresholding

In [4], a thresholding method was proposed, where the threshold of the sampled sequences is determined by an automatic gain control (AGC) mechanism, so that it is independent of the transmit power and the link attenuation. Here, we present an alternative thresholding method which is more efficient especially in environments where deep fades do not occur, for example in line-of-sight (LoS) situations. The motivation behind the proposed method is to detect fades of smaller depth in order to increase the security of the system. Specifically, a larger number of fades in a smaller time interval increases the time required for a brute force check by the eavesdropper. Also, the proposed method can cause uncertainty to the eavesdropper about the number of fades, if fades of smaller depth cannot be detected by the eavesdropper.

Let g_A and g_B be the sampled sequences of length L that both transceivers – A and B – generated by sampling the channel magnitude envelope. Each sample is represented by $(i, g_K[i])$, where i is the position of the sample in the sequence and $g_K[i]$, $K \in \{A, B\}$, denotes its value. Transceivers A and B form the sets S_A and S_B, respectively, which contain all local maxima and minima of g_A and g_B.

We define the sets $S_K^{\max}$ and $S_K^{\min}$, which contain the local maxima and minima of g_K, respectively, multiplied by a scaling factor, $u \leq 1$. The purpose of this factor is to lower the threshold curve so that it only detects dents in the channel magnitude envelope that can be attributed to multipath fading, and not other anomalies of the envelope that might not be present at both transceivers. These sets are formally defined as

$$S_K^{\max} = \{(i, ug_K[i]) | g_K[i-1] < g_K[i] \wedge g_K[i+1] < g_K[i], i = 2, \ldots, L-1\} \quad (1)$$

$$S_K^{\min} = \{(i, ug_K[i]) | g_K[i-1] > g_K[i] \wedge g_K[i+1] > g_K[i], i = 2, \ldots, L-1\} \quad (2)$$

Thus, the set S_K is

$$S_K = S_K^{\max} \cup S_K^{\min} \quad (3)$$

Then, each transceiver calculates a least-square polynomial curve [22] by using the elements of S_K as data points. The degree of the polynomial can be selected depending on the length of the sampling time frame and the maximum Doppler shift. Afterwards a sequence of length L, s_K, is formed by both transceivers by sampling their respective least-square curves at the points 1, 2, …, L. Each transceiver generates a bit string ρ_K of length L by comparing each element of s_K with its respective element of g_K. For each $i \in [0, L]$, if the value of $s_K[i]$ is greater than that of $g_K[i]$, the corresponding element $\rho_K[i]$ of the bit string ρ_K is set equal to 0, otherwise, it is set equal to 1. Thus, the bit strings ρ_A and ρ_B are given by

$$\rho_K[i] = \left\{ \begin{matrix} 1, & g_K[i] \leq s_K[i] \\ 0, & g_K[i] > s_K[i] \end{matrix} \right\}, \quad i = 1, 2, \ldots, L \quad (4)$$

Before the sampling process, a low-pass filter should be used in order to smoothen the sequence g_K and eliminate high-frequency components that can potentially harm the effectiveness of this method.

An example of the proposed thresholding method for a simulated channel magnitude envelope is shown in Fig. 2. Specifically, Fig. 2*a* depicts the perceived channel magnitude at both transceivers and their generated least-square thresholds, whereas Fig. 2*b* depicts the corresponding bit strings ρ_A and ρ_B.

The main advantage of the above proposed thresholding technique is its ability to detect fades of smaller depth compared to a constant threshold. This is useful because, assuming that the eavesdropper knows the number of deep fades in the considered time interval, the time required to

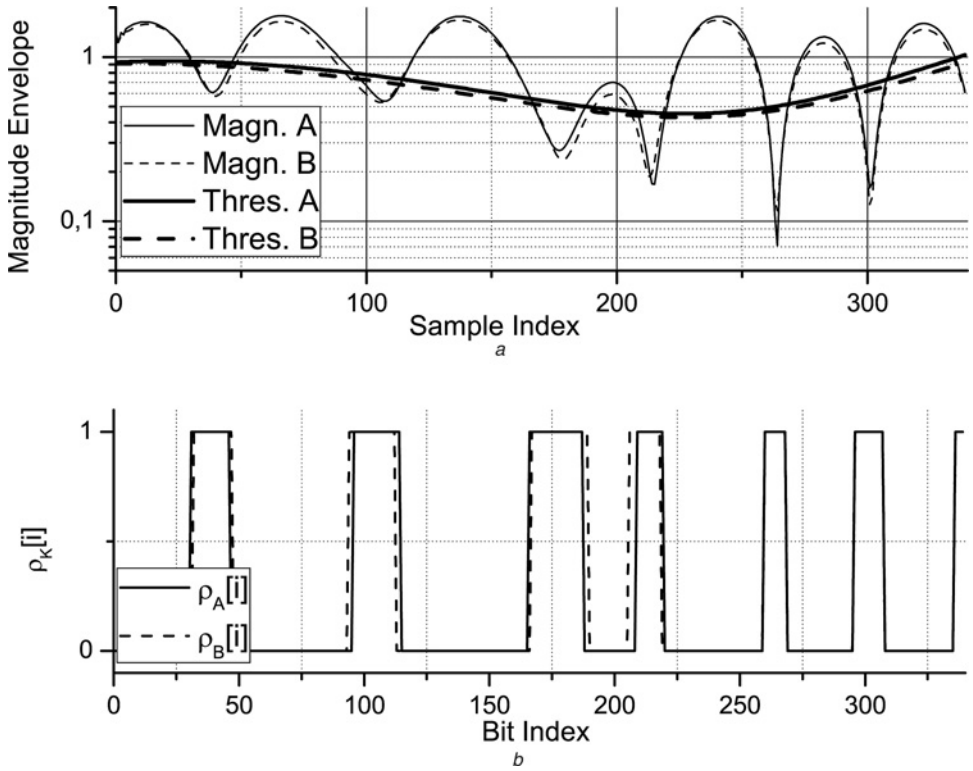

Fig. 2 *Proposed thresholding method for a simulated channel magnitude envelope*

a Channel magnitudes and thresholds for *A* and *B*
b Bit strings generated by *A* and *B*

perform a brute force check on the value of ρ_K increases for a larger number of fades, thus strengthening the system against such attacks. Also, the ability of the legitimate transceivers to detect fades of small depth might cause the eavesdropper to miscalculate the number of fades by ignoring or not being aware of fades with a smaller depth.

4 Neural network-based error reconciliation

Assume that transceivers A and B have generated bit strings ρ_A and ρ_B of length L, respectively. The eavesdropper is oblivious to the fading characteristics of the communication channel between the two legitimate nodes, and thus cannot deduce the values of ρ_A and ρ_B. The correlation between the channels perceived by A and B will always be less than 1. This means that there will be discrepancies between the bit strings generated by the thresholding process. This is clearly shown in Fig. 2, where the two bit strings are not identical, though the channel envelopes are highly correlated. However, if ρ_A and ρ_B are not identical, they cannot be used as cryptographic keys. The method presented in this section uses the two similar bit strings to generate a cryptographic key of arbitrary length, which will be known to both transceivers.

4.1 Neural network description and operation

Let L be the length of ρ_A and ρ_B, ρ the cryptographic key, and L_t the length of ρ, which can be selected arbitrarily based on the desired key length. The neural network that will be used handles binary inputs and outputs and consists of an input layer of L nodes, a hidden layer of N nodes and an output layer of L_t nodes. The value of N is selected by taking into account that higher values increase the complexity, but also the error reconciliation capability of the key exchange scheme.

An overview of the proposed scheme is given in the following.

1. Transceiver A creates a binary neural network with the parameters mentioned above and randomly initialises its synaptic weights.
2. Transceiver A randomly generates the cryptographic key ρ of length L_t.
3. The neural network is trained by using a training set that consists of inputs similar to ρ_A. This is described in Section 4.2.
4. The synaptic weights of the neural network are transmitted to transceiver B.
5. Transceiver B applies ρ_B as the input to the neural network with the received synaptic weights. The neural network generates an output of length L_t which should ideally be identical to ρ.

In Step 2, the cryptographic key is randomly generated. Here, it should be noted that in order to maximise the security of the key, each bit of ρ, which is denoted by $\rho[i]$, $i = 1, 2, \ldots, L_t$, should be generated in a way that ensures $P(\rho[i] = 0) = P(\rho[i] = 1)$. This is done so that for security purposes the generated key is uniformly distributed. Also, in Step 4, to ensure the correct transmission of the required information, the use of an error correction scheme is recommended. To the best of the authors' knowledge, there is no documented way for the eavesdropper to deduce the value of the cryptographic key with the knowledge of only the neural network's synaptic weights.

doi: 10.1049/iet-ifs.2011.0360

4.2 Training of the neural network

First, we define the training set that is used for the training process of the neural network. The motivation behind the construction of the training set is that it should contain bit strings that simulate the kind of errors that appear in practical situations. In [21], an efficient bit string representation was proposed, where each bit string is represented by a set of pairs whose first element denotes the position of the beginning of a fade, and its second element denotes the end of the fade. In this context, a fade means a run of consecutive 1s in a bit string. Bit discrepancies usually occur because of miscalculations of the position of the beginning or the end of a fade. For the reader's convenience, ρ_K – which has previously been defined in (4) – can thus be redefined as

$$\rho_K \equiv \left\{\left(i_l^K, j_l^K\right) | l = 1, \ldots, t\right\} \tag{5}$$

where t denotes the number of fades detected by the thresholding process, which is equal to the number of runs of consecutive 1s in ρ_K. Let

$$h_l(s, k) = \left\{\begin{matrix} s, & k = l \\ 0, & k \neq l \end{matrix}\right\}, l, k \in \{1, \ldots, t\}, s \in \{-M, \ldots, M\} \tag{6}$$

Given the bit string ρ_A and using the notation described above, we define the strings $T_1^{s,k}$ and $T_2^{s,k}$ as

$$T_1^{s,k} = \left\{\left(i_l^A + h_l(s, k), j_l^A\right) | \left(i_l^A, j_l^A\right) \in \rho_A\right\} \tag{7}$$

$$T_2^{s,k} = \left\{\left(i_l^A, j_l^A + h_l(s, k)\right) | \left(i_l^A, j_l^A\right) \in \rho_A\right\} \tag{8}$$

Thus, we define the set of bit strings T, which constitute the neural network's training set, as

$$T = \left\{T_j^{s,k} | s \in \{-M, \ldots, M\}, k \in \{1, \ldots, t\}, j \in \{1, 2\}\right\} \tag{9}$$

where M is a parameter that denotes the maximum shift that is applied to a fade's beginning or end. Practically, this means that the training set consists of strings that are formed by $\rho_A[i]$, modified so that the beginning or the end of one fade is shifted by $|s|$ bits to the left or to the right. The motivation behind this formation of the training set is that, as can be seen in Fig. 2, discrepancies between ρ_A and ρ_B usually occur at the positions of the bit string that correspond to a level crossing from the thresholding process at the beginning or end of a deep fade, thus the training set should be formed by inputs that simulate such errors. Hence, the training set is formed so that the neural network can detect and correct this kind of discrepancies.

When applying an input bit string to the neural network, the set {0, 1} is mapped to the set {–1, 1} in order to perform the necessary calculations. We have two layers of neurons, one for the hidden layer, whose inputs are the nodes of the input layer, and one for the output layer, whose inputs are the nodes of the hidden layer. In order to describe the neural network's training algorithm, we consider a layer with N_1 inputs, N_2 nodes and N_2 outputs. This can refer to either the hidden or the outer layer. The synaptic weight for the jth input of the ith neuron of this layer is denoted by w_{ij}. All weights w_{ij} are initialised randomly to be either –1 or 1, so that $P(w_{ij} = -1) = P(w_{ij} = 1)$. If x_{ij} denotes the jth input of the ith neuron, then its output is

$$\sigma_i = \text{sgn}\left(\sum_{j=1}^{N_1} w_{ij} x_{ij}\right) \tag{10}$$

where sgn is a function which extracts the sign of a real number. It follows from (10) that the possible outputs of a neuron are –1 and 1. For the neural network's training, a Hebbian learning rule [23] is used. Given the bit string $\rho_A[i]$, the neural network's training process as performed by transceiver A, before the synaptic weights are transmitted to B, consists of the following steps:

1. The cryptographic key ρ is generated and the neural network's synaptic weights are initialised.
2. The output layer of the neural network is trained by using a Hebbian rule with ρ_A as an input, as previously mentioned. This process is performed m times.
3. ρ_A is applied as input of the neural network, and the output of the hidden layer, denoted by ρ_h, is calculated.
4. The training set T is formed as in (9) and the neurons of both hidden and output layers are trained by using ρ_h and ρ as target outputs, respectively, as described in Step 2. Each element of T is used m times.

In the process described above, m is a parameter that can be selected based on the time available for the neural network's training. Simulations have revealed that values of $m > 1$ generally improve the performance of the neural network's error reconciliation capabilities.

The security of the proposed method is based on the following: in order to deduce the value of the cryptographic key, the eavesdropper would have to somehow perform a completely accurate reversal of the training process. As previously mentioned, there is no such documented method, and if it were attempted to develop such a method, it would be impeded to a great degree by the randomisation of the initial synaptic weights. Therefore there is no way for the eavesdropper to fully reproduce the original key.

4.3 Complexity analysis

In this section, we analyse the complexity of the training process of the neural network in terms of synaptic weight updates. As outlined in Section 4.2, the training process of the neural network is essentially performed in two stages. In the first stage, only the output layer is trained. By multiplying the number of elements of the hidden layer N, the number of elements of the output layer L_t, and the size of the training set m, we obtain the number of weight updates in this phase

$$N_1 = mL_tN \tag{11}$$

The number of updates in the second phase is calculated in a similar manner. In this case, the size of the training set is $2t(2M + 1)$. Here, two layers of neurons (output and hidden) are trained, with L_tN and LN weights to be updated, respectively. Therefore the number of weight updates in this phase is

$$N_2 = 2mt\left(L_tN + LN\right)(2M + 1) \tag{12}$$

Table 1 Neural network training complexity

Key length L_t	N_{total}
50	1.0545×10^7
80	1.1364×10^7
100	1.1910×10^7
200	1.464×10^7
210	1.4913×10^7
340	1.8462×10^7

Parameters: $N = 100$, $t = 5$, $M = 4$, $L = 340$, $m = 3$

Therefore the total number of calculations is

$$N_{\text{total}} = N_1 + N_2 = mL_tN + 2mt(L_tN + LN)(2M + 1) \quad (13)$$

The complexity of the above technique is higher than the one proposed in [21], whose performance is compared with our proposed method through simulations in Section 6. This can be concluded by taking into account that both methods involve a channel thresholding process, however the difference in complexity is because of the error reconciliation method. The proposed neural network-based method definitely requires more calculations than [21]. Nonetheless, the simulations performed show that the proposed method has a much higher key agreement rate compared to the method proposed in [21], as shown in Fig. 4. Table 1 shows the number of weight updates during the training process of the neural network for the experimental values used in the simulations presented in Section 6.

5 Masked linear block coding-based error reconciliation

In this section, we present a category of efficient error reconciliation methods based on linear block coding. Bose-Chaudhuri-Hocquenghem (BCH)-based error reconciliation has been examined in [21] and has been applied in quantum key distribution in [24]. However, the proposed method adds a layer of protection in order to strengthen the security of the method against an eavesdropper attack. In these methods, the discrepancies between ρ_B and ρ_A are considered as errors, and an error correction coding method is used in order to correct these discrepancies.

5.1 Error reconciliation algorithm

As in the neural network-based reconciliation method, we assume that transceivers A and B have generated the bit strings ρ_A and ρ_B, respectively, and they both have a length of L bits. Furthermore, we consider a linear block coding scheme, such as a BCH code, where the generator matrix is in standard form. Therefore with this coding scheme, a message of L bits is encoded to a length n codeword, which consists of L bits identical to the original message and $n–L$ parity bits. We also assume that this code can correct up to t errors. The proposed error reconciliation method is based on the idea that the bit discrepancies between ρ_A and ρ_B can be considered as errors, and if A encodes ρ_A with a linear block coding method, B can decode it by knowing only ρ_B and the $n–L$ parity bits, given that there are no more than t bit discrepancies.

Also, in order to generate a uniformly distributed cryptographic key, we define a universal hash family $\{U_k\}_{k\in\{0,1\}^m}$ with range $\{0, 1\}^{L_t}$, where L_t is the desired length of the cryptographic key, as in the work of Azimi-Sadjadi *et al.* [21]. This means that a specific bit string k of length m characterises a function $U_k(\cdot)$ that belongs to the universal hash family, and hashes the input bit string by producing a uniformly distributed output bit string of length L_t. This function will ultimately produce the cryptographic key. We introduce this function because, for security purposes, we require that the key is uniformly distributed. We will also define the average run duration (ARD) of a string as the average length of groups of consecutive 1s, which we refer to as runs. This method also requires a pseudo-random bit sequence generator $H_{s_0}(L)$, where s_0 is the seed and L is the length of the generator's output. Given a seed s, the generator outputs a bit sequence of length L.

This method's security is based on the fact that it is hard to deduce the original message given only the $n–L$ parity bits, however, it is theoretically possible if the appropriate computational power is available. In order to strengthen the security of this method, we introduce a bit masking scheme which can increase the method's security, which will be demonstrated in the following. The key exchange algorithm consists of the following steps:

1. Transceiver A randomly generates a bit string k of length m, which defines a function $U_k(\cdot)$ that belongs to the previously mentioned universal hash family $\{U_k\}_{k\in\{0,1\}^m}$ with range $\{0, 1\}^{L_t}$, and transmits k to B.
2. Transceiver A calculates the ARD of ρ_A. By using the pseudo-random bit sequence generator $H_{s_0}(L)$, a sequence is generated with $s_0 = \text{ARD}_A$ and length equal to L. Then, the bitwise XOR $\rho'_A = \rho_A \oplus H_{\text{ARD}_A}(L)$ is calculated.
3. Transceiver A encodes ρ'_A to a length n codeword by using the previously mentioned linear block coding scheme.
4. The $n–k$ parity bits of the codeword are transmitted to transceiver B.
5. Transceiver B calculates the ARD of ρ_B, denoted as ARD_B. We assume that $|\text{ARD}_A - \text{ARD}_B| \le v$. In order to recover the value of ρ'_A, a set S of bit strings is formed

$$S = \{\rho_B \oplus H_w(L),\ w \in [\text{ARD}_B - v,\ \text{ARD}_B + v]\} \quad (14)$$

The value of v can be selected taking under consideration that larger values of v increases the time required for B to recover the original message, but cover more possible values of $|\text{ARD}_A - \text{ARD}_B|$.
6. Transceiver B appends the $n–L$ parity bits to each of the members of S and an appropriate decoding algorithm is applied on each of these bit strings. This will only be possible in the case where $w = \text{ARD}_A$, since the outputs of $H_{s_0}(L)$ are uncorrelated for different values of s_0. The decoding process results in a bit string ρ, which should be equal to ρ'_A.
7. The cryptographic key is generated by applying $U_k(\cdot)$ to ρ and ρ'_A. Specifically, since ideally ρ and ρ_A are identical, $U_k(\rho)$ and $U_k(\rho_A)$ are also identical, and are used as the cryptographic key.

The construction of the linear block code used in this algorithm and the decoding algorithm are not examined in this work, since the use of specific algorithms is not

doi: 10.1049/iet-ifs.2011.0360

required in this method, and they can be selected based on the requirements of the implementation. This is also true for the pseudo-random sequence generator. For example a Blum Blum Shub generator [25] could be used for this purpose. It should be noted that the exact computational complexity cannot be calculated in this case, since it depends on the selection of these algorithms.

At this point, we provide a justification why this method provides stronger security compared to unmasked BCH error reconciliation. In order to deduce the value of the key, the eavesdropper would have to deduce the value of ρ_A by using only the $n–k$ parity bits by checking words that, when BCH encoded, would give the $n–k$ parity bits as a result. The eavesdropper does not have any knowledge about the ARD of ρ_A and ρ_B, since channel sampling is performed over a relatively small time period on a slow fading channel and the eavesdropper will not be able to deduce the value of the ARD based on the channel's statistical properties. However, in [4] it is assumed that the number t of deep fades might be known to the eavesdropper, but not their location or length.

Therefore assuming that the number t of deep fades but not their location or length are known to the eavesdropper, then the eavesdropper would only check for bit strings that fit this criterion, specifically strings that consist of t runs of ones. Our improvement eliminates this requirement, since with the addition of the mask which is produced by using a hash function with a uniformly distributed output, ρ_A' would not fit these criteria, contrary to ρ_A, which is used in [4]. Therefore the eavesdropper would have to brute force check a much larger number of strings (essentially all bit strings with length L are candidates) and afterwards remove all possible values of the mask that could have been applied on ρ_A in order to find one that fits the aforementioned criterion. Therefore our method provides a significant improvement in security.

6 Simulations and discussion

In this section, we present simulation results for the performance of the proposed key exchange scheme. We first consider the neural network-based error reconciliation method, since it is evident that simulations are required in order to fully understand its properties. For the simulations, the parameters were chosen to have practically reasonable values. Specifically, the maximum Doppler shift was $f_d = 1.54$ Hz and the sampling frequency was $f_s = 100$ Hz. The bit strings ρ_A and ρ_B had a length of 340 bits, the repetition parameter was $m = 3$, and the maximum shift parameter was $M = 4$. We also assumed that the synaptic weights were equal at A and B, meaning that there were no transmission errors.

Fig. 3 depicts the average level-crossing rate (LCR) for the proposed least-square thresholding method, plotted against the degree of the polynomial, calculated with a scaling factor of $u = 1$. The simulations were performed for two maximum Doppler shift frequencies: 1.54 and 2.22 Hz. Rayleigh and Rician fading channels were simulated for each of these frequencies. The K-factor for the Rician channels was $K = 10$ dB. It should be noted that a zero degree polynomial represents a constant threshold which was used in the past. We observe that in all cases, a polynomial of a relatively small degree greater than zero can lead to an increase of the LCR. This applies even to the LoS scenarios, modelled here by Rician fading. We can

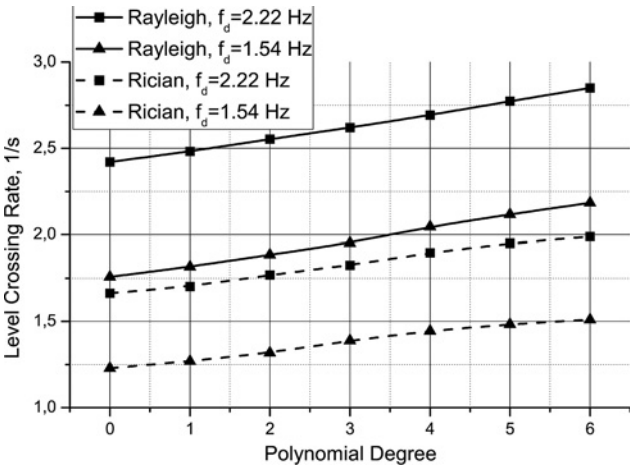

Fig. 3 *Level crossing rate for Rayleigh and Rician channels plotted against the polynomial degree for the proposed thresholding method*

thus conclude that the proposed thresholding technique has better fade detection capabilities than the one which uses a constant threshold. As previously mentioned, this leads to an increase of the security level against eavesdropper activity.

Fig. 4 depicts the key agreement percentage after the error reconciliation process, plotted against the number of bit discrepancies between ρ_A and ρ_B, rounded up to the nearest even number. The simulations were performed for three key lengths: 50, 100 and 200 bits, and the scaling factor was $u = 0.7$. We observe that in all cases, the success rate drops as the number of errors increase. In all cases, we achieve a high key agreement rate, which remains above 90% for up to 8 bit discrepancies. We also observe that the success rate decreases as the key length increases, which is visible especially for larger numbers of errors. This indicates that smaller key lengths lead to better error reconciliation capabilities. Another error reconciliation method proposed in [21] was also tested, named 'SFIR Construction #2'. The corresponding key agreement percentages are also depicted in Fig. 4. We observe that the proposed neural

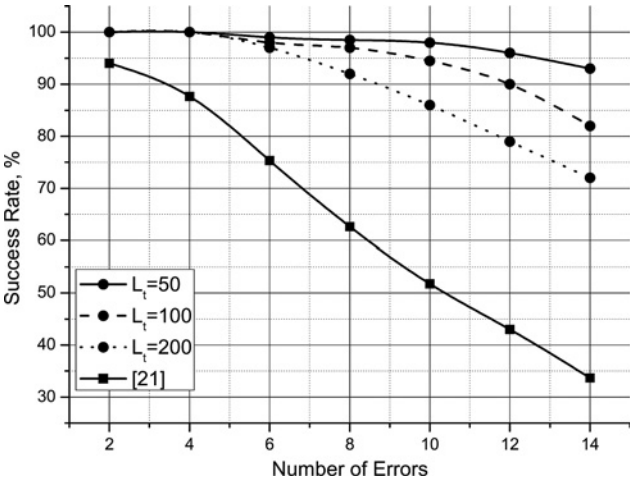

Fig. 4 *Key agreement success rate for the proposed method and a method from the literature plotted against the number of bit errors for three different key lengths*

network-based method offers much higher agreement rates in all cases. It should be noted that in SFIR Construction #2, the key length does not affect the scheme's error reconciliation capability, so only one graph is presented for this method. We also note that even in the case of faulty key exchange, this would not mean that the exchange would not be secure, but that the legitimate transceivers would not have agreed on a value for the cryptographic key, without any compromise in the method's security. A retransmission scheme could be implemented so that the process is repeated in case of faulty exchange.

In order to test the method's security level, a simulation was conducted where transceivers A and B applied the proposed key exchange scheme, while an eavesdropper who has intercepted the neural network's synaptic weights performs a brute force check by testing various inputs to the neural network. Three key lengths were tested in order to examine the security of the scheme in relation to the key length. The lengths examined were 80, 210 and 340 bits. In each of these cases the eavesdropper performed $3 \cdot 10^4$ checks. Fig. 5 depicts the number of outputs with a specific Hamming distance from the actual cryptographic key (unknown to the eavesdropper) plotted against the Hamming distance for each of the three key lengths. We observe that the Hamming distance from the actual key follows a near-Gaussian distribution with a mean value equal to half of the key's length. This demonstrates that the eavesdropper cannot extract any useful information from the neural network's outputs. It should also be noted that the eavesdropper has no way to determine the Hamming distance of the neural network's outputs from the actual key, or whether an output is equal to the actual key, indicating that the proposed key exchange method is information-theoretically secure. If, however, we have to minimise the probability that the neural network intercepted by the eavesdropper will generate the actual key as an output, it is best to choose a large key length. This is because the quotient of the standard deviation of the Gaussian curve, σ, and the key length, L_t, σ / L_t, decreases as the key length increases. This means that the Hamming distance of 0 is relatively farther from the center of the distribution for larger key lengths.

One of the most important advantages of the proposed neural network-based key exchange method is its customisability. For example in a noisy channel where there might be a large number of discrepancies between ρ_A and ρ_B, it is possible to increase the error reconciliation capabilities of the neural network by increasing the size of

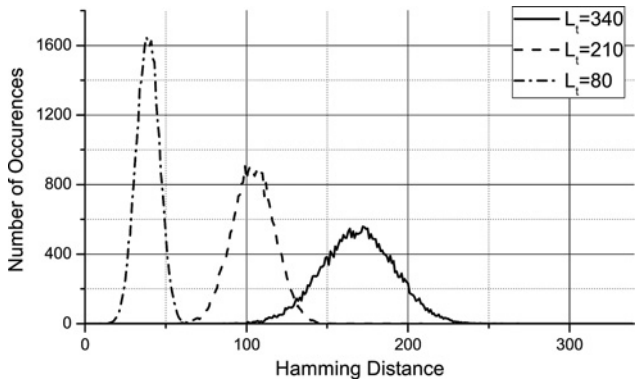

Fig. 5 *Hamming distance of the neural network's output from the actual cryptographic key during a brute force check*

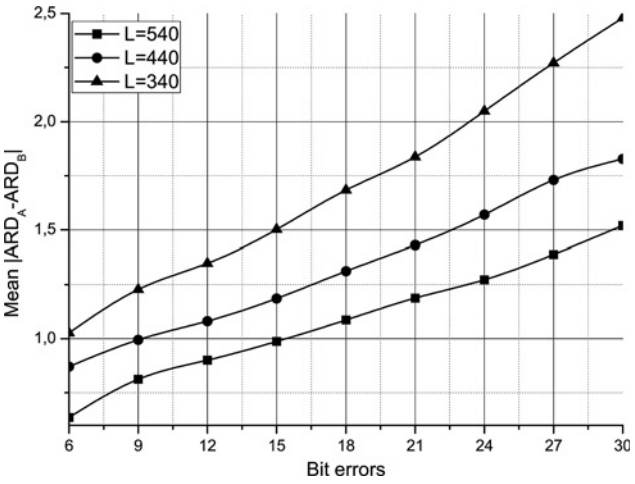

Fig. 6 *Average $|ARD_A–ARD_B|$ plotted against the number of bit errors, for three key lengths*

the training set or the number of neurons in the hidden layer. However, in this case, the neural network's training process will require more time. Thus, there is a trade-off between these two aspects of the key exchange method's operation. The simulations performed also revealed some interesting observations about the key length. Specifically, an increase in the key length produces more secure keys, but might reduce, to some extent, the key agreement probability. A trade-off can also be made between these two factors.

Simulations were also conducted for the linear block coding-based error reconciliation method. Specifically, for three key lengths (340, 440 and 540 bits) and with the parameters noted above, simulations were conducted in order to estimate the appropriate value of parameter v in the 5th step of the error reconciliation algorithm described in Section 5.1. In our implementation, the linear block code used was a (n, k) BCH code, and a Blum–Blum–Shub random sequence generator was used in order to generate the bit masks.

Fig. 6 depicts the average value of $|\text{ARD}_A - \text{ARD}_B|$, plotted against the number of bit discrepancies between ρ_A and ρ_B, rounded up to the nearest even number. The simulations were performed for three lengths of ρ_A and ρ_B: 340, 440 and 540 bits, and the scaling factor was $u = 0.7$. We observe that the average difference between the values of ARD perceived by A and B increases as the number of bit discrepancies increases. Also, as the key length increases, the average ARD difference decreases. This is because for larger lengths, the thresholding process detects a larger number of fades, so the same number of bit discrepancies affects the average ARD less than for a shorter ρ_A with fewer runs of ones. We can use this knowledge in order to select an appropriate value for v depending on the parameters of the implementation, considering that larger values of v will increase the success rate of the method, but also the time required to perform the decoding of the message.

7 Conclusions

We have introduced a novel PHY-layer key exchange algorithm for wireless communications and demonstrated its performance and security level. We presented a least-square-based channel thresholding method in order to extract a bit string from the channel's fading characteristics.

doi: 10.1049/iet-ifs.2011.0360

Next, the concept behind neural network-based error reconciliation was presented, and its specific characteristics, as well as the proposed training process, were described. Finally, a linear block coding error reconciliation method with emphasis on security was also proposed.

Simulations have shown that the techniques proposed in this work have a multitude of benefits, such as efficiency, security and customisability. They are also able to function well in noisy channels where discrepancies appear between the bit strings generated by the two transceivers. Future work will address authentication issues and explore the capability of neural networks to perform user verification. Also, the key exchange algorithm's performance will be tested in various practical settings, such as indoor networks and communication between handheld devices.

8 Acknowledgment

This paper was presented in part at IEEE PIMRC 2011, Toronto, Canada, 11–14 September 2011.

9 References

1 Diffie, W., Hellman, M.E.: 'New directions in cryptography', *IEEE Trans. Inf. Theory*, 1976, **22**, (6), pp. 644–654
2 Bloch, M., Barros, J.: 'Physical-layer security' (Cambridge University Press, 2011, 1st ed.)
3 Hershey, J., Hassan, A., Yarlagadda, R.: 'Unconventional cryptographic keying variable management', *IEEE Trans. Commun.*, 1995, **43**, (1), pp. 3–6
4 Azimi-Sadjadi, B., Kiayias, A., Mercado, A., Yener, B.: 'Robust key generation from signal envelopes in wireless networks'. ACM Conf. on Comput. and Commun. Security'07, 2007, pp. 401–410
5 Mathur, S., Ye, N.M.C., Reznik, A.: 'Radio-telepathy: extracting a secret key from an unauthenticated wireless channel'. MobiCom'08, 2008, pp. 128–139
6 Xiao, L., Greenstein, L., Mandayam, N., Trappe, W.: 'Using the physical layer for wireless authentication in time-variant channels', *IEEE Trans. Wirel. Commun.*, 2008, **7**, (7), pp. 2571–2579
7 Yu, P., Baras, J., Sadler, B.: 'Physical-layer authentication', *IEEE Trans. Inf. Forensics Sec.*, 2008, **3**, (1), pp. 38–51
8 Patwari, N., Croft, J., Jana, S., Kasera, S.K.: 'High-rate uncorrelated bit extraction for shared secret key generation from channel measurements', *IEEE Trans. Mob. Comput.*, 2010, **9**, (1), pp. 17–30
9 Croft, J., Patwari, N., Kasera, S.K.: 'Robust uncorrelated bit extraction methodologies for wireless sensors'. Proc. Ninth ACM/IEEE Int. Conf. on Information Processing in Sensor Networks, series IPSN'10, 2010, pp. 70–81
10 Zeng, K., Wu, D., Chan, A., Mohapatra, P.: 'Exploiting multiple-antenna diversity for shared secret key generation in wireless networks'. Proc. 29th Conf. on Inf. Communication, ser. INFOCOM'10, 2010, pp. 1837–1845
11 Tsouri, G.R., Wulich, D.: 'Securing OFDM over wireless time-varying channels using subcarrier overloading with joint signal constellations', *EURASIP J. Wirel. Commun. Netw.*, 2009, **2009**, pp. 6:1–6:18
12 Wilhelm, M., Martinovic, I., Schmitt, J.B.: 'Key generation in wireless sensor networks based on frequency-selective channels – design, implementation, and analysis', *CoRR*, 2010, http://arxiv.org/abs/1005.0712
13 Hamida, S.T.-B., Pierrot, J.-B., Castelluccia, C.: 'An adaptive quantization algorithm for secret key generation using radio channel measurements'. Proc. Third Int. Conf. on New Technologies, Mobility and Security, series NTMS'09, 2009, pp. 59–63
14 Wilhelm, M., Martinovic, I., Schmitt, J.B.: 'On key agreement in wireless sensor networks based on radio transmission properties'. Fifth IEEE Workshop on Secure Netw. Protocols, 2009. NPSec 2009., October 2009, pp. 37–42
15 Forman, M., Young, D.: 'The generation of shared cryptographic keys through half duplex channel impulse response estimation at 60 GHz'. 2010 Int. Conf. on Electromagnetics in Advanced Applications (ICEAA), 2010, pp. 627–630
16 Jana, S., Premnath, S.N., Clark, M., Kasera, S.K., Patwari, N., Krishnamurthy, S.V.: 'On the effectiveness of secret key extraction from wireless signal strength in real environments'. Proc. 15th Annual Int. Conf. on Mobile Computation and Network, Series MobiCom'09, 2009, pp. 321–332
17 Di Renzo, M., Debbah, M.: 'Wireless physical-layer security: the challenges ahead'. Int. Conf. on Advanced Technologies for Commun., 2009 (ATC'09), 2009, pp. 313–316
18 Mathur, S., Reznik, A., Ye, C., Mukherjee, R., Rahman, A., Shah, Y., Trappe, W., Mandayam, N.: 'Exploiting the physical layer for enhanced security', *IEEE Wirel. Commun.*, 2010, **17**, (5), pp. 63–70
19 Zafer, M., Agrawal, D., Srivatsa, M.: 'A note on information-theoretic secret key exchange over wireless channels'. 47th Annual Allerton Conf. on Communication, Control, and Computation, 2009, 30 September 2009–2 October 2009
20 Martinovic, I., Pichota, P., Schmitt, J.B.: 'Jamming for good: a fresh approach to authentic communication in WSNs'. Proc. Second ACM Conf. on Wireless Network Security, New York, NY, USA, 2009, pp. 754–761
21 Azimi-Sadjadi, B., Kiayias, A., Mercado, A., Yener, B.: 'Secret communication over fading channels', in Ruoheng, L., Wade, T. (Ed.): 'Securing Wireless Communications at the Physical Layer' (Springer Publishing Company Incorporated, 2009, 1st edn.)
22 Hildebrand, F.B.: 'Introduction to numerical analysis' (McGraw-Hill, New York, 1973, 2nd ed.)
23 Haykin, S.: 'Neural networks: a comprehensive foundation' (Prentice-Hall, 1999)
24 Traisilanun, W., Sripimanwat, K., Sangaroon, O.: 'Secret key reconciliation using BCH code in quantum key distribution'. Communications and Information Technologies, 2007. ISCIT'07. Int. Symp., October 2007, pp. 1482–1485
25 Blum, L., Blum, M., Shub, M.: 'A simple unpredictable pseudo-random number generator', *SIAM J. Comput.*, 1986, **15**, pp. 364–383