ECC with Hidden Generator Point in WSNs

Ravi Kishore Kodali
Department of Electronics and Communication Engineering
National Institute of Technology, Warangal
Andhra Pradesh, India

***Abstract*—Wireless Sensor Networks (WSNs), comprising of tiny nodes with limited energy, computational and communication resources, are being widely used in various application areas ranging from pollution and weather monitoring to military. Even though every application may not require data to be exchanged in a secure manner, few WSN applications may have certain security requirements. The data is encrypted by a sender and sent over the wireless links and the same is decrypted at the receiver. To meet this purpose, symmetric key cryptographic (SKC) algorithms may be used. Such SKC primitives require keys to be made available before carrying out the data transfer between the nodes. Alternately, Public Key Cryptographic techniques, such as RSA algorithm can be considered. Even though, RSA is a popular algorithm providing good security level, it is computationally intensive involving large key sizes. The RSA can not be used in WSNs, as the nodes have limited resources. Presently, it is infeasible to implement the RSA algorithm using any of the WSN nodes commercially available. Elliptic curve cryptography (ECC), another public key cryptographic (PKC) algorithm providing same level of security with smaller key size requirements, can be used as an alternative in order to provide security in WSN applications. ECC encryption and decryption use domain parameters, which includes the *Generator point* to be published. In most of the outdoor WSN applications, the deployment of the nodes is random and the nodes could be captured and an attacker could launch man-in-middle (MIM) attack, and break the public key thereby leading to security breach in the network. A technique to overcome such an attack is proposed in this work and the same is compared with two other similar approaches.**

Keywords- Security, ECC, WSN

I. INTRODUCTION

Wireless sensor networks (WSNs) are in great demand and assisting in monitoring the sensitive and fragile parts of the world. They are finding their applications in diversified areas, as the WSN nodes are of tiny size and inexpensive. In WSN applications, the nodes are deployed randomly over the field and the nodes communicate among themselves to form a network [1]. Figure 1 shows a WSN, in which the sensors in the nodes sense physical phenomena related parameters from their surroundings and pass the digitized data about the target area to the querying users making use of the Internet through the gateway node in the WSN using multi-hop wireless communication links. The nodes have various resource constraints such as lesser computational capabilities, limited battery energy and small memory. For example, the IRIS mote consists of an 8- bit ATMega 1281 micro-controller [XM2110]. The IRIS mote has a maximum data rate of 250 Kbits/s, IEEE 802.15.4 compliant RF transceiver and direct sequence spread spectrum (DSSS)radio, and the flash memory size being 512 KB [2]. Some of the WSN applications like military, fire detection, etc. [3] demand the need for secure and authenticated information exchange in the network. The standard cryptographic algorithms such as the RSA require a large number of computations, and are unsuitable for WSN nodes [4]. On the contrary, Elliptic curve cryptography (ECC) has the capability of providing a similar security level with lesser key sizes as compared to the RSA.

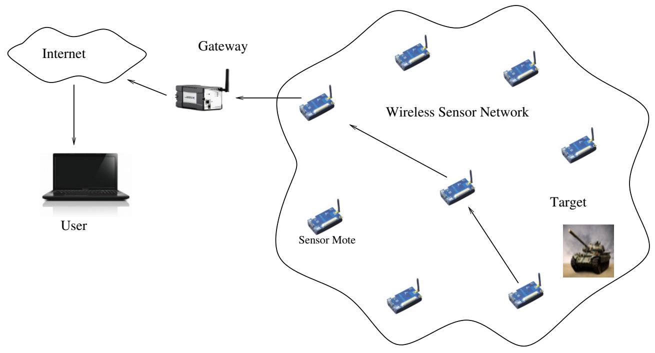

Fig. 1. Wireless Sensor Network [1]

The underlying strength of ECC depends on the elliptic curve discrete logarithm problem (ECDLP), which requires full exponential time to solve. Hence, ECC is used to provide security and it is more suitable for the resource constrained WSN nodes. Table I shows that with a key size of 160- bits, ECC provides a similar level of security as in RSA/DH with a key size of 1024- bits.

TABLE I. COMPARABLE KEY SIZES [5]

RSA/DH	1024	2048	3072	7689
ECC	163	233	283	409

In outdoor WSN applications the nodes are deployed randomly. An intruder can easily gain access to the nodes and can capture the data exchanged among the nodes through wireless links and can take advantage of the resource constrained nature of the nodes to break the public key, thereby breaching the security of the network. Such an attack is called man-in-middle (MIM) attack [6]. The WSNs are vulnerable to such attacks. The WSN nodes which use ECC for encryption and decryption publish the domain parameters which include a *Generator point*, (G). This can help an intruder to launch a MIM attack easily. This work discusses techniques, which make use of the ECC, with a hidden generator point to overcome the MIM attack.

The rest of the paper is organized as follows: Section II discusses the Elliptic curve cryptography, section III presents ECC with hidden generator point, section IV provides a brief description about the hardware used,section V gives a comparison of different techniques and section VI provides conclusions.

II. Elliptic Curve Cryptography

A standard elliptic curve (EC), $E_P(a,b)$, specifically for the purpose of cryptography over the prime field (F_P) is given as:

$$\boxed{y^2 \; mod \; p = (x^3 + ax + b) \; mod \; p,} \tag{1}$$

where a,b $\in F_P$ and $(4a^3 + 27b^2) \; mod \; p \neq 0$ [7]. The points, lying on $E_P(a,b)$, are calculated using equation (1). Addition of two points (Point Addition) and doubling of a point (Point Doubling) are the basic operations performed over EC, and are given in Table II.

TABLE II. EC Mathematical Operations [8]

EC Operation	Slope (S)	x_3	y_3
Point Addition $P(x_1,y_1)$ $+Q(x_2,y_2)$	$\frac{y_2 - y_1}{x_2 - x_1}$	$S^2 - x_1 - x_2$	$S(x_1 - x_3) - y_1$
Point Doubling $2P(x_1,y_1)$	$\frac{3x_1^2 + a}{2y_1}$	$S^2 - 2x_1$	$S(x_1 - x_3) - y_1$

The formulae for point addition and point doubling operations as given in the Table II, consider the elliptic curve using the affine co-ordinate system. Various co-ordinate systems, like Jacobian, projective and mixed co-ordinate systems [8] can be used to carry out this operation, which leads to less number of prime field operations.

The two frequently used operations in ECC are: scalar multiplication and modular reduction. The scalar multiplication based on ECDLP, consumes about 85% of computational cost in ECC [5]. A scalar multiplication comprises of point addition and point doubling operations. The scalar multiplication operation on elliptic curve is represented as: $T = k * G$, where T, G are the points on the elliptic curve and k is an integer in the prime field. Different scalar multiplication algorithms, recode the integer, k, to reduce the number of 1s in it i.e. reduce its hamming weight (W). This reduces the number of point addition operations required to carry out a scalar multiplication, thereby speeding up of the operation. Algorithm 1 shows the Binary method for scalar Multiplication.

Algorithm 1 Binary Method for Scalar Multiplication [9]

```
Input: A point P ∈ E($F_q$), an integer k of $l$ bit
$k = \sum_{j=0}^{l-1} K_j 2^j, \; K_j \in \{0,1\}$
Output: Q=kP
1: Q← $\infty$
2: for all $j = l - 1$ to 0 do
3:     Q← [2]Q
4:     if $k_j = 1$ the $Q \leftarrow Q + P$
5: end for
6: Return Q
```

In scalar multiplication, point addition and point doubling operations are used to determine computational cost of different algorithms. The number of $1'$s in the binary representation of k is called its Hamming weight (W) and l is the total number of bits in k. The computational cost of the Binary method is given by equation (2).

$$\boxed{Cost = (W - 1)A \; + \; (l - 1)D} \tag{2}$$

As the computational cost of Binary Method for scalar multiplication is very high, an implementation of this method in WSN nodes consumes more time and energy. Hence, a more efficient and less computationally expensive, sliding window method is introduced for the implementation in WSN nodes as discussed in Section V

In this method, digits used for representing of k can be extended beyond two bits as in Binary, $\{0,1\}$. This reduces the number of point additions. But this advantage comes at a cost, certain values, that are multiple of G, should be pre-computed and stored in the memory, such that they are added or subtracted to the Q [10] during multiplication. The memory required to hold these pre-computed values becomes a constraint. The sliding window method processes at most consecutive w digits of the scalar integer k, such that the decimal equivalent of the window-w consecutive digit should be odd. This method has no fixed window width, the same can be varied from 1 to w and 0 bit is ignored.

Algorithm 2 Binary Sliding window for scalar multiplication [11]

```
Input : Generator point G, k, window width-w
Output:$Q = kG$
1: Calculate [x]G where $x = 1, 3, 5...., (2^{(w-1)} - 1)$
2: $j \leftarrow l - 1$, where $l$ is length of k
3: while $j \geq 0$ do
4:     if $(K_j == 0)$
5:     $Q \leftarrow [2]Q$ ,$N \leftarrow 0$, $j \leftarrow j - 1$
6:     end if
7:     else
8:     $i \leftarrow maximum(j - w + 1, 0)$
9:     while $K_i == 0$ do
10:        $i \leftarrow i + 1$
11:    end while
12:    for $d = 1 \quad to \quad (j - i + 1)$ do
13:        $d = d + 1 \; and \; Q \leftarrow [2]Q$
14:    end for
15:    $N \leftarrow (K_j.......K_i)_2$
16:    $j \leftarrow i - 1$
17:    end else
18:    $Q \leftarrow Q \oplus [N]G$
19: end while
20: Return $Q$
```

Algorithm 2 presents the scalar multiplication for the sliding window method with binary representation of integer k.

The computational cost for the binary sliding window method is given by equation (3).

$$\boxed{Cost = \frac{l}{w + v(w)} A \; + \; LD} \quad (3)$$

The number of pre-computations required in this method is:

$$\boxed{P = LD + (2^{w-1} - 1)A,} \quad (4)$$

where $V(w)$, as given in equation (5), is the average length of a run of $0'$s within the window [7].

$$\boxed{V(w) = \frac{4}{3} - \frac{(-1)^w}{3 * 2^{w-2}}} \quad (5)$$

III. ECC with Hidden Generator Point

The key exchange mechanism in traditional ECC is shown in Figure 2.

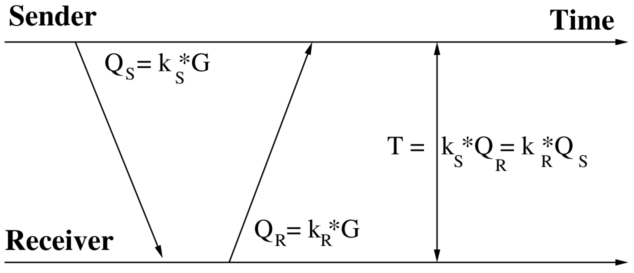

Fig. 2. Elliptic Curve Diffie-Hellamn [12]

Both the sender and the recipient nodes select an elliptic curve (EC), $E_P(a, b)$ and the domain parameters which includes generator point. The generator point, G, is a point on the EC, $E_P(a, b)$, with the highest order [10]. Both the nodes independently select the private keys by choosing scalar integers randomly in F_P. Then both the nodes calculate their respective public keys, Q_S and Q_R, by multiplying the G, with the corresponding private keys, k_S and k_R. Then Q_S and Q_R are made published in the network. Both the nodes again multiply these public keys with their respective private keys and a shared secret key, $T = k_S * Q_R = k_R * Q_S$, is generated. In ECC, the domain parameters which include the *Generator point*, G, are made public and also the intruder comes to know, making the network vulnerable to the man-in-middle (MIM) attack. In order to secure the network against such attacks, ECC with a *Hidden generator point* is implemented and to carry out the same, three different protocols are given.

In the first protocol as given in [13], a *Certificate Authority (CA)* is made use of. Consider the prime field, F_31, the EC points over the elliptic curve (EC), $E_{31}(-3, 1)$ and the distribution of points over the EC are illustrated in Figure 3. An EC point, G, is selected as the *Generator point* such that $G \in E_{31}(-3, 1)$. It is assumed that every EC point has the same order. The EC points are distributed into different groups so as to assist in the *Generator point* selection. Once the G selection is made, the nodes can communicate among themselves in same way as in the shared key generation. This protocol can be implemented easily as and it involves lesser computations. However, a *CA* needs to be used in the WSN, making its implementation difficult. If the CA related information is captured, the protocol's security strength gets weakened as there can be a security breach in the network.

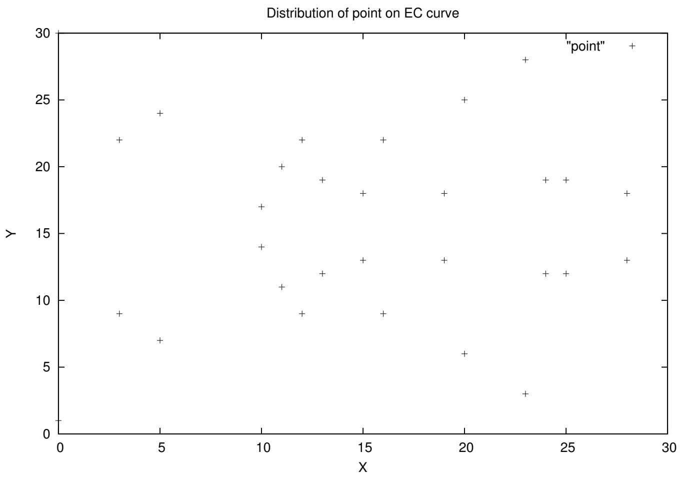

Fig. 3. Distribution of points on the Elliptic curve

The second protocol as given in [13] does not require a CA. The message exchange in the second protocol is shown in Figure 4. In this, sender and receiver exchange few messages before sending the data, D.

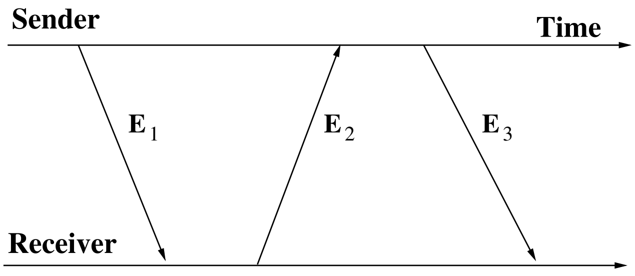

Fig. 4. Second Protocol operation [13]

A *Generator point*, G along with a private key, k_S are selected by the sender. The inverse of the private key, k_S^{-1} is calculated such that $k_S * k_S^{-1} = 1$.
In a similar manner, private key, k_R is selected by the receiver and its inverse, k_R^{-1}, is also calculated.

1) A message point, E_1, using equation (6), is computed and the same is sent.

$$\boxed{E_1 = (k_S^{-1} G, k_S^{-1} D + k_S^{-1} G)} \quad (6)$$

2) The receiver multiplies E_1 with k_R^{-1} and E_2 is calculated using equation 2 and it is sent back to the sender.
$k_S^{-1} D = k_S^{-1} D + k_S^{-1} G - k_S^{-1} G$
$E_2 = k_S^{-1} k_R^{-1} D$
3) After receiving E_2, it is multiplied with the sender's private key using equation (7) and it is sent to the receiver.

$$\boxed{E_3 = k_S^{-1} k_R^{-1} D * k_S = k_R^{-1} D} \quad (7)$$

4) At the receiver, E_3 is again multiplied with its private key to extract the message using equation (8).

$$\boxed{Message = k_R^{-1} D * k_R = D} \tag{8}$$

5) Finally, the recipient node can decrypt the message by multiplying E_3 with its private key.

The above steps need to be performed for every new message to be exchanged, requiring more computational overhead.

It can be noticed that a message exchange using the second protocol involves many compute intensive and energy consuming scalar operations and its implementation using the WSN nodes exhausts the energy soon.

A. Proposed Protocol

To overcome the computational overhead problem of the second protocol, another technique supporting same security level is being proposed. In this protocol, a unique shared key is generated for each communication session between the nodes. Figure 5 illustrates how a unique shared key is generated between the nodes without a common generator point.

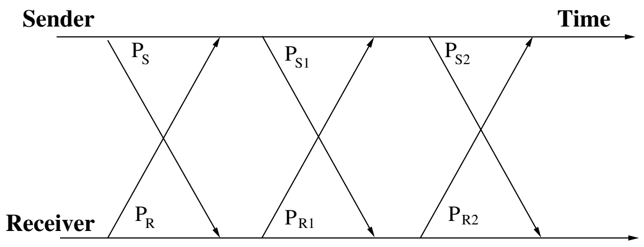

Fig. 5. Shared Key exchange mechanism using the proposed protocol [14]

In this protocol, the implementation of ECC with hidden generator point has been carried out. A shared key between the nodes is generated during the initial stage of each session involving few message exchanges. Both the sender and the recipient nodes have chosen the private keys, k_S and k_R respectively such that $k_S * k_S^{-1} = 1$ and $k_R * k_R^{-1} = 1$. Now, both the nodes compute the public keys, P_S and P_R using equation (9) and the same are made public.

$$\boxed{P_S = k_S^{-1} * G_S} \tag{9a}$$

$$\boxed{P_R = k_R^{-1} * G_R} \tag{9b}$$

Both the nodes receive the public key of each other. Then, the nodes multiply each other's public key with the inverse of its own private key and the modified keys are sent to each other. The sender computes P_{S1} using equation (10a) and the receiver uses equation (10b) to calculate P_{R1}.

$$\boxed{P_{S1} = k_R^{-1} * G_R * k_S^{-1}} \tag{10a}$$

$$\boxed{P_{R1} = k_S^{-1} * G_S * k_R^{-1}} \tag{10b}$$

After P_{S1} and P_{R1} are received, these are multiplied with their respective private keys using equation (11) and the same are transmitted again.

$$\begin{aligned} P_{S2} &= P_{R1} * k_S \\ &= k_S^{-1} * G_S * k_R^{-1} * k_S \\ &= k_R^{-1} * G_S \end{aligned} \tag{11a}$$

$$\begin{aligned} P_{R2} &= P_{S1} * k_R \\ &= k_R^{-1} * G_R * k_S^{-1} * k_R \\ &= k_S^{-1} * G_R \end{aligned} \tag{11b}$$

When P_{S2} is received by the recipient node, it is multiplied with the recipient's private key and the Generator point of the sending node, G_S is extracted by using equation (12b). After P_{R2} is received by the sending node, it is multiplied with the sender's private key and the Generator point of the receiving node,G_R is extracted by using equation (12a).

$$\boxed{P_{R2} * k_S = k_S^{-1} * G_R * k_S = G_R} \tag{12a}$$

$$\boxed{P_{S2} * k_R = k_R^{-1} * G_S * k_R = G_S} \tag{12b}$$

Both the sender and receiver nodes have the Generator points of each other, G_R and G_S, respectively. Now, the shared key is computed by adding both these Generator points.

$$\boxed{Sharedkey(G_C) = G_S + G_R} \tag{13}$$

After calculating the shared key by using equation (13), the messages between sender and receiver nodes can be securely exchanged. Every time a new shared key is computed as and when another session is established between the nodes.

IV. Hardware Description

The MDA100 sensor board is used in this work. The MDA100 sensor board, as shown in Figure 6, contains precision thermistor, light sensor and general prototyping area. All ADC channels of the node (ADC0-7) can be used to support the sensors mounted over this prototyping area. Various sensor devices can be connected physically within this area.

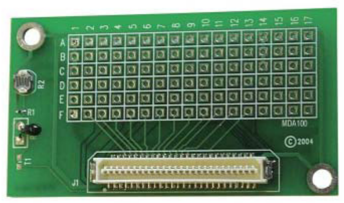

Fig. 6. Sensor board

The IRIS WSN nodes from *MEMSIC*, as shown in Figure 7, have been used. Table III provides the specifications of the IRIS node. The programming board MIB520 as shown in Figure 8, has been made use of in order to program the IRIS node.

Fig. 7. IRIS mote

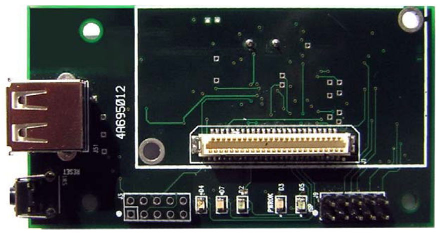

Fig. 8. Programming Board

TABLE III. IRIS WSN NODE SPECIFICATIONS [15]

Parameter	Value
RAM	8 KB
ROM	128 KB
Clock Frequency	7.37 MHz
Supply voltage	2.7 to 3 V
TX Power	-17 to 3 dBm
TX current consumption	16 mA
RX current consumption	15 mA

V. COMPARISON

The EC $E_P(a, b)$ parameters in the 192- bit NIST field are as follows:
P_{192} = ffff ffff ffff ffff ffff ffff ffff fffe ffff ffff ffff ffff
a = ffff ffff ffff ffff ffff ffff ffff fffe ffff ffff ffff fffc
b = 6421 0519 e59c 80e7 0fa7 e9ab 7224 3049 feb8 deec c146 b9b1

In this work, three different protocols for hiding the generator point in the WSN implementation of ECC are presented. The IRIS WSN nodes, used for the implementation of the protocols considered, are supported by an ATmega 128L 8- bit micro-controller. Hence, the prime field arithmetic operations like addition, subtraction, multiplication and division need to make use of the 8-bit CPU. The 192- bit values are stored in 24 * 8 array. The extended Euclidean algorithm is used to carry out the inversion operation. All the EC point operations, point addition, point doubling and scalar multiplication are implemented using IRIS nodes. The point addition and point doubling are implemented using Affine and Projective coordinate systems. The scalar multiplication is carried out using Binary method and Sliding window method. The Projective co-ordinate system is used to implement the scalar multiplication operation while implementing the three protocols.

In order to carry out the implementation of these three protocols using WSN node, few EC points on the $E_P(a, b)$ in the 192-NIST field using equation (1) are generated and these are stored in each of the nodes prior to the deployment of the network. While selecting the points to be stored it has been ensured that these are of higher order. The nodes are supported with TinyOS operating system and the application development has been carried using nesC language.

TABLE IV. COMPARISON OF THE TIME CONSUMED BY THE PROTOCOLS

Protocol Type	Time in seconds
First protocol	39
Second protocol	79
Proposed protocol	123

In an example using the proposed protocol, a shared key, $G_C(x_C, y_C)$ has been generated and the same is given as:
x_C = 2ca3 ae14 478c 6bc1 a470 7a48 e165 c5a0 42bc ad3d 2efb d8ab
y_C = e811 f047 757e beee 379e f078 4279 5ded 1b81 2bef 0a9c 2518

The implementation of the first protocol involves the same number of computations as in the ECC. This approach needs a certificate authority. If the information received from the CA is compromised, the network security also gets breached and making the selection of generator point vulnerable. The security level supported by this protocol tends to be low.

The implementation of the second protocol involves a new generator point selection as and when a new message is to be sent, resulting in improved security level. However, the improvement in the security involves higher number of computations.

The proposed protocol does not use a common generator point between the sender and and the receiver. Both the nodes select their generator points independently. This protocol proposes a unique scheme to find a shared key between the nodes without needing a common generator point between the nodes. Once the shared key is generated, the same can be used while performing encryption and decryption using ECC at both the sender and recipient nodes. This method requires more computations when compared with that of the first protocol and its security level is slightly less than that of the second protocol.

Whenever a communication session between two nodes involving data exchange needs to be established, both the nodes generate a shared key independently by using the messages, namely P_{S1}, P_{R1}, P_{S2}, P_{R2}. Using this shared key, the data to be forwarded is encrypted using ECC at the sender. The receiver node decrypts the received cipher data using the same shared key and ECC. After forwarding the data, the sender and receiver nodes may get into sleep state by turning off their radio transceivers. As when either of nodes need to send data to other node, a new shared key needs to be generated as it is going to be new session. As an example, it is considered that data messages are exchanged between the nodes 10 times in each communication session. Table V provides the number of basic EC operations performed by a node during one session.

TABLE V. COMPARISON OF EC OPERATIONS REQUIRED BY THE THREE PROTOCOLS FOR MIM ATTACK

Protocol type	Number of Scalar Multiplications	Number of Point Additions
First protocol	2	10
Second protocol	30	10
Proposed protocol	4	11

As can be observed from Table V, first protocol requires least number of computations and the second protocol involves highest number of computations. The number of computations required by the proposed protocol is very close to the first protocol, which also provides good amount of security when compared with that of the second protocol.

VI. CONCLUSION

As the nodes in a WSN are vulnerable to physical node capture, the stored data and other parameters and their values in the nodes may become accessible to an attacker and thereby leading to the MIM attack. The MIM attack is overcome by implementing the ECC using WSN nodes by hiding the generator point. Based on the comparison of the three protocols considered in terms of the number of the EC computations required by each one, the proposed protocol can be considered as the most optimum in terms of computations. However, the communication costs are on the higher side as multiple messages are exchanged at the beginning of a session only once. It provides additional security as it involves a new shared key being generated at the beginning of each session. All these protocols and scalar multiplication techniques have been implemented using IRIS WSN nodes from *MEMSIC*.

REFERENCES

[1] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: a survey," *Computer networks*, vol. 38, no. 4, pp. 393–422, 2002.

[2] C. T. Inc., "Micaz wireless measurement system," June, 2004.

[3] G. Werner-Allen, K. Lorincz, M. Ruiz, O. Marcillo, J. Johnson, J. Lees, and M. Welsh, "Deploying a wireless sensor network on an active volcano," *Internet Computing, IEEE*, vol. 10, no. 2, pp. 18–25, 2006.

[4] M. Shand and J. Vuillemin, "Fast implementations of rsa cryptography," in *Computer Arithmetic, 1993. Proceedings., 11th Symposium on.* IEEE, 1993, pp. 252–259.

[5] N. Gura, A. Patel, A. Wander, H. Eberle, and S. Shantz, "Comparing elliptic curve cryptography and rsa on 8-bit cpus," *Cryptographic Hardware and Embedded Systems-CHES 2004*, pp. 925–943, 2004.

[6] X. Huang, P. G. Shah, and D. Sharma, "Protecting from attacking the man-in-middle in wireless sensor networks with elliptic curve cryptography key exchange," in *Network and System Security (NSS), 2010 4th International Conference on.* IEEE, 2010, pp. 588–593.

[7] D. Hankerson, A. Menezes, and S. Vanstone, *Guide to elliptic curve cryptography.* Springer, 2004.

[8] M. Brown, D. Hankerson, J. López, and A. Menezes, *Software implementation of the NIST elliptic curves over prime fields.* Springer, 2001.

[9] R. K. Kodali and H. S. Budwal, "High performance scalar multiplication for ecc," in *Computer Communication and Informatics (ICCCI), 2013 International Conference on.* IEEE, Jan 2013, pp. 1–4.

[10] O. YAYLA, "Scalar multiplication on elliptic curves," Ph.D. dissertation, Masters Thesis, Department of Cryptography, Middle East Technical University, August 2006. Available at http://www3. iam. metu. edu. tr/iam/images/3/3e/O% C4% 9Fuzyaylathesis. pdf(link tested 01-Dec-2011), 2006.

[11] X. Huang, D. Sharma, and H. Cui, "Fuzzy controlling window for elliptic curve cryptography in wireless sensor networks," in *Information Networking (ICOIN), 2012 International Conference on.* IEEE, 2012, pp. 312–317.

[12] X. Huang, D. Sharma, and P. Shah, "Efficiently fuzzy controlling with dynamic window in elliptic curve cryptography sensor networks," in *Proceeding of the International MultiConference of Engineers and Computer Scientists*, vol. 1, 2011.

[13] X. Huang, P. G. Shah, and D. Sharma, "Fast scalar multiplication for elliptic curve cryptography in sensor networks with hidden generator point," in *Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC), 2010 International Conference on.* IEEE, 2010, pp. 243–249.

[14] R. K. Kodali, "Implementation of ecc with hidden generator point in wireless sensor networks," in *Communication Systems and Networks (COMSNETS), 2014 Sixth International Conference on*, Jan 2014, pp. 1–4.

[15] K. R. Kishore, "Elliptic curve based digital envelope in wsn," in *Advanced Electronic Systems (ICAES), 2013 International Conference on.* IEEE, Sep 2013, pp. 292–296.