We are IntechOpen, the world's leading publisher of Open Access books Built by scientists, for scientists

6,900
Open access books available

186,000
International authors and editors

200M
Downloads

Our authors are among the

154
Countries delivered to

TOP 1%
most cited scientists

12.2%
Contributors from top 500 universities

CLARIVATE ANALYTICS
BOOK CITATION INDEX
INDEXED

WEB OF SCIENCE™

Selection of our books indexed in the Book Citation Index
in Web of Science™ Core Collection (BKCI)

Interested in publishing with us?
Contact book.department@intechopen.com

Numbers displayed above are based on latest data collected.
For more information visit www.intechopen.com

A New Cross-Layer FPGA-Based Security Scheme for Wireless Networks

Michael Ekonde Sone

Abstract

This chapter presents a new cross-layer security scheme which deploys efficient coding techniques in the physical layer in an upper layer classical cryptographic protocol system. The rationale in designing the new scheme is to enhance security-throughput trade-off in wireless networks which is in contrast to existing schemes which either enhances security at the detriment of data throughput or vice versa. The new scheme is implemented using the residue number system (RNS), non-linear convolutional coding and subband coding at the physical layer and RSA cryptography at the upper layers. The RNS reduces the huge data obtained from RSA cryptography into small parallel data. To increase the security level, iterated wavelet-based subband coding splits the ciphertext into different levels of decomposition. At subsequent levels of decomposition, the ciphertext from the preceding level serves as data for encryption using convolutional codes. In addition, throughput is enhanced by transmitting small parallel data and the bit error correction capability of non-linear convolutional code. It is shown that, various passive and active attacks common to wireless networks could be circumvented. An FPGA implementation applied to CDMA could fit into a single Virtex-4 FPGA due to small parallel data sizes employed.

Keywords: residue number system (RNS), RSA cryptography, field programmable gate array (FPGA), subband coding, convolutional coding, CDMA

1. Introduction

Generally, wireless networks consist of low capacity links with nodes that rely on batteries. An efficient communication scheme for such networks should minimize both congestion in the links and control information in the nodes. Security is a critical parameter in wireless applications and any efficient communication scheme has to integrate security vulnerabilities of the system in its implementation. Unfortunately, existing schemes have network security implemented at the upper layer such as the application layer; meanwhile parameters such as congestion, which affect data throughput, are the physical layer. Hence, any attempt to increase the security level in a communication system greatly compromises data throughput. In [1], the authors developed a metric to estimate a timeframe for cyberattacks using the RSA public key cryptography. In the analysis, the authors estimated the attacker's human time in carrying out a successful attack based on the key length.

Such an implementation at the upper layer will curb any security attack at the prescribed time but will greatly compromise data throughput at the physical layer due to the huge modular exponentiation involved in its implementation. In [2], the authors developed a secure and efficient method for mutual authentication and key agreement protocol with smart cards. The implementation, which is based on the constant updating of the password, will involve a considerable amount of control information, which is detrimental to the optimum functioning of the nodes. Research work using different information-theoretic models to develop physical layer security based on the characteristics of the wireless links has been carried out [3, 4]. However, the existing methods for implementing physical layer security under the different information-theoretic security models is expensive and requires assumptions about the communication channels that may not be accurate in practice [5]. Hence any deployment of the physical-layer security protocol to supplement a well-established upper layer security scheme will be a pragmatic approach for robust data transmission and confidentiality [6]. It is in this light that, a new cross-layer approach is presented in this research. Major research efforts have targeted cross-layer implementation of security schemes in wireless networks [7–9]. In this research, the proposed cross-layer security scheme uses signal processing techniques as well as efficient coding and well-established cryptographic algorithm to implement a security scheme, which greatly enhances security-throughput trade-off, and curb many security threats common to wireless networks.

The rest of the chapter is organized in eight subsequent sections. In Section 2, we present the background knowledge required for the design and implementation of the new cross layer security scheme. This will involve a review of the different techniques used in the development of the new security scheme. The first subsection presents the implementation of the multi-level convolutional cryptosystem. This implementation involves the combination of subband coding and a new non-linear convolutional coding. Next, a review of the residue number system (RNS) with brief description of the Chinese remainder theorem (CRT) is presented. We conclude the section with an overview of RSA public-key cryptography. Section 3 presents the protocol for the implementation of the new cross layer security scheme. The FPGA-based implementation applied to CDMA using the new layered security scheme will be presented in Section 4. In Section 5, cryptanalysis of the cross-layer security scheme will be carried out in order to quantify the security. Quantification of data throughput is performed in section 6 while different security threats which could be circumvented by the cross-layer security scheme are presented in Section 7. We end the chapter in Section 8 with conclusions of our work.

2. Background

This section presents the background knowledge required for the design and implementation of the new cross layer security scheme. It involves a review of the different techniques used in the development of the new security scheme.

2.1 Multi-level convolutional cryptosystem

The multi-level convolutional cryptosystem constitutes the second stage of implementation at the physical layer. It receives integers from the RNS implemented at the first stage. The multi-level cryptosystem is implemented using subband coding and non-linear convolutional cryptosystem.

2.1.1 Subband coding

The integers from the RNS block are split into different levels of decomposition based on subband coding. Subband coding is implemented using integer wavelet lifting scheme [10, 11]. It is shown in [12] that, a judicious choice of filter banks could result into an integer transform despite the fact that, wavelet transform is an approximation process. A four-tap Daubechies polyphase matrix, which results into integer transforms, is given as follows [12]:

$$P^{new}(z) = \begin{bmatrix} h_e(z) & g_e^{new}(z) \\ h_o(z) & g_o^{new}(z) \end{bmatrix} \tag{1}$$

where h and g are filter coefficients with suffix e and o denoting even and odd coefficients. The factorization of the polyphase matrix is as follows [12–14].

$$P(z) = \tilde{P}(z) = \begin{bmatrix} 1 & -\sqrt{3} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ \frac{\sqrt{3}}{4} + \frac{\sqrt{3}-2}{4}z^{-1} & 1 \end{bmatrix} \begin{bmatrix} 1 & z \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \frac{\sqrt{3}+1}{\sqrt{2}} & 0 \\ 0 & \frac{\sqrt{3}-1}{\sqrt{2}} \end{bmatrix} \tag{2}$$

(Eq. (2)) forms the basis of integer to integer wavelet transform which in effect is progressive transmission. The factored coefficients $h_1 = -\sqrt{3}$, $h_2 = \frac{\sqrt{3}}{4}$, $h_3 = \frac{\sqrt{3}-2}{4}$, $h_4 = \frac{\sqrt{3}+1}{\sqrt{2}}$, $h_5 = \frac{\sqrt{3}-1}{\sqrt{2}}$ are used to compute the approximate and detail sequences. In such transmission, the original data is split into two portions, namely approximate and detail sequences. The detail sequence is transmitted while the approximate sequence is further split into two halves. The process is repeated until the final data point. Using (Eq. (2)), the approximate, a_n and the detail, d_n sequences could be computed as follows [12, 15]:

$$|a_n|_{m_j} = \left| \left| \left| |x_{2l} h_1|_{m_j} + x_{2l-1} \right|_{m_j} \cdot h_2 \right|_{m_j} + x_{2l-1} \right|_{m_j}$$

$$|d_n|_{m_j} = \left| \left| |x_{2l} h_1|_{m_j} + x_{2l-1} \right|_{m_j} + a_{n-1} \right|_{m_j} \tag{3}$$

where a_n and d_n are the approximation and detail sequences of wavelet coefficients of the nth sample. The subsequent transmissions are fed into the non-linear convolutional coding block as depicted in **Figure 1** for the first level kth detail and
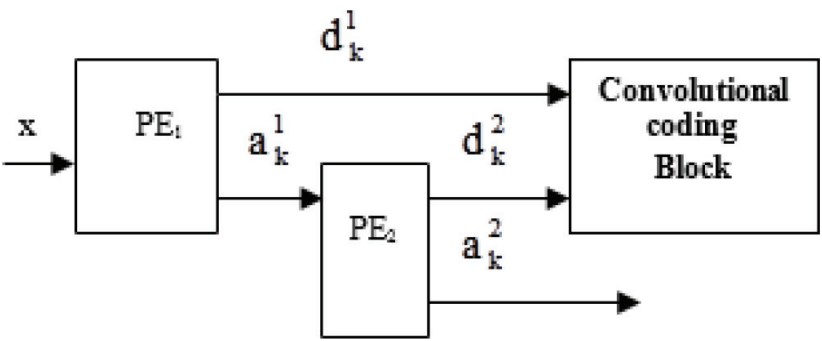

Figure 1.
Synopsis for the computation of the kth coefficients for the 1st and 2nd levels of decomposition.
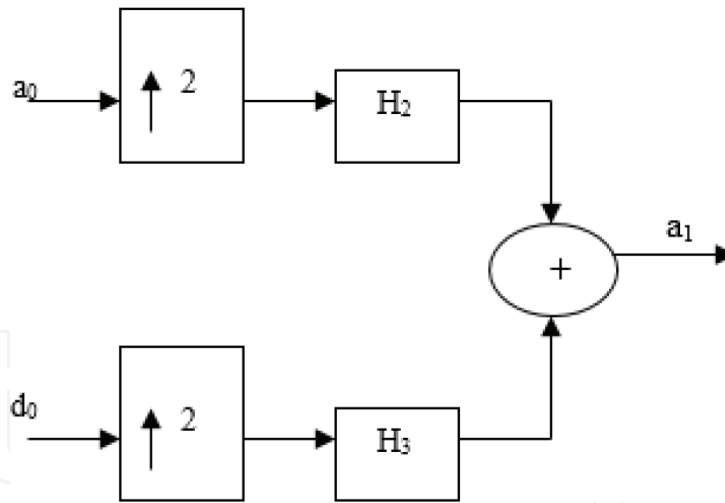
Figure 2.
The first stage of the inverse transform.

approximation sequences [12, 15]. The processing blocks (PEs) shown in the figure depicts the computations of (Eq. (3)).

The final decomposition level which comprises one data point will give one approximation coefficient and one detail coefficient.

At the destination, the inverse wavelet transform is performed to obtain the successive approximation sequences. (Eq. (3)) is used to perform the inverse trans-form by reversing the operations for the forward transform and flipping signs [12, 15]. The process starts with the approximation and detail coefficients a_0 and d_0 respectively obtained at the final decomposition level of the forward transform. The first stage of the inverse transform is shown in **Figure 2** [12, 15].

The ($\uparrow 2$) symbol represents upsampling by 2, which means that zeros are inserted between samples while H_2 and H_3 are the filter coefficients used in (Eq. (3)).

2.1.2 Nonlinear convolutional cryptosystem

A major advantage of symmetric cryptography is the ability of composing prim-itives to produce stronger ciphers although on their own the primitives will be weak. Hence, the vulnerable convolutional code block will be cascaded into differ-ent stages using the product ciphers obtained from the S-box and P-box to form a non-linear convolutional cryptosystem.

- **Key generation:** The specifications of the private keys used in the implementation of the cascaded convolutional cryptosystem are as follows [12, 15, 16]:

    a. States of each transducer or convolutional code block in the cascade given by the contents of the sub-matrices in the generator matrix;

    b. The transition functions. These are mappings used to compare the input bits and present state and switches to the appropriate next state;

    c. n-bit S-boxes. They are used to shuffle the output bits.

    d. n-bit P-boxes. They are used for the different permutations per level of decomposition.

For illustrative purposes, an (8, 8, 2) convolutional encoder will be considered to demonstrate the keys generation process.

2.1.2.1 States of convolutional code block or transducer

For an 8×8 matrix, there are at least 2^{16} ways or keys in which the connections of a register to the modulo-2 adder could be made. A possible key which gives the contents of the 8×8 matrix are shown in (Eq. (4))

$$g^1_{m1,0} = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}, g^1_{m1,1} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}, g^1_{m1,2} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \qquad (4)$$

The generator matrix is used to specify the following set of 8 vectors.

$X(7) := A1(7) \oplus A3(7);$ $X(6) := A1(7) \oplus A1(6) \oplus A3(6).$
$X(5) := A1(5) \oplus A3(5) \oplus A3(6);$ $X(4) := A1(4) \oplus A3(4) \oplus A3(5).$
$X(3) := A1(3) \oplus A3(3) \oplus A3(4);$ $X(2) := A1(2) \oplus A3(2) \oplus A3(3).$
$X(1) := A1(1) \oplus A3(1);$ $X(0) := A1(0) \oplus A3(0).$

It should be recalled that, for an (n,k,L) convolutional encoder, each vector has Lk dimensions and contains the connections of the encoder to the modulo-2 adder.

The structure of the (8, 8, 2) convolutional encoder is shown in **Figure 3** with A2, A3 representing the registers M^1_1, M^1_2 while A1 represents the input data.

2.1.2.2 The transition functions

For an (8,8,2) convolutional code, there are $2^8 = 256$ mappings or keys. There are two sets of transition functions denoted as f_1 for the two possible states.

For example, a transition function that compares input data to state 1 and remains in state 1 is given as follows:

$$f_1(1, \{[00000000], [00000001], [00000010], [00000011], [00000100],$$
$$[00000101], [00000110], [00000111]\}) = \{1\} \qquad (5)$$

In (Eq. (5)), if the input data is any of the sequences $\{[00000000],$ [00000001], [00000010], [00000011], [00000100], [00000101], [00000110], [00000111]\}, the present state of the transducer which is state 1 is retained.
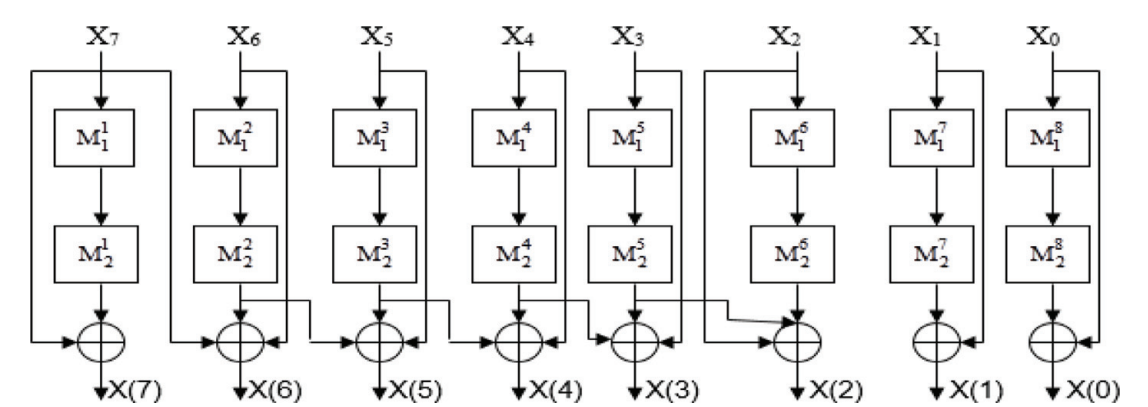

Figure 3.
Structure of an (8, 8, 2) convolutional encoder.

A transition function that compares input data to state 1 and switches to state 2 is given as follows:

$$f_1(1, \{[00001000], [00001001], [00001010], [00001011], [00001100],$$
$$[00001101], [00001110], [00001111]\}) = \{2\}, \tag{6}$$

In (Eq. (6)), if the input data is any of the sequences $\{[00001000], [00001001], [00001010], [00001011], [00001100], [00001101], [00001110], [00001111]\}$, the present state of the transducer which is state 1 is changed to state 2.

At the destination, the transition functions are similar to those for the encoder at the source but change roles. The transition functions are very critical in the implementation of convolutional cryptosystem since it accounts for its dynamic nature, hence an increase in security level.

2.1.2.3 S-box entries

For an (8,8,2) convolutional code, using 2-bit shuffling boxes, there are 16 S-boxes or keys. For higher n-bit shuffling boxes, the number of keys increases, for example 8-bit shuffling boxes will give 2^8 keys. The four 2-bit S-boxes used to illustrate the scheme are shown in **Table 1**. Given an 8-bit data sequence as $[A_7, A_6, A_5, A_4, A_3, A_2, A_1, A_0]$, the look-up S-box, $Sub_{1,1}$ is used to shuffle the first pair of bits, $[A_7, A_6]$, $Sub_{1,2}$ is used to shuffle the second pair $[A_5, A_4]$, $Sub_{1,3}$ is used to shuffle the third pair $[A_3, A_2]$, and $Sub_{1,4}$ is used to shuffle the last pair $[A_1, A_0]$.

2.1.2.4 P-box entries

The interconnections between inputs and outputs are implemented using a permutation set look-up table. For an (8,8,2) code, the eight (08) inputs and outputs could be permuted or interconnected in at least 7^7 = 823,543 ways. A permissible permutation is shown in **Table 2** [12, 15].

Input	00	01	10	11
output	00	11	10	01

$Sub_{1,1}$

Input	00	01	10	11
output	01	00	11	10

$Sub_{1,2}$

Input	00	01	10	11
output	10	01	00	11

$Sub_{1,3}$

Input	00	01	10	11
output	11	10	01	00

$Sub_{1,4}$

Table 1.
2-bit shuffle look-up-table.

Input	1	2	3	4	5	6	7	8
Output	8	7	6	5	4	3	2	1

Table 2.
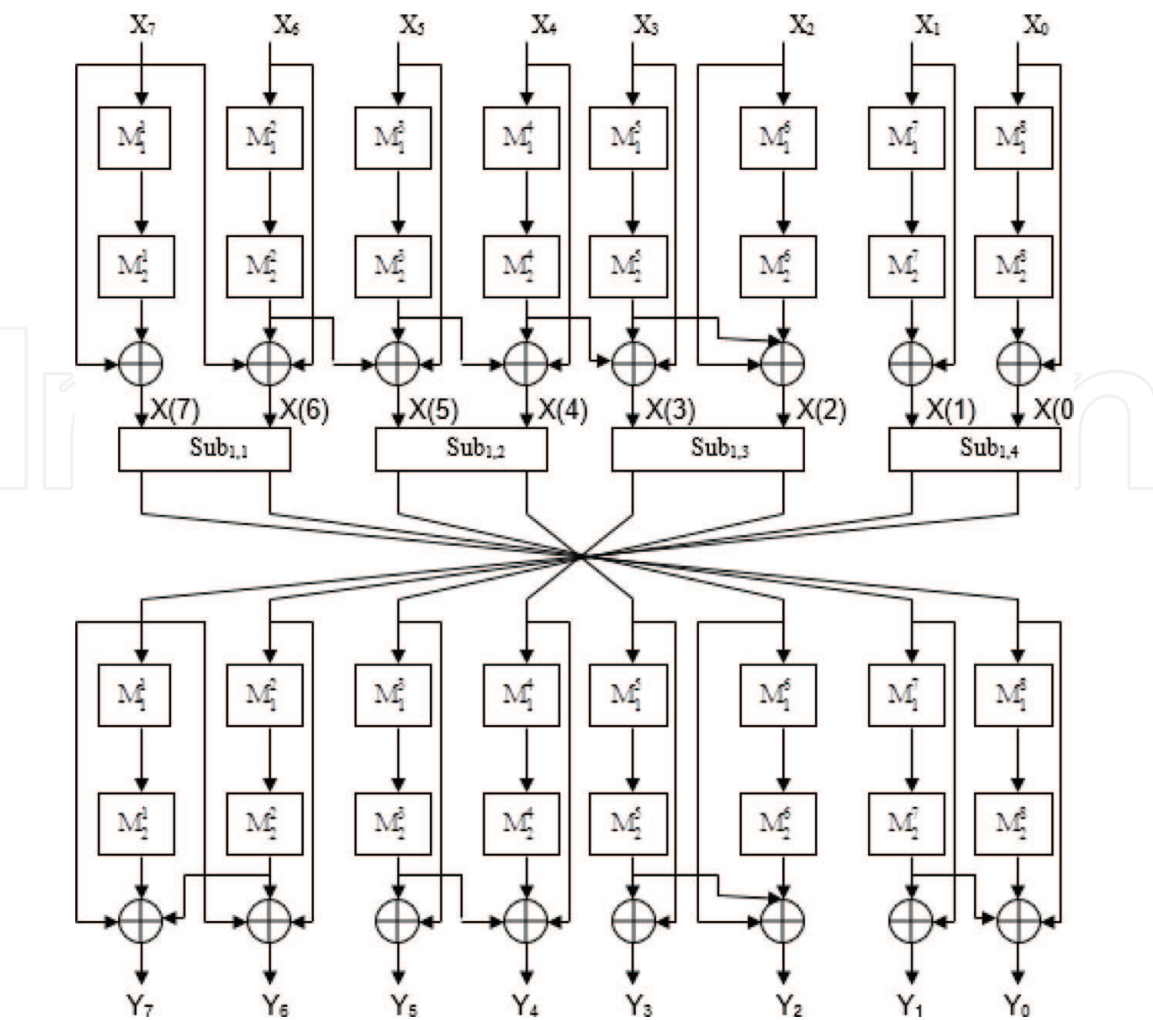*Input–output interconnect look-up-table for encoder.*

Figure 4.
Initial structure of the cascade before encoding.

After the specification of the keys, the vulnerable convolutional code block will be cascaded into different stages using the product ciphers obtained from the S-box and P-box. Using two (02) stages, a non-linear (8, 8, 2) 2-cascaded is as shown in **Figure 4**.

In **Figure 4**, $Sub_{1,1}$, $Sub_{1,2}$, $Sub_{1,3}$ and $Sub_{1,4}$ are S-boxes used for pairwise bit shuffling and the input vector to the first transducer, $\{X_7, X_6, X_5, X_4, X_3, X_2, X_1, X_0\}$ is the output set from the subband encoding block while the output vector $\{Y_7, Y_6, Y_5, Y_4, Y_3, Y_2, Y_1, Y_0\}$ is the ciphertext from the second transducer stage.

It is worth noting that the security level could be greatly increased by increasing the number of stages to be cascaded.

2.2 Residue number system (RNS)

The residue number system uses the Chinese remainder theorem (CRT) to compute unknown values from the remainders left or residues when unknown values are divided by known numbers.

The modular Chinese remainder theorem states that [17, 18]:

Assume m_1, m_2, ..., m_N are positive integers, relatively prime pairs: $(m_i, m_k) = 1$ if i # k. Let $\{b_1, b_2, ..., b_N\}$ be arbitrary integers, then the system of simultaneous linear congruence

$$x_1 = b_1 \ (\text{mod } m_1) \ - - - x_N = b_N \ (\text{mod } m_N) \tag{7}$$

Authentication and non-repudiation	Confidentiality
Choose plaintext X.	Choose plaintext X.
Compute $X_s \equiv X^d \pmod{m}$	Use Bob's public key (m, e) to compute
Send (X, X_s) to Alice. X_s is the RSA digital signature of message, X	$C \equiv X^e \pmod{m}$.
	Send ciphertext, C to Bob

Table 3.
Summary of security services of RSA public-key cryptography.

has exactly one solution modulo the product m_1, m_2, ..., m_N. The solution to the simultaneous linear congruence is formally given as [15, 17, 18].

$$|x|_M = \left| \sum_{j=1}^{L} \hat{m}_j \left| \frac{x_j}{\hat{m}_j} \right|_{m_j} \cdot x_j \right|_M \qquad (8)$$

where $\hat{m}_j = \frac{M}{m_j}$ and $M = \prod_{j=1}^{N} m_j$.

(Eq. (8)) establishes the uniqueness of the solution. In this research, the integers $T_j = |\frac{x_j}{\hat{m}_j}|$ often referred to as the multiplicative inverses of $\hat{m}_j$ are computed a priori by solving the linear congruence in (Eq. (7)). In other words, if $M_j = M/m_j$, then the multiplicative inverses, T_j are computed by solving the congruence $T_j M_j = 1 \pmod{m_j}$.

2.3 RSA public-key cryptography

- Key creation
  - Choose secrete primes p and q and compute m = p.q
  - Choose encryption exponent, e
  - Compute d satisfying e.d $\equiv$ 1 mod ((p – 1). (q – 1)).
  - Public key: (m, e) and Private key: d

- Security services achieved using RSA cryptography are authentication and non-repudiation based on digital signatures and confidentiality based on encryption. Implementation of these services are summarized in **Table 3**

3. Protocol of implementation of cross-layer security scheme

The new scheme is implemented at the application and physical layers. The detail operations of the application and physical layers at the source and destination are as follows:

- Source:
  - Application layer:

    a. Traditional RSA encryption

  - Physical layer:

    a. Step 1: Residue number system (RNS) converts the message points into residues based on the moduli set;

b. Step 2: RNS-based RSA ciphertext is converted into different levels of decomposition using subband coding;

c. Step 3: Symmetric encryption using Convolutional cryptosystem;

At the destination, the entire process is reversed starting with convolutional decoding at the physical layer and ending with RSA decryption at the application layer.

3.1 Illustrative example

Consider an arbitrary array of integers for plaintext as follows {398, 453, 876, 200, 356, 165, 265, 897}.

- Source

  ○ Application layer: Traditional RSA encryption

    a. Primes, p = 13; q = 37 ⇒ n = p. q = 481

    b. Encryption key, e = 5

    c. Decryption key: e. d ≡ 1 mod (432) ⇒ d = 173

Array due to RSA encryption is given as {151, 293, 252, 135, 304, 315, 265, 182}

  ○ Physical layer:

    a. Step 1: Moduli set of {107, 109, 113} is used to convert the RSA ciphertext into 8-bit data point arrays. The residue set, r_1 for m_1 = 107 is as follows:

r_1 = {44, 79, 38, 28, 90, 101, 51, 75}

    b. Step 2: Subband coding is performed to split residues obtained using moduli set into three levels of decomposition since m = 8 = 2^3 data points are used. Subband coding is basically down sampling by 2 using (Eq. (3)). The corresponding arrays for the first level of decomposition are as follows:

r_{11} = {−9, −48, −79, −27}; − r12 = {−9, −42, −75, −21}; − r13 = {−9, −30, −67, −9}

Note that r_{11} refers to first level of decomposition array for modulus, m_1 = 107 and the first element is obtained using (Eq. (3)) with integer lifting filter coefficients set, h = {2, 0, 0} as follows: r_1(1) − 2 × r_1(0) = 79– 2 × 44 = −9.

The same procedure is performed for the second and third levels of decomposition.

    c. Step 3: **Table 4** summarizes the manual computation of the encryption and decryption process of the convolutional cryptosystem for the data r_{11}(0) = −9 from the subband encoding stage based on the entries of the product cipher and combinational logic of the non-linear (8,8,2) 2-cascaded convolutional transducer in **Figure 4**.

- Destination
    - Physical layer:

The entire process is reversed starting with convolutional decoding at the physical layer which was illustrated in **Table 4**.

    a. Subband decoding: It involves upsampling and the use of quadrature mirror filter (QMF) bank as depicted in **Figure 5** [12, 13]

The ($\uparrow 2$) symbol represents upsampling by 2, which means that zeros are inserted between samples. The QMF bank are the coefficients derived from the 4-tap Daubechies filter bank [12, 13, 19]. Hence, using upsampling and the QMF bank coefficients the residue sets r_1, r_2 and r_3 are retrieved.

Figure 5 will be used to perform a numerical illustration of subband decoding for the first level of decomposition of the array of modulus $m_1 = 107$ to obtain approximation coefficients a_1.

The moduli sets obtained from subband encoding for the three levels of decomposition for $m_1 = 107$ are as follows:

- Level 3: $r_{31} = \{2, 44\}$; − Level 2: $r_{21} = \{-50, -20\}$; − Level 1: $r_{11} = \{-9, -48, -79, -27\}$.

For subband decoding, the entire process is reversed with level 3 of encoding becoming level 1 for decoding.

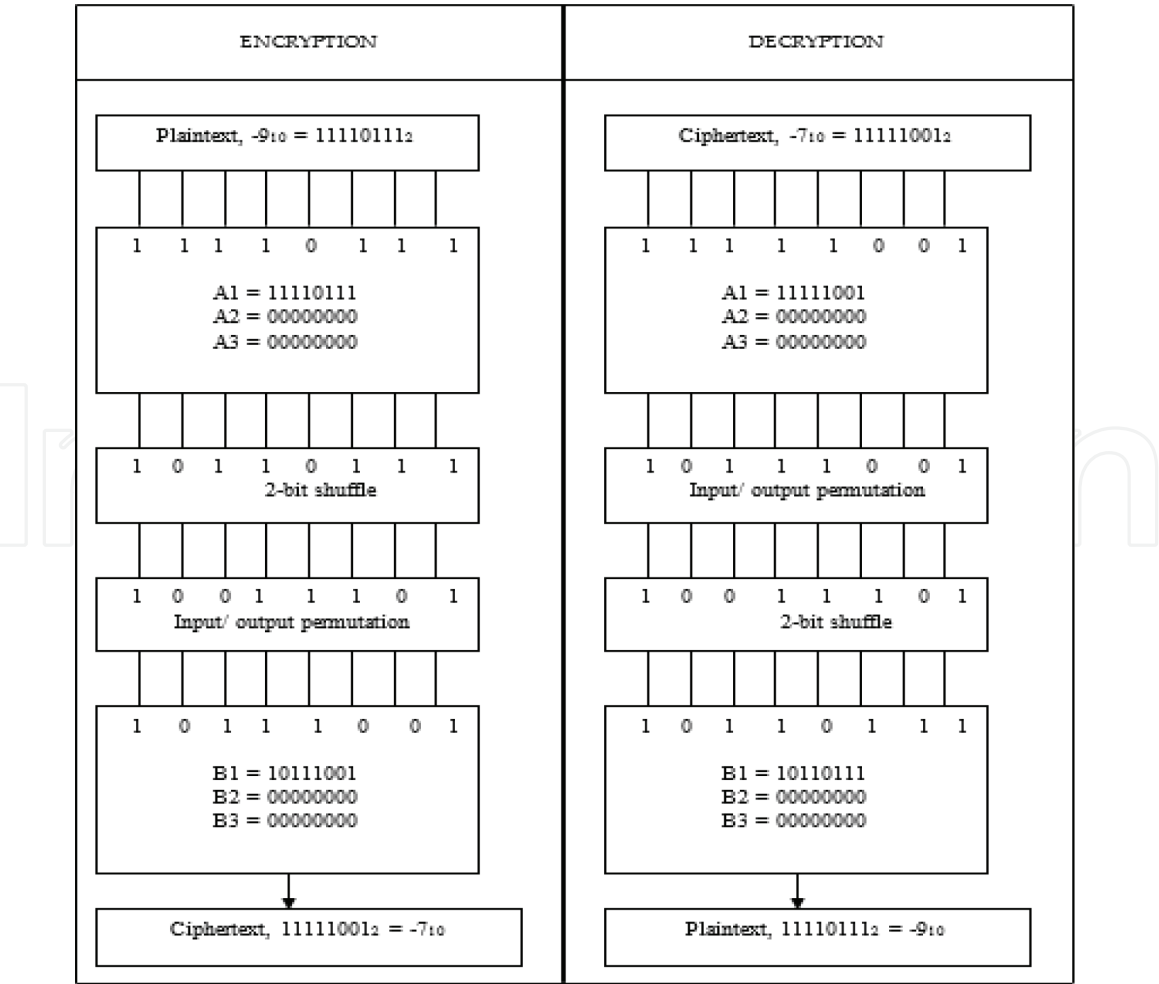| ENCRYPTION	DECRYPTION
Plaintext, $-9_{10} = 11110111_2$	Ciphertext, $-7_{10} = 11111001_2$
1 1 1 1 0 1 1 1 A1 = 11110111 A2 = 00000000 A3 = 00000000	1 1 1 1 1 0 0 1 A1 = 11111001 A2 = 00000000 A3 = 00000000
1 0 1 1 0 1 1 1 2-bit shuffle	1 0 1 1 1 0 0 1 Input/ output permutation
1 0 0 1 1 1 0 1 Input/ output permutation	1 0 0 1 1 1 0 1 2-bit shuffle
1 0 1 1 1 0 0 1 B1 = 10111001 B2 = 00000000 B3 = 00000000	1 0 1 1 0 1 1 1 B1 = 10110111 B2 = 00000000 B3 = 00000000
Ciphertext, $11111001_2 = -7_{10}$	Plaintext, $11110111_2 = -9_{10}$

Table 4.
Manual computation for the first sample of first level of decomposition for modulus 107.

r_{31} from subband encoding namely $a_0 = 2$ and $d_0 = 44$ is used. From **Figure 5**, upsampling performed on the approximation, a_0 and detail, d_0 data points gives the sets $y\_1 = \{2, 0\}$ and $z\_1 = \{44, 0\}$ respectively. Using 4-tap Daubechies integer lifting filter coefficients set, $h = \{2, 0, 0\}$ we have.

$$w\_1(0) = z\_1(0) - h(2)*y\_1(0) - h(3)*y\_1(1) = 44{-}0 - 0 = 44$$
$$w\_1(1) = z\_1(1) - h(2)*y\_1(0) - h(3)*y\_1(1) = 0{-}0 - 0 = 0$$
$$a\_1(0) = y\_1(0) - h(1)*w\_1(0) = 2{-}2*44 = -86$$
$$a\_1(1) = y\_1(1) - h(1)*w\_1(1) = 0{-}2*0 = 0$$

Hence the first level approximation data points, a_1 are obtained as follows.

$a_1(0) = w\_1(0) = 44$ and $a_1(1) = w\_1(1) + z\_1(0) = 0{-}86 = -86$
$\Rightarrow a_1 = \{44, -86\}$.

The process is repeated to obtain the second and third levels approximation data points. The third level approximation data points, a_3 should be equal to residue set, r_1 obtained from the RNS-based RSA ciphertext using the modulus, $m_1 = 107$.

b. RNS-based Chinese Remainder Theorem (CRT): It is applied to the residue sets r_1, r_2 and r_3.

(Eq. (8)) will be used to retrieve the RSA ciphertext from the residue sets. Using (Eq. (8)) and moduli set, $m = \{107, 109, 113\}$ to compute the first data point of the ciphertext set we have.

$$
\begin{aligned}
X &\equiv (r_{11} \times \hat{m}_1 \times T_1)\,(r_{21} \times \hat{m}_2 \times T_2)\,(r_{31} \times \hat{m}_3 \times T_3)(\bmod M) \\
&\equiv ((44 \times 12{,}317 \times 9) \;+\; (42 \times 12{,}091 \times 68) \;+\; (38 \times 11{,}663 \times 33))(\bmod 1{,}317{,}919) \\
&\Rightarrow X = 151
\end{aligned}
$$

The same process is repeated to obtain all the other data points of the RSA ciphertext set. The RSA ciphertext set is fed to the RSA decryption block at the application layer.

○ Application layer: RSA decryption

Decryption of the first data point is given as $M = 151^d \bmod n = 151^{173} \bmod 481 = 398$ which represents the original data which was sent at the source.

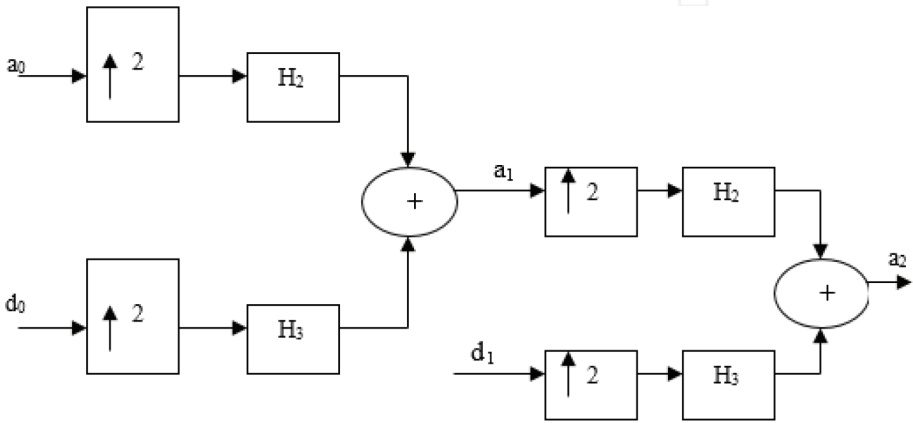The same process is repeated to obtain all the other data points of the plaintext set.

Figure 5.
The first two stages of the wavelet inverse transform.

4. FPGA implementation of security scheme applied to CDMA

In this section, FPGA implementation of new scheme applied to CDMA, the new VHDL code package to implement A mod n operations, synthesis report and behavioral simulation results will be presented.

4.1 FPGA implementation of new scheme applied to CDMA

Code division multiple access (CDMA) enables several users to transmit messages simultaneously over the same channel bandwidth in such a way that each transmitter/ receiver user pair has its own distinct signature code for transmitting over the common channel bandwidth. This distinct signature is ensured by using spread spectrum techniques whereby the message from each user is transmitted using orthogonal waveforms. In orthogonal signaling, the residues are mapped to orthogonal waveforms which constitute the CDMA signal [20]. The orthogonal waveforms used in this research are Walsh functions.

Considering an (8, 8, 2) multi-level cryptosystem for illustrative purposes, the mapping process will involve $M = 2^8 = 256$ orthogonal waveforms. Using the dynamic range of $(-128, 126)$, a set of $M = 2^8 = 256$ orthogonal waveforms is required to completely represent all the integers or symbols. Based on this, the corresponding Hadamard matrix obtained from the procedure elaborated in [21] is as follows:

$$
\begin{aligned}
H_{256} &= H_{128} \otimes H_{128} \\
&= \begin{pmatrix} H_{128} & H_{128} \\ H_{128} & \overline{H}_{128} \end{pmatrix}
\end{aligned}
$$

The H_{256} matrix is a large matrix comprising of 256 rows and 256 columns. The Hadamard matrix results into a multi–dimensional array. Multi–dimensional arrays are arrays with more than one index. Multi–dimensional arrays are not allowed for hardware synthesis. One way around this is to declare two one–dimensional array types. This approach is easier to use and more representative of actual hardware. The VHDL code used to declare the two one–dimensional array types is shown in **Figure 6** [22].

The other operations in the hardware Walsh function generator implementation are trivial since they involve modulo–2 addition with built–in operators in VHDL code to handle such operations.

4.2 New VHDL code package

In this research, a new algorithm is presented which implements modular exponentiation without the use of the Montgomery algorithm. A package is developed in the VHDL code to extract residues similarly to the X mod N operation for any randomly generated data. Meanwhile, the large operand lengths which resulted

```
Subtype Depth_Typ is Integer range 0 to 255;
Subtype Width_Typ is Integer range 255 downto 0;
Subtype Data_Typ is Bit_vector (Width_Typ);
Type Memory_Typ is array (Depth_Typ) of Data_Typ;
```

Figure 6.
VHDL code for synthesizable 256 × 256 Hadamard matrix.

from the modular exponentiation are reduced using binary exponentiation and the RNS.

The principle used to develop the package is as follows:

To perform the x = X mod N calculation where X has a large operand length of b bits say b = 1024 bits and N is modulus of small operand length of b_1 bits say b_1 = 8 bits, the following steps are used:

1. X is converted to binary equivalent;

2. The b_1 least significant bits of the b bits of X are chosen;

3. The integer equivalent, x_1 of the chosen b_1 bits is determined;

4. The residue, x = X mod N is obtained from the following equation;

$$x = x_1 + \left(2^{b_1-1} - N\right) \times \frac{X}{2^{b_1-1}} \tag{9}$$

5. If the residue, x is greater than the modulus, N the process is iterated until the residue is less than the modulus.

(Eq. (9)) forms the basis for the x = X mod N calculation.

Based on this new algorithm which implements modular exponentiation without the use of the Montgomery algorithm, the entire physical layer security scheme could fit into a single FPGA chip.

4.3 Behavioral simulation and synthesis report

In order to verify the performance of the proposed architecture, a VHDL programme was written and implemented on a Xilinx Virtex-4 FPGA chip (device: xc4vlx 200, package: ff 1513, speed grade − 11) [22]. Sixteen (16) randomly generated integers were fed into the FPGA. For this value, the number of bonded IOBs is 760 out of 960 resulting to 79% resource used. The behavioral simulation results for array {39,870, 45,378, 87,654, 20,087, 35,689, 16,592, 564, 276,509, 89,732, 56,287, 4527, 89,065, 4321, 7654, 5489, 512} using moduli set {111, 115, 119} are displayed in [15].

In [15], the complete synthesis report showing device utilization summary is presented. Due to the additional implementation of orthogonal signaling compared to the implementation in [15], the following parameters are different compared to results displayed in [15]:

The device utilization summary is as follows:

• Number of slices: 5411 out of 89,088 6%

• Number of slice flip flops: 60 out of 178,176 0%

• Number of four input LUTs: 7452 out of 178,176 4%

• Total REAL time to Router completion: 24 min 7 s.

• Total REAL time to place and route (PAR) completion: 24 min 41 s.

• Pin delays less than 1.00 ns: 21928 out of 30,651 71.5%.

5. Cryptanalysis of cross-layer scheme

The cryptanalysis will be performed separately at the application and physical layers and later combined in the cross-layer scheme to demonstrate the high security level of the new scheme compared to separate implementations.

5.1 Cryptanalysis at the application layer

The RSA public key cryptography is implemented at the application layer. The most successful method to break the RSA cryptosystem is the Number Field Sieve (NFS) method used for partial key exposure attacks. The NFS is based on a method known as "Fermat Factorization": one tries to find integers x, y, such that $x^2 \equiv y^2$ mod n but $x \neq \pm y$ mod n [12]. We assume that the two primes p and q should be close and approximately equal to the square root of n, where n = p.q. If one of the integers could be written as x = (p + q)/2 then number of steps, S_1 required to determine the other integer, y could be computed as follows [23].

$$S_1 = \frac{p+q}{2} - \sqrt{n} = \frac{\left(\sqrt{q} - \sqrt{p}\right)^2}{2} = \frac{\left(\sqrt{n} - p\right)^2}{2\,p} \tag{10}$$

It is partial key exposure attack since the number of steps, S_1 required for the attack depends on one of the primes.

Table 5 gives a summary of the number of steps required to break the traditional RSA cryptography implemented at the application layer using Fermat Factorization.

5.2 Cryptanalysis at the physical layer

Security at the physical layer is ensured by the multi-level convolutional cryptosystem which encrypts already encrypted data emanating from the RNS-based RSA. The cryptanalysis of the multi-level convolutional cryptosystem will be based on the ciphertext-only attack whereby, it is assumed that the attacker knows ciphertext of several messages encrypted with the same key and/or several keys. The keys used in the encryption are those mentioned in Section 2.1.2 for the non-linear (8, 8, 2) 2-cascaded convolutional cryptosystem.

It is shown in [12] that, for an (n, k, L) convolutional code, each generator matrix reveals at most p – k – 1 values of a private parameter, using Gaussian elimination for p blocks of input data. Hence, if q is the number of states, then to completely break the (k, k, L) N-cascaded cryptosystem, the minimum number of plaintext-ciphertext pairs (u, v) required is [12].

Operand key length	Total number of steps
16-bit	1
32-bit	1
64-bit	1.8×10^8
128-bit	8.0×10^{17}
256-bit	1.26×10^{25}
512-bit	2.53×10^{63}
1024-bit	3.3×10^{140}

Table 5.
Number of steps required to break the traditional RSA.

$$S_2 = \left\{ \left[p. \left\lceil \frac{q.2^k}{p-k-1} \times \frac{q.2^k}{p} \times \frac{k!}{p} \times \frac{1}{p} \times \left(\frac{k}{2}\right)^2 \times 2^2 \right] \right]^N \right\} \tag{11}$$

For an (8, 8, 2) 2-cascaded cryptosystem, k = 8 and the least number of plaintext-ciphertext blocks required is p = 10 due to the number of rows and columns in the generator matrix. Assuming q = 2 states, S_2 could be as

$$S_2 = \left\{ \left[10. \left\lceil \frac{2.2^8}{10-8-1} \times \frac{2.2^8}{10} \times \frac{8!}{10} \times \frac{1}{10} \times \left(\frac{8}{2}\right)^2 \times 2^2 \right] \right]^2 \right\} = 7.8 \times 10^{34} \tag{12}$$

Table 6 gives a summary of the number of steps required to break the (8, 8, 2) 2-cascaded cryptosystem using ciphertext-only attack.

5.3 Cryptanalysis of the new cross-layer security scheme

At the upper layer, huge key lengths such as 1024 bits and 2048 bits are used to implement the RSA. Such implementations will greatly compromise throughput at the physical layer due to modular exponentiation. Hence, the main objective of the new cross-layer security scheme is to increase security level at the physical layer despite the small valued data points transmitted derived from the RNS-based RSA in order to enhance throughput. Cryptanalysis is performed on the small residue RSA encrypted values. The analysis will be based on partial key exposure and ciphertext-only attacks at the physical layer for eavesdropper who could wiretap the transmitted data. The number of steps, S required to break the new cross-layer security scheme should be a product of S_1 and S_2 given as [12].

$$S = S_1 . \left\{ \left[p. \left\lceil \frac{q.2^k}{p-k-1} \times \frac{q.2^k}{p} \times \frac{k!}{p} \times \frac{1}{p} \times \left(\frac{k}{2}\right)^2 \times 2^2 \right] \right]^N \right\} \tag{13}$$

Table 7 gives a summary of the number of steps required to break the new cross-layer security scheme by using partial key exposure attack and ciphertext-only attacks for different cascaded stages.

Comparing **Tables 5–7**, it can be seen that high security levels comparable to the traditional 1024-bit RSA implemented at the upper layer could be attained using short operand key lengths of 128 bits and 256 bits for cross-layer security

Operand key length	Total number of steps
8-bit	7.8×10^{34}
16-bit	2.1×10^{57}
32-bit	1.4×10^{57}
64-bit	1.5×10^{62}
128-bit	6.1×10^{71}
256-bit	9.87×10^{87}
512-bit	2.68×10^{112}
1024-bit	1.42×10^{134}

Table 6.
Number of steps required to break the (8, 8, 2) 2-cascaded cryptosystem.

Operand key length	Total number of steps		
	N = 2	N = 3	N = 4
16-bit	2.1×10^{57}	9.6×10^{85}	4.4×10^{114}
32-bit	1.4×10^{57}	5.2×10^{85}	1.9×10^{114}
64-bit	1.5×10^{62}	1.8×10^{93}	2.3×10^{124}
128-bit	6.1×10^{71}	4.8×10^{107}	3.7×10^{143}
256-bit	9.87×10^{87}	9.8×10^{131}	9.7×10^{175}
512-bit	2.68×10^{112}	4.4×10^{168}	7.2×10^{224}

Table 7.
Number of steps required to break the cross-layer security scheme.

implemented at the physical layer. It is worth noting that, the security level could be much higher compared to the values displayed in **Table 7** if the S-boxes were implemented using 4-bit and 8-bit shuffling instead of the aforementioned 2-bit shuffling.

6. Data throughput quantification

The data throughput, T could be given as [24].

$$T = R(1–P_e)^N \tag{14}$$

where P_e is the bit error probability, N is the number of bits in the block length and R is a fixed transmission rate for the frames. For $P_e << 1$, the throughput could approximate to

$$T \cong R(1–NP_e) \tag{15}$$

From (Eq. (15)) it could be seen that, for a fixed transmission rate, R the throughput, T could be increased by either minimizing N or P_e. In this section, it will be shown how convolutional coding could be used to achieve both conditions through orthogonal signaling and forward error correction respectively.

6.1 Coded orthogonal signaling

It is shown in [16, 25] that, for coded orthogonal signaling, the bit error probability to transmit k-bit symbols is as follows:

$$P_b \leq \left(\frac{2^{k-1}}{2^k - 1} \right) \sum_{3}^{2^k} a_d \left[\frac{4 \left(1 + \frac{k}{L} \bar{\gamma}_b \right)}{\left(2 + \frac{k}{L} \bar{\gamma}_b \right)^2} \right]^L \tag{16}$$

where a_d denotes the number of paths of distance d from the all-zero path which merge with the all-zero path for the first time and $d_{free} = 3$ in this case, is the minimum distance of the code. d_{free} is also equal to the diversity, L. $\bar{\gamma}_b$ is the average signal-to-noise ratio (SNR) per bit [25]. For each convolutional code, the transfer function is obtained and the sum of the coefficients $\{a_d\}$ calculated.

For illustrative purposes, the transfer function, T(D) of smaller convolutional codes such as (2, 2, 2) and (4, 4, 2) will be used.

The transfer function T(D) for the (2, 2, 2) code is given as follows [16]:

$$T(D) = D^3 + 2\,D^4 + - - - \tag{17}$$

The transfer function for the (4, 4, 2) code is given as follows [16]:

$$T(D) = D^3 + 2D^4 + 3D^5 + 5D^6 + 9D^7 + 16D^8 + 28D^9 + 49D^{10} + 85D^{11} + - - - \tag{18}$$

For both the (2, 2, 2) code and the (4, 4, 2) code, $d_{\text{free}} = L = 3$. Using the values of L and $\{a_d\}$, the probability of a binary digit error, P_b as a function of the SNR per bit, $\overline{\gamma}_b$ is shown in **Figure 7** for k = 2 and 4 [16].

The curves illustrate that, the error probability increases with an increase in k for the same value of SNR. Hence, better performance for wireless transmission should involve lower order codes and many independent parallel channels rather than higher order codes with fewer independent parallel channels. Hence, high data throughput could be attained by using small number of bits in the block length, N.

6.2 Forward error correction (FEC) code

The Viterbi algorithm [25] is the most extensively decoding algorithm for Convolutional codes and has been widely deployed for forward error correction in wireless communication systems. In this sub-section Viterbi algorithm will be applied to the non-linear convolutional code. The constraint length, L for a (n,k,m) convolutional code is given as L = k(m-1). The constraint length is very essential in convolutional encoding since a Trellis diagram which gives the best encoding representation populates after L bits. Hence to encode blocks of n bits, each block has to be terminated by L zeros (0 s) before encoding.
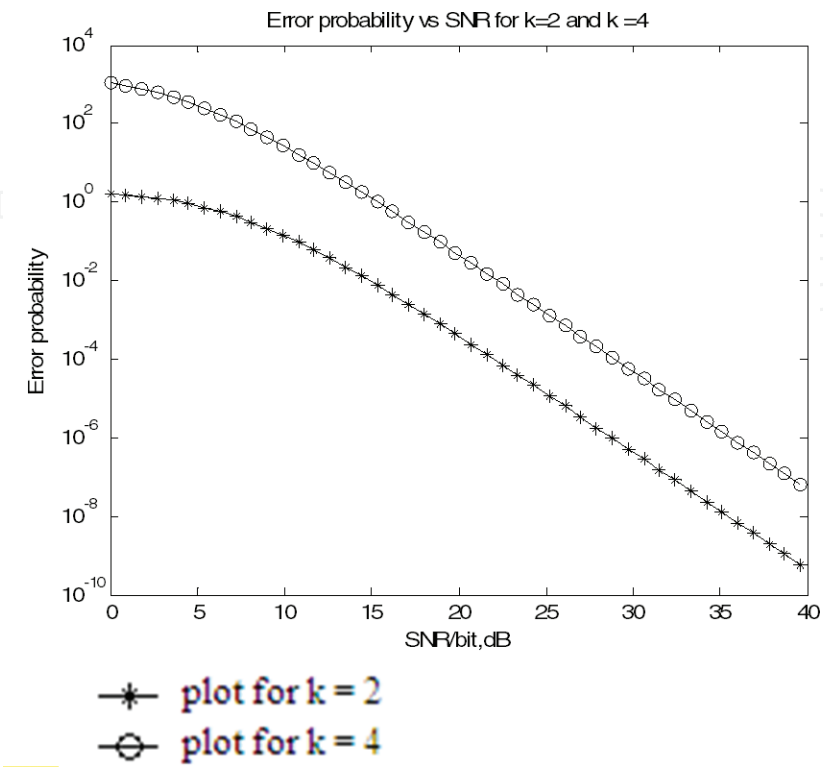

Figure 7.
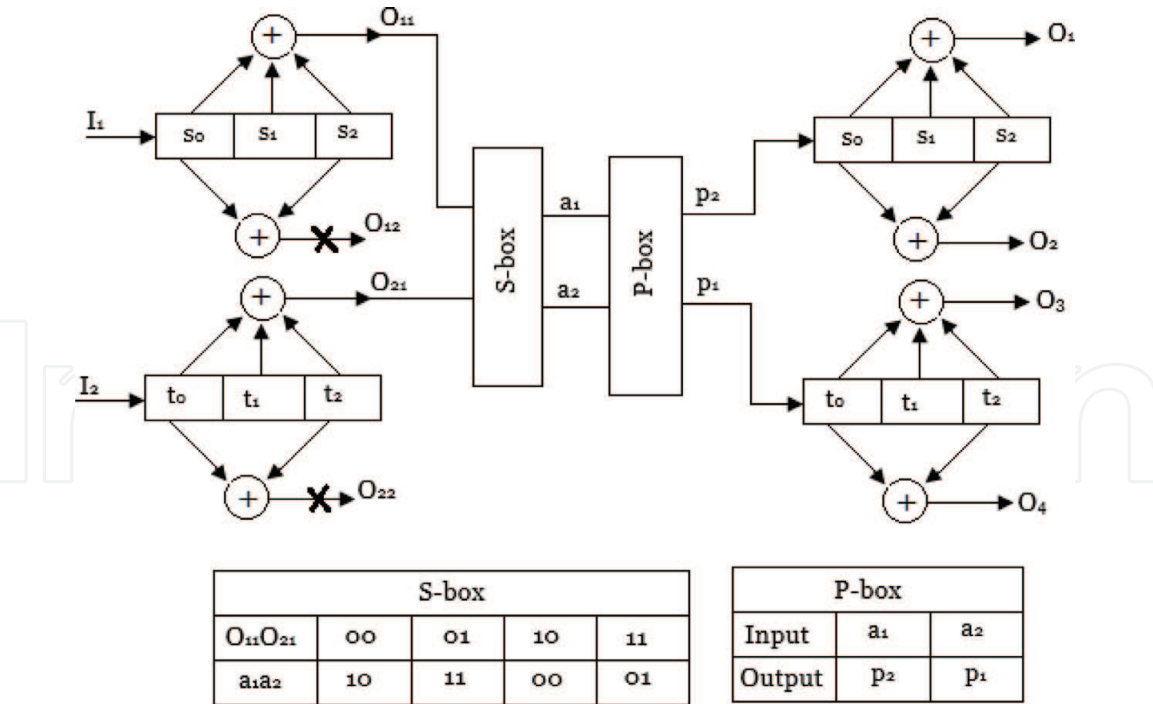*Performance of coded orthogonal signaling for k = 2 and k = 4.*

Figure 8.
2-stage non-linear (4,2,3) convolutional code.

For illustrative purposes, a non-linear (4,2,3) convolutional code will be used to demonstrate encoding and Viterbi decoding. A possible non-linear (4,2,3) convolutional code showing mod-2 connections and the product cipher is shown in **Figure 8**.

6.2.1 Example: encode/decode the message M = 10,011

- Encoding process

The constraint length, $L = k(m-1) = 2(3–1) = 4$.
Hence 4 zeros will be appended to message M before encoding. The modified message becomes M' = 10110000. Transition tables in appendix are used to encode the modified message.

a. Using transition tables in appendix, the transmitted sequence from the 1st stage is given as T_{in} = 10 01 01 11

b. S-box output is given as S = 00 11 11 01

c. P-box output is given as P = 00 11 11 10

d. Transmitted sequence into the 2nd stage is given as P = 00 11 11 10

e. Using transition tables in appendix, the final transmitted sequence which is the output bits from the 2nd stage is given as T_{out} = 0000 1111 0101 1001

- Viterbi decoding process

In performing the Viterbi algorithm, a bit in the sequence T_{out} will be altered. Let the received sequence be T_R = 1000 1111 0101 1001 instead of T_{out} = 0000 1111 0101 1001. The Viterbi algorithm applied to the 2nd stage is summarized in **Table 8**.

Incoming bits	Input State	Output bits	Output state	Present metric	Cumulative metric
	0 0 0 0	0 0 0 0 00 0 0 0 0	3	3
1 0 0 0	0 0 0 0	0 0 1 1	0 0 1 0	1	1
	0 0 0 0	1 1 0 0	1 0 0 0	3	3
	0 0 0 0	1 1 1 1	1 0 1 0	1	1
	0 0 0 0	0 0 0 0	0 0 0 0	0	3
	0 0 0 0	0 0 1 1	0 0 1 0	2	5
	0 0 0 0	1 1 0 0	1 0 0 0	2	5
1 1 1 1	0 0 0 0	1 1 1 1 11 1 0 1 0	4	7
	1 0 0 0	1 0 0 0	0 1 0 0	1	4
	1 0 0 0	1 0 1 1	0 1 1 0	3	6
	1 0 0 0	0 1 0 0	1 1 0 0	1	4
	1 0 0 0	0 1 1 1	1 1 1 0	3	6
	1 0 1 0	1 0 1 0	0 1 0 1	0	7
0 1 0 1	1 0 1 0	1 0 0 1	0 1 1 1	2	9
	1 0 1 0	0 1 1 0	1 1 0 1	2	9
	1 0 1 0	0 1 0 1 11 1 1 1 1	4	11
	1 1 1 1	0 1 0 1	0 1 0 1	2	13
1 0 0 1	1 1 1 1	0 1 1 0	0 1 1 1	0	11
	1 1 1 1	1 0 0 1 10 1 1 0 1	4	15
	1 1 1 1	1 0 1 0	1 1 1 1	2	13

Table 8.
Viterbi algorithm applied to 2nd stage of (4,2,3) code.

Incoming bits	Input State	Output bits	Output state	Present metric	Cumulative metric
	0 0 0 0	0 0	0 0 0 0	1	1
1 0	0 0 0 0	0 1	0 0 1 0	0	0
	0 0 0 0 10 1 0	1 0 0 0	2	2
	0 0 0 0	1 1	1 0 1 0	1	1
	1 0 0 0	1 0	0 1 0 0	0	2
0 1	1 0 0 0	1 1	0 1 1 0	1	2
	1 0 0 0	0 0	1 1 0 0	1	3
	1 0 0 0 11 0 1	1 1 1 0	2	4
	1 1 1 0 00 0 1	0 1 0 1	2	6
0 1	1 1 1 0	0 0	0 1 1 1	1	5
	1 1 1 0	1 1	1 1 0 1	1	5
	1 1 1 0	1 0	1 1 1 1	0	4
	0 1 0 1 00 1 1	0 0 0 0	2	8
1 1	0 1 0 1	1 0	0 0 1 0	1	7
	0 1 0 1	0 1	1 0 0 0	1	7
	0 1 0 1	0 0	1 0 1 0	0	6

Table 9.
Viterbi algorithm applied to 1st stage of (4,2,3) code.

The bits above the arrows will constitute the retrieved sequence from the 2nd stage. Hence, the retrieved sequence is given as, R_1 = 00 11 11 10. This sequence is fed to the P-box.

- P-box output is given as P1 = 00 11 11 01. Sequence, P1 is fed to the S-box

- S-box output is given as S1 = 10 01 01 11

Sequence, S1 is fed into the 1st stage to retrieve the final correct message. The Viterbi algorithm applied to the 1st stage is summarized in **Table 9**.

For a good trellis, the final state is the all-zero state as seen in the winning path in **Table 9**. The final received sequence is identical to the original transmitted message of M' = R_{final} = 10110000 despite the first bit error. Hence, using the non-linear convolutional code, the error bit was identified and corrected. The forward error correction capability will therefore enhance throughput, since the bit error rate, P_e is reduced.

7. Security attacks in wireless networks

Most attacks in wireless networks are classified into two categories: passive and active. Passive attacks such as eavesdropping and traffic analysis do not interfere with normal network operations as opposed to active attacks. Some of the attacks could be circumvented by the cross-layer security scheme presented in this research due to the following characteristics inherent in its implementation:

- RSA cryptographic algorithm at the upper layer: **Table 3** summarized the security services, which could be achieved by implementing RSA cryptography such as authentication, non-repudiation and confidentiality. These services are essential network security requirements which are vital in curbing attacks such as eavesdropping, masquerade attack and information disclosure since there will be a possibility of not attaining the final all-zero state if message is modified.

- Convolutional cryptosystem at the physical layer: The different keys generated are essential in ensuring confidentiality while the forward error correction capability is essential in curbing message modification attack.

Other attacks such as denial of service and replay attack could be circumvented if the cross-layer security scheme is associated with Transmission Control Protocol (TCP).

8. Conclusions

In this chapter, we have described a new cross-layer security scheme which has the advantage of enhancing both security and throughput as opposed to existing schemes which either enhances security or throughput but not both. The new scheme is implemented using the residue number system (RNS), non-linear convolutional coding and subband coding at the physical layer and RSA cryptography at the upper layers. By using RSA cryptography, the scheme could be used in encryption, authentication and non-repudiation with efficient key management as opposed to existing schemes, which had poor key management for large wireless networks since their implementation, was based on symmetric encryption techniques. Results show that, the new algorithm exhibits high security level for key sizes of 64, 128 and 256 bits when using three or more convolutional-cascaded stages. The security level is far above the traditional 1024-bit RSA which is already vulnerable. The vulnerability of 1024-bit RSA has led to the proposal of implementing higher levels such as 2048-bit and 4096-bit. These high level RSA schemes when implemented will greatly compromise throughput due to modular exponentiation. Hence the usefulness of a scheme such as the one presented in this chapter. In addition, Viterbi algorithm was performed for the new non-linear convolutional code in order to highlight the error correction capability. It was shown that, by using error correction codes on many small block lengths compared to one huge block length, throughput increases. Hence non-linear convolutional code is very critical in the implementation of the new scheme, since it contributes in enhancing both security and throughput. The entire scheme could be implemented at different access points in a wireless network since it fits in a single FPGA. Finally, the new cross-layer security scheme is essential in circumventing some attacks in wireless and computer networks.

A. Appendix: transition tables

Input Bits (stage 1) I_1 I_2	Input State (Stage 1) S_1 S_2 t_1 t_2	Output Bits (Stage 1) O_{11} O_{21}	Output State (Stage 1) S_1 S_2 t_1 t_2
0 0	0 0 0 0	0 0	0 0 0 0
0 1	0 0 0 0	0 1	0 0 1 0
1 0	0 0 0 0	1 0	1 0 0 0
1 1	0 0 0 0	1 1	1 0 1 0
0 0	0 0 0 1	0 1	0 0 0 0
0 1	0 0 0 1	0 0	0 0 1 0
1 0	0 0 0 1	1 1	1 0 0 0
1 1	0 0 0 1	1 0	1 0 1 0
0 0	0 0 1 0	0 1	0 0 0 1
0 1	0 0 1 0	0 0	0 0 1 1
1 0	0 0 1 0	1 1	1 0 1 1
1 1	0 0 1 0	1 0	1 0 1 1
0 0	0 0 1 1	0 0	0 0 0 1
0 1	0 0 1 1	0 1	0 0 1 1
1 0	0 0 1 1	1 0	1 0 0 1
1 1	0 0 1 1	1 1	1 0 1 1
0 0	0 1 0 0	1 0	0 0 0 0
0 1	0 1 0 0	1 1	0 0 1 0
1 0	0 1 0 0	0 0	1 0 0 0
1 1	0 1 0 0	0 1	1 0 1 0
0 0	0 1 0 1	1 1	0 0 0 0
0 1	0 1 0 1	1 0	0 0 1 0
1 0	0 1 0 1	0 1	1 0 0 0
1 1	0 1 0 1	0 0	1 0 1 0
0 0	0 1 1 0	1 1	0 0 0 1
0 1	0 1 1 0	1 0	0 0 1 1
1 0	0 1 1 0	0 1	1 0 0 1
1 1	0 1 1 0	0 0	1 0 1 1
0 0	0 1 1 1	1 0	0 0 0 1
0 1	0 1 1 1	1 1	0 0 1 1
1 0	0 1 1 1	0 0	1 0 0 1
1 1	0 1 1 1	0 1	1 0 1 1

Input Bits (stage 1) I_1 I_2	Input State (Stage 1) S_1 S_2 t_1 t_2	Output Bits (Stage 1) O_{11} O_{21}	Output state (Stage 1) S_1 S_2 t_1 t_2
0 0	1 0 0 0	1 0	0 1 0 0
0 1	1 0 0 0	1 1	0 1 1 0
1 0	1 0 0 0	0 0	1 1 0 0
1 1	1 0 0 0	0 1	1 1 1 0
0 0	1 0 0 1	1 1	0 1 0 0
0 1	1 0 0 1	1 0	0 1 1 0
1 0	1 0 0 1	0 1	1 1 0 0
1 1	1 0 0 1	0 0	1 1 1 0
0 0	1 0 1 0	1 1	0 1 0 1
0 1	1 0 1 0	1 0	0 1 1 1
1 0	1 0 1 0	0 1	1 1 0 1
1 1	1 0 1 0	0 0	1 1 1 1
0 0	1 0 1 1	1 0	0 1 0 1
0 1	1 0 1 1	1 1	0 1 1 1
1 0	1 0 1 1	0 0	1 1 0 1
1 1	1 0 1 1	0 1	1 1 1 1
0 0	1 1 0 0	0 0	0 1 0 0
0 1	1 1 0 0	0 1	0 1 1 0
1 0	1 1 0 0	1 0	1 1 0 0
1 1	1 1 0 0	1 1	1 1 1 0
0 0	1 1 0 1	0 1	0 1 0 0
0 1	1 1 0 1	0 0	0 1 1 0
1 0	1 1 0 1	1 1	1 1 0 0
1 1	1 1 0 1	1 0	1 1 1 0
0 0	1 1 1 0	0 1	0 1 0 1
0 1	1 1 1 0	0 0	0 1 1 1
1 0	1 1 1 0	1 1	1 1 0 1
1 1	1 1 1 0	1 0	1 1 1 1
0 0	1 1 1 1	0 0	0 1 0 1
0 1	1 1 1 1	0 1	0 1 1 1
1 0	1 1 1 1	1 0	1 1 0 1
1 1	1 1 1 1	1 1	1 1 1 1

Input Bits (Stage 2) P_1 P_2	Input State (Stage 2) S_1 S_2 t_1 t_2	Output Bits (Stage 2) O_1 O_2 O_3 O_4	Output State (Stage 2) S_1 S_2 t_1 t_2
0 0	0 0 0 0	0 0 0 0	0 0 0 0
0 1	0 0 0 0	0 0 1 1	0 0 1 0
1 0	0 0 0 0	1 1 0 0	1 0 0 0
1 1	0 0 0 0	1 1 1 1	1 0 1 0
0 0	0 0 0 1	0 0 1 1	0 0 0 0
0 1	0 0 0 1	0 0 0 0	0 0 1 0
1 0	0 0 0 1	1 1 1 1	1 0 0 0
1 1	0 0 0 1	1 1 0 0	1 0 1 0
0 0	0 0 1 0	0 0 1 0	0 0 0 1
0 1	0 0 1 0	0 0 0 1	0 0 1 1
1 0	0 0 1 0	1 1 1 0	1 0 1 1
1 1	0 0 1 0	1 1 0 1	1 0 1 1
0 0	0 0 1 1	0 0 0 1	0 0 0 1
0 1	0 0 1 1	0 0 1 0	0 0 1 1
1 0	0 0 1 1	1 1 0 1	1 0 0 1
1 1	0 0 1 1	1 1 1 0	1 0 1 1
0 0	0 1 0 0	1 1 0 0	0 0 0 0
0 1	0 1 0 0	1 1 1 1	0 0 1 0
1 0	0 1 0 0	0 0 0 0	1 0 0 0
1 1	0 1 0 0	0 0 1 1	1 0 1 0
0 0	0 1 0 1	1 1 1 1	0 0 0 0
0 1	0 1 0 1	1 1 0 0	0 0 1 0
1 0	0 1 0 1	0 0 1 1	1 0 0 0
1 1	0 1 0 1	0 0 0 0	1 0 1 0
0 0	0 1 1 0	1 1 1 0	0 0 0 1
0 1	0 1 1 0	1 1 0 1	0 0 1 1
1 0	0 1 1 0	0 0 1 0	1 0 0 1
1 1	0 1 1 0	0 0 0 1	1 0 1 1
0 0	0 1 1 1	1 1 0 1	0 0 0 1
0 1	0 1 1 1	1 1 1 0	0 0 1 1
1 0	0 1 1 1	0 0 0 1	1 0 0 1
1 1	0 1 1 1	0 0 1 0	1 0 1 1

Input Bits (Stage 2) P_1 P_2	Input State (Stage 2) S_1 S_2 t_1 t_2	Output Bits (Stage 2) O_1 O_2 O_3 O_4	Output state (Stage 2) S_1 S_2 t_1 t_2
0 0	1 0 0 0	1 0 0 0	0 1 0 0
0 1	1 0 0 0	1 0 1 1	0 1 1 0
1 0	1 0 0 0	0 1 0 0	1 1 0 0
1 1	1 0 0 0	0 1 1 1	1 1 1 0
0 0	1 0 0 1	1 0 1 1	0 1 0 0
0 1	1 0 0 1	1 0 0 0	0 1 1 0
1 0	1 0 0 1	0 1 1 1	1 1 0 0
1 1	1 0 0 1	0 1 0 0	1 1 1 0
0 0	1 0 1 0	1 0 1 0	0 1 0 1
0 1	1 0 1 0	1 0 0 1	0 1 1 1
1 0	1 0 1 0	0 1 1 0	1 1 0 1
1 1	1 0 1 0	0 1 0 1	1 1 1 1
0 0	1 0 1 1	1 0 0 1	0 1 0 1
0 1	1 0 1 1	1 0 1 0	0 1 1 1
1 0	1 0 1 1	0 1 0 1	1 1 0 1
1 1	1 0 1 1	0 1 1 0	1 1 1 1
0 0	1 1 0 0	0 1 0 0	0 1 0 0
0 1	1 1 0 0	0 1 1 1	0 1 1 0
1 0	1 1 0 0	1 0 0 0	1 1 0 0
1 1	1 1 0 0	1 0 1 1	1 1 1 0
0 0	1 1 0 1	0 1 1 1	0 1 0 0
0 1	1 1 0 1	0 1 0 0	0 1 1 0
1 0	1 1 0 1	1 0 1 1	1 1 0 0
1 1	1 1 0 1	1 0 0 0	1 1 1 0
0 0	1 1 1 0	0 1 1 0	0 1 0 1
0 1	1 1 1 0	0 1 0 1	0 1 1 1
1 0	1 1 1 0	1 0 1 0	1 1 0 1
1 1	1 1 1 0	1 0 0 1	1 1 1 1
0 0	1 1 1 1	0 1 0 1	0 1 0 1
0 1	1 1 1 1	0 1 1 0	0 1 1 1
1 0	1 1 1 1	1 0 0 1	1 1 0 1
1 1	1 1 1 1	1 0 1 0	1 1 1 1

Author details

Michael Ekonde Sone
College of Technology, University of Buea, Buea, Cameroon

*Address all correspondence to: michael.sone@ubuea.cm

IntechOpen

References

[1] Patterson W, Sone ME. A metric to assess cyberattacks. In: Proceedings of the AHFE 2018 International Conference on Human Factors in Cybersecurity; AISC 782; Florida, USA. 2018. pp. 33-43

[2] Guo C, Chang C-C, Chang S-C. A secure and effcient mutual authentication and key agreement protocol with smart cards for wireless communications. International Journal of Network Security. 2018;**20**(02): 323-331

[3] Bloch M et al. Wireless information-theoretic security. IEEE Transactions on Information Theory. 2008:2515-2534

[4] Wyner AD. The wire-tap channel. Bell System Technical Journal. October 1975;**54**(8):1355-1387

[5] Lin F, Oggier F. Coding for wiretap channels. In: Physical Layer Security in Wireless Communications. Auerbach Publications, CRC Press, Taylor & Francis Group. 2013

[6] Shui Y-S, Cheng SY, Wu H-C, Huang SC-H, Chen H-H. Physical Layer Security in Wireless Networks: A Tutorial. IEEE Wireless Communications; April 2011

[7] Rosario BBD, Rajaram A. CDMA based secure cross layer framework for authentication and scheduling in MANET. Journal of Theoretical and Applied Information Technology. November 2014

[8] Babu KS, Chandrasekharaiah K. Cross layer based security in MANET. International Journal of Advanced Research in Computer Science. May 2013;**4**(5) Special issue

[9] Rajaram A, Palaniswami S. A trust-based cross-layer security protocol for Mobile ad hoc networks. (IJCSIS)

International Journal of Computer Science and Information Security. 2009; **6**(1)

[10] Uytterhoeven G, Van Wulpen F, Jansen M, Roose D, Bultheel A. WAILLI: Wavelets with integer lifting. In: Technical Report TW 262. Leuven: Department of Computer Science, Kaththolieke Universiteit Leuven; 1997

[11] Sweldens W. The lifting scheme: A construction of second generation wavelets. SIAM Journal on Mathematical Analysis. 1997;**29**(2). Preprint 1996

[12] Sone ME. Efficient key management scheme to enhance security-throughput trade-off performance in wireless networks. In: Proceeding of Science and Information Conference (SAI); July 28-30 2015; London, UK. 2015. pp. 1249-1256

[13] Daubechies I, Sweldens W. Factoring wavelet transforms into lifting steps. Journal of Fourier Analysis and Applications. 1998;**4**(3). Preprint

[14] Mashen MJ. Factoring wavelet transforms into lifting steps. In: 1997 Circuits and Systems (ISCAS 2004), vol. 2; University of Australia. 2004. pp. 667-700

[15] Ningo NN, Sone ME. Efficient key management FPGA-based cryptosystem using the RNS and iterative coding. International Journal of Information and Communication Technology. 2010;**2**(4): 302-322

[16] Sone ME, Ningo NN. A simple FPGA-based wireless transmitter/receiver convolutional cryptosystem. International Journal of Computers and Applications;**33, 2**:2011

[17] Yang LL, Hanzo L. Redundant residue number system based error

correction codes. In: Proceedings of
IEEE VTC'01 (Fall); October; Atlantic
City, NJ. 2001. pp. 1472-1476

[18] Soderstrand M, Jenkins W, Jullien
GA, Taylor FJ. Residue Number System
Arithmetic: Modern Applications in
Digital Signal Processing, IEEE Press
Reprint Series. IEEE Press; 1986

[19] Mashen MJ. Factoring wavelet
transforms into lifting steps. In: 1997
Circuits and Systems (ISCAS 2004),
vol. 2; University of Australia. 2004.
pp. 667-700

[20] Yang L-L, Hanzo L. Performance
analysis of coded M-ary orthogonal
signaling using errors-and-erasures
decoding over frequency selective
fading channels. IEEE Journal on
Selected Areas in Communications. Feb.
2001;**19**:211-221

[21] Geadah YA, Corinthios MJG.
Natural, dyadic, and sequence order
algorithms and processors for the
Walsh-Hadamard transforms. IEEE
Transactions on Computers. May 1977;
26(5):435-442

[22] Cohen B. VHDL coding styles and
methodologies. In: An In-Depth
Tutorial. 2nd ed. Massachusetts, USA:
Kluwer Academic Publishers; 2001

[23] Lenstra A, Lenstra HW Jr. The
Development of the Number Field
Sieve. Lecture Notes in Mathematics,
vol. 1554. Springer-Verlag; 1994

[24] Haleem MA, Mathur CN,
Chandramouli R, Subbalakshmi KP.
Opportunistic encryption: A trade-off
between security and throughput in
wireless networks. IEEE Transactions on
Dependable and Secure Computing.
October-December 2007;**4**(4)

[25] Proakis J. Digital Communications.
4th ed. New York: McGraw-Hill; 2001