Scalar Multiplication of a Dynamic Window with Fuzzy Controller for Elliptic Curve Cryptography

Xu Huang
Faculty of Information Sciences and Engineering
University of Canberra
Canberra, Act 2601 Australia
Xu.Huang@canberra.edu.au

John Campbell
Faculty of Information Sciences and Engineering
University of Canberra
Canberra, Act 2601 Australia
John.Campbell@canberra.edu.au

Frank Gao
Faculty of Information Sciences and Engineering
University of Canberra
Canberra, Act 2601 Australia
Frank.Gao@canberra.edu.au

Abstract—**Elliptic curve cryptography (ECC) provides solid potential for wireless sensor network security due to its small key size and its high security strength. However, there is a need to reduce key calculation time to satisfy the full range of potential applications, in particular those involving wireless sensor networks (WSN). Scalar multiplication operation in elliptical curve cryptography accounts for 80% of key calculation time on wireless sensor network motes. In this paper, two major contributions are made: (a) we propose an algorithm based on 1's complement subtraction to represent scalar in scalar multiplication which offer less Hamming weight and will significantly improve the computational efficiency of scalar multiplication; and (b) we present a fuzzy controller for dynamic window sizing to allow the program to run under optimum conditions by allocating available RAM and ROM at the sensor node within a wireless sensor network. The simulation results showed that the average calculation time decreased by approximately 15% in comparison to traditional algorithms in an ECC wireless sensor network.**

Keywords- Elliptic curve cryptography (ECC), scalar multiplication, non-adjacent form, Hamming weight, one's complement subtraction, fuzzy controller

I. INTRODUCTION

Rapid growth in very large scale integrated (VLSI) technology, embedded systems and micro electro mechanical systems (MEMS) has enabled production of inexpensive sensor nodes which can communicate information over shorter distances with efficient use of power [1]. The sensor node detects information, processes it with the help of an in-built microcontroller and communicates results to a sink or base station. The base station is a more powerful node linked which can be linked to a central station via satellite or internet communication. Wireless sensor networks can be deployed in various applications including environmental monitoring e.g. volcano detection [2,3], distributed control systems [4] , detection of radioactive sources [5], agricultural and farm management [6], and computing platform for tomorrows' internet[7] .

Xu Huang is at the Faculty of Information Sciences and Engineering, University of Canberra, Australia.
John Campbell is at the Faculty of Information Sciences and Engineering, University of Canberra, Australia.
Frank Gao is a PhD candidate at the Faculty of Information Sciences and Engineering, University of Canberra, Australia

Compared to traditional networks, a wireless sensor network has many resource constraints [4]. The MICA2 mote consists of an 8 bit ATMega 128L microcontroller working on 7.3 MHz. As a result nodes of WSN have limited computational power. Normally, radio transceiver of MICA motes can achieve maximum data rate of 250 Kbits/sec which restricts available communication resources. The flash memory which is available on the MICA mote is only 512 Kbyte. Apart from these limitations, the onboard battery is 3.3.V with 2A-Hr capacity. Due to the above restrictions the current state of art protocols and algorithms are expensive for sensor networks due to their high communication overhead.

Elliptic Curve Cryptography was introduced by Victor Miller [9] and Neal Koblitz [10] independent of each other in the early eighties. The advantage of ECC over other public key cryptography techniques such as RSA, Diffie-Hellman is that the best known algorithm for solving ECDLP which is the underlying hard mathematical problem in ECC which will take the fully exponential time. On the other hand the best algorithm for solving RSA and Diffie-Hellman takes sub exponential time [11]. In summary, the ECC problem can only be solved in exponential time and, to date, there is a lack of sub exponential methods to attack ECC.

An elliptic curve E over $GF(p)$ can be defined by $y^2 = x^3 + ax + b$ where $a, b \in GF(p)$ and $4a^3 + 27b^2 \neq 0$ in the $GF(p)$.

The point (x, y) on the curve satisfies the above equation and the point at infinity denoted by ∞ is said to be on the curve.

If there are two points on the curve namely, $P(x_1, y_1)$, $Q(x_2, y_2)$ and their sum is given by point $R(x_3, y_3)$ the algebraic formulas for point addition and point doubling are given by following equations:

$$x_3 = \lambda^2 - x_1 - x_2$$

$$y_3 = \lambda(x_1 - x_3) - y_1$$

$$\lambda = \frac{y_2 - y_1}{x_2 - x_1}, \quad \text{if } P \neq Q$$

978-0-7695-4159-4/10 $26.00 © 2010 IEEE
DOI 10.1109/NSS.2010.16

$$\lambda = \frac{3x^2 + a}{2y_1}, \quad \text{if } P = Q$$

Where the addition, subtraction, multiplication and inverse are the arithmetic operations over $GF(p)$.

II. EASE ELLIPTIC CURVE DIFFIE-HELLMAN SCHEME (ECDH) PROPOSED FOR WSN

As per [13] the original Diffie-Hellman algorithm with RSA requires a key of 1024 bits to achieve sufficient security but *Diffie Hellman based on ECC* can achieve the same security level with only 160 bit key size.

Initially Alice and Bob agree on a particular curve with base point P. They generate their public keys by multiplying P with their private keys namely K_A and K_B. After sharing public keys, they generate a shared secret key by multiplying public keys by their private keys. The secret key is $R = K_A * Q_A = K_B * Q_A$. With the known values of Q_A, Q_B and P, it is computationally intractable for an eavesdropper to calculate K_A and K_B which are the private keys of Alice and Bob. As a result, adversaries cannot calculate R the shared secret key.

In ECC two heavily used operations are involved: scalar multiplication and modular reduction. Gura et al [14] showed that 85% of execution time is spent on scalar multiplication. Scalar Multiplication is the operation of multiplying point P on an elliptic curve E defined over a field $GF(p)$ with positive integer k which involves point addition and point doubling. Operational efficiency of kP is affected by the type of coordinate system used for point P on the elliptic curve and the algorithm used for recoding of integer k in scalar multiplication.

This research paper proposes an innovative algorithm based on one's complement for representation of integer k which accelerates the computation of scalar multiplication in wireless sensor networks.

The number of point doubling and point addition operations in scalar multiplication depends on the recoding of integer k. Expressing integer k in binary format highlight this dependency.

The number of zeros and number of ones in the binary form, their places and the total number of bits will affect the computational cost of scalar multiplications. The Hamming weight as represented by the number of non-zero elements, determines the number of point additions and bit length of integer K determines the number of point doublings operations in scalar multiplication.

One point addition when $P \neq Q$ requires one field inversion and three field multiplications [13]. Squaring is counted as regular multiplication. This cost is denoted by $1I + 3M$, where I denotes the cost of inversion and M denotes the cost of multiplication.

One point doubling when $P = Q$ requires $1I + 4M$ as we can neglect the cost of field additions as well as the cost of multiplications by small constant 2 and 3 in the above formulae.

III. THE EXISTING METHODS OF SCALAR MULTIPLICATION

Before we introduce our new method the existing methods of scalar multiplication need to be briefly mentioned.

Binary Method: Scalar multiplication is the computation of the form $Q = kP$, where P and Q are the elliptic curve points and k is positive integer. This is obtained by repeated elliptic curve point addition and doubling operations. In binary method the integer k is represented in binary form:

$$k = \sum_{j=0}^{l-1} K_j 2^j, \; K_j \in \{0,1\}$$

The binary method scans the bits of K either from left-to-right or right-to-left. The binary method for the computation of kP is given in the following *algorithm 1*, as shown below:

Algorithm 1: Left to right binary method for point multiplication

Input: A point $P \in E(F_q)$, an

$$k = \sum_{j=o}^{l-1} K_j 2^j, \; K_j \in \{0,1\}$$

l bits integer

Output: $Q = kP$

1. $Q \leftarrow \infty$
2. For $j = l - 1$ to 0 do:
   - 2.1 $Q \leftarrow 2Q$,
   - 2.2 if $k_j = 1$ the $Q \leftarrow Q + P$.
3. Return Q.

The cost of multiplication when using binary method depends on the number of non-zero elements and the length of the binary representation of k. If the representation has $k_{l-1} \neq 0$ then binary method require $(l - 1)$ point doublings and $(W\text{-}1)$ where l is the length of the binary expansion of k, and W is the Hamming weight of k (ie, the number of non-zero elements in expansion of k). For example, if $k = 629 = (1001110101)_2$, it will require $(W\text{-}1) = 6 - 1 = 5$ point additions and $l - 1 = 10 - 1 = 9$ point doublings operations.

Signed Digit Representation Method: The subtraction has virtually the same cost as addition in the elliptic curve group. The negative of point (x, y) is $(x, -y)$ for odd characters. This leads to scalar multiplication methods based on addition – subtraction chains, which help to reduce the number of curve operations. When integer k is represented with the following form, it is a *binary signed digit representation*.

$$k = \sum_{j=0}^{l} S_j 2^j, \; S_j \in \{1,0,-1\}$$

When a signed-digit representation has no adjacent non zero digits, i.e. $S_j S_{j+1} = 0$ for all $j \geq 0$ it is called a non-adjacent form (NAF) .

The following *algorithm 2* computes the NAF of a positive integer given in binary representation.

Algorithm 2: Conversion from Binary to NAF

Input: An integer $k = \sum_{j=0}^{l-1} K_j 2^j,\ K_j \in \{0,1\}$

Output: NAF $k = \sum_{j=0}^{l} S_j 2^j,\ \ S_j \in \{1,0,-1\}$

1. $C_0 \leftarrow 0$
2. *For j = 0 to l do:*
3. $C_{j+1} \leftarrow [(K_j + K_{j+1} + C_j)/2]$
4. $S_j \leftarrow K_j + C_j - 2C_{j+1}$
5. *Return* $(S_l \ldots S_0)$

NAF usually has fewer non-zero digits than binary representations. The average hamming weight for NAF form is $(n-1)/3.0$. So generally it requires $(n-1)$ point doublings and $(n-1)$ /3.0 point additions. The binary method can be revised accordingly and is given another algorithm for NAF, and this modified method is called the *Addition Subtraction* method.

IV. Proposed Algorithm Based on One's Complement for Recoding of Scalar *K*

A subtraction by utilization of the 1's complement is most common in binary arithmetic. The 1's complement of any binary number may be found by the following equation [19]:

$$C_1 = (2^a - 1) - N \qquad \text{(I)}$$

where C_1 = 1's complement of the binary number, a = number of bits in N in terms of binary form, N = binary number

A close observation of the equation (I) reveals the that any positive integer can be represented by using minimal non-zero bits in its 1's complement form provided that it has a minimum of 50% Hamming weight. The minimal non-zero bits in positive integer scalar are very important to reduce the number of intermediate operations of multiplication, squaring and inverse calculations used in elliptical curve cryptography as we have seen in previous sections.

The equation (I) can therefore be modified as per below:

$$N = (2^a - C_1 - 1) \qquad \text{(II)}$$

For example, let us take N =1788

N= $(11011111100)_2$ in its binary form

C_1= 1's Complement of the number of N= $(00100000011)_2$

a is in binary form so we have a = 11

After putting all the above values in the equation II we derive:

1788 = 2^{11} − 00100000011 −1, this can be reduced as below:

1788 = 100000000000−00100000011−1 (III)

So we have

1788= 2048 − 256− 2− 1−1

As is evident from equation III, the Hamming weight of scalar N has reduced from 8 to 5 which will save 3 elliptic curve addition operations. One addition operation requires 2 Squaring, 2 Multiplication and 1 inverse operation. In this case a total of 6 Squaring, 6 Multiplication and 3 Inverse operations will be saved.

The above recoding method based on one's complement subtraction combined with sliding window method provides a more optimized result.

Let us compute [763] P (in other words k = 763) as an example, with a sliding window algorithm with K recoded in binary form and window sizes ranging from 2 to 10. It is observed that as the window size increases the number of pre-computations also increases geometrically. At the same time number of additions and doubling operations decrease.

Algorithm for sliding window scalar multiplication on elliptic curves.

---

1. $Q \leftarrow P_\infty$ *and* $i \leftarrow l$ - 1
2. while $i \geq 0$ do
3. if $n_i = 0$ then $Q \leftarrow [2]Q$ and $i \leftarrow i - 1$
4. else
5. $s \leftarrow \max(i - k + 1, 0)$
6. *while* $n_s = 0$ do $s \leftarrow s + 1$
7. for $h = 1$ to $i - s + 1$ do $Q \leftarrow [2]Q$
8. $u \leftarrow (n_i \ldots\ldots n_s)_2$ $[n_i = n_s = 1$ and $i - s + 1 \leq k]$
9. $Q \leftarrow Q \oplus [u]P$ [u is odd so that [u]P is precompute d]
10. $i \leftarrow s - 1$
11. return Q

Now we present the details for the different window size to find out the optimal window size using the following example:

Window Size *w* = 2

763 = $(1011111011)_2$

No of precomputations = $2^w - 1 = 2^2 - 1 = [3]\,P$

763 = 10 11 11 10 11

The intermediate values of Q are

P, 2P 4P, 8P, 11P, 22P, 44P, 47P, 94P, 95P, 190P, 380P, 760P, 763P

Computational cost = 9 doublings, 4 additions, and 1 pre-computation.

Window Size $w = 3$

No of pre-computations = $2^w - 1 = 2^3 - 1 = [7]\ P$

So all odd values: [3]P, [5]P, [7]P

763 = 101 111 101 1

= [5]P [7]P [5]P [1]P

The intermediate values of Q are

5P, 10P, 20P, 40P, 47P, 94P, 188P, 376P, 381P, 762P, 763P

Computational cost = 7 doublings, 3 additions, and 3 pre-computations.

We continue to derive the remaining calculations for Window Size $w = 6$, Window Size $w = 7$, Window Size $w = 8$, Window Size $w = 9$, and Window Size $w = 10$. The results for all calculations are presented in Table 1.

The effects of "doublings" and "additions" as shown in Table 1 are further considered below.

Figure 1 shows the trade off between window size and number of pre-computations. It is clear that we need to pay particular attention to window size by taking account of the calculations performed. In particular the doubling operations will guide the discussion in the next section where we will consider a fuzzy logical controller for selecting a dynamic optimal window size to minimize the calculation cost of ECC within the resource limitations of wireless sensor networks.

Table 1: Window Size Vs No of doublings, additions and Pre computations

Window Size	No of Doublings	No of Additions	No of Pre computations
2	9	4	1
3	7	3	3
4	6	2	7
5	5	1	15
6	4	1	31
7	3	1	61
8	3	1	127
9	1	1	251
10	0	0	501

V. Fuzzy Controller System in ECC

the It is clear, from above description that there is a tradeoff between the computational cost and the window size as shown in Table 1. However, this tradeoff is underpinned by the balance between computing cost (or the RAM cost) and the pre-computing (or the ROM cost) of the node in the network.

It is also clear that, from above description that the variety of wireless network working states will make this control complex and calculations could be relatively more expensive.

Therefore, we propose a fuzzy dynamic control system, to provide dynamic control to ensure the optimum window size is obtained by tradeoff between pre-computation and computation cost.

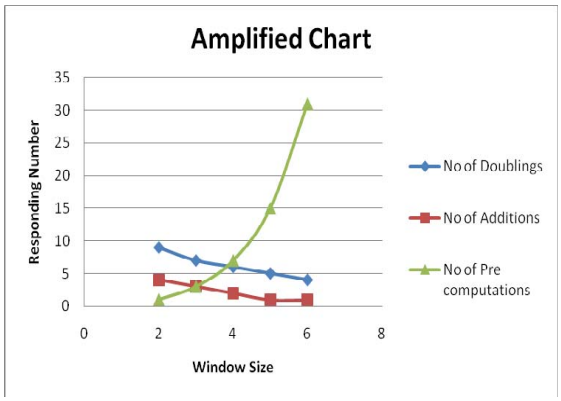


Figure 1: Part of the chart of Table one

The fuzzy decision problem introduced by Bellman and Zadeh [20], it has a goal of the maximization of the minimum value of the membership functions of the objectives to be optimized. Accordingly, the fuzzy optimization model can be represented as a multi-objective programming problem as follows [21]:

$$Max: \min\{\mu_s(D)\}\ \&\ \min\{\mu_l(U_l)\}\ \ \forall s \in S\ \&\ \forall l \in L$$

$$\text{such that}\quad A_l \le C_l \qquad \forall l \in L,$$

$$\sum_{r \in R_p} x_{rs} = 1 \qquad \forall p \in P\ \&\ \forall s \in S,$$

$$x_{rs} = 0 \text{ or } 1 \qquad \forall r \in R\ \&\ \forall s \in S$$

In above equation, the objective is to maximize the minimum membership function of all delays, denoted by D, and the difference between the recommend value and the measured value, denoted by U.

The fuzzy control system is extended from [20] and shown in Figure 2. For accurate control, we designed a two input fuzzy controller. This involved a tradeoff between accuracy and control costs. Using a one input fuzzy controller would be much less costly than a two input controller, but would not return the accuracy level we require. The first estimated angle is input and then the set motor controller will run following the feedback values from the photon receiver and after comparisons. There are three outputs: anti-clock (or called "negative"); no change (or called "zero"); and clockwise (or called "positive"). Two ranges are defined, namely "negative" = -100 to 0.0) and "positive" = 0.0 to +100.

There are two inputs for this control system, namely (1) the "difference" of the window size between the pre-existing value and the measured value. When these two values are perfectly matched, the output of the controller is recognized and this "difference" ideal value will be zero. (2) the "differential" of

the difference that describes the "change rate" and indicates the direction of change (window size is increasing or decreasing).

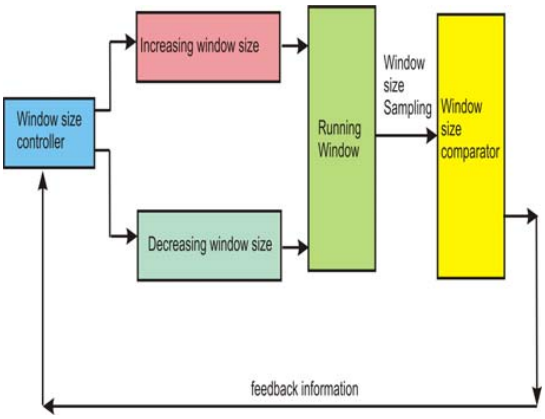
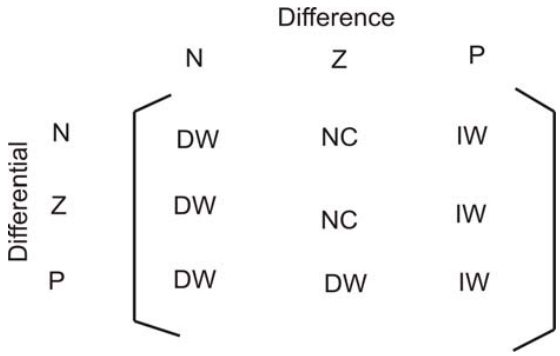


Figure 2: Two inputs fuzzy window control system

For the fuzzy logic rule structure and matrix, we have:

1. If the pre-existing window size – measured value = negative and *d*(pre-existing window size – measured value)/*dt* = negative THEN output = decreasing window size.
2. If the pre-existing window size – measured value = zero and *d*(pre-existing window size – measured value)/*dt* = negative THEN output = no change
3. If the pre-existing window size – measured value = positive and *d*(pre-existing window size – measured value)/*dt* = negative THEN output = increasing window size
4. If the pre-existing window size – measured value = negative and *d*(pre-existing window size – measured value)/*dt* = zero THEN output = decreasing window size
5. If the pre-existing window size – measured value = zero and *d*(pre-existing window size –measured value)/*dt* = zero THEN output = no change (zero)
6. If the pre-existing window size – measured value = positive and *d*(pre-existing window size – measured value)/*dt* = zero THEN output = increasing window size
7. If the pre-existing window size – measured value = negative and *d*(pre-existing window size – measured value)/*dt* = positive THEN output = decreasing window size
8. If the pre-existing window size – measured value = zero and *d*(pre-existing window size – measured value)/*dt* = positive THEN output = decreasing window size
9. If the pre-existing window size – measured value = positive and *d*(pre-existing window size – measured value)/*dt* = positive THEN output = increasing window size

Therefore, we can construct a matrix from the nine rules as shown in below:



where DW = decreasing window size, IW = increasing window size, and NC= no change (or zero).

As an example, we have two inputs: "difference" = ─1.0 and "differential" = +2.5, as shown in Figure 3.

As another example, consider when the "difference = -2.25" and the "differential = +2.5". This gives a calculated result of ─59.6% which calls for a decreasing window as shown in Figure 3. The defuzzification process for clear output is provided by "root sum square" (RSS).

Some final testing results are shown below:

1. Two inputs: "difference" = -3 and "differential" = +1.25 then output = -55.1%, decreasing window size;
2. Two inputs: "difference" = 0 and "differential" = +2.5 then output = -50.1%, decreasing window size;
3. Two inputs: "difference" = +3 and "differential" = +2.5 then output = 10.8%, Increasing window size.
4. Two inputs: "difference" = +1 and "differential" = +2.5 then output = +12.1%, increasing window size;

The dynamic control system works and the result is fairly reasonable, which allows sufficient photon to be communicated by the wireless channel.

Finally we apply the proposed algorithm to the same number 763 to show the effectiveness of algorithm with a window size of 3.

As we know that

$763 = (10111111011)_2$

Consider record 763 with equation II in one's complement subtraction form and we have:

763 = 10000000000 – 0100000100-1

=10100000101

With window *w* size of 3, we obtain:

763 = 10<u>1</u> 000000 <u>101</u>

Here "1" means "-1"

Now let us apply the proposed algorithm to the same number 763 to show the effectiveness of our algorithm with window size of 3.

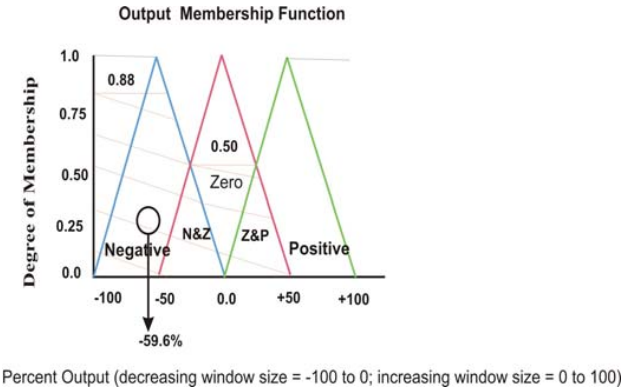


Figure 3: An example for the fuzzy calculation. The Output of the Fuzzy controller.

The intermediate values of Q are:

3P, 6P, 12P, 24P, 48P, 96P, 192P, 384P, 768P, 763P

Hence the Computational Cost = 8 doublings, 1 addition and 3 pre-computations.

With equation II, the computational cost has been reduced from 3 additions as in the binary method to only 1 addition in one's complement subtraction form. The number of pre-computations has remained the same. This can be proved for different window sizes.

In our simulations, the proposed method together with a fuzzy window size controller makes the ECC calculation almost 15% more efficient than traditional methods in ECC wireless sensor network. As mentioned above, the use one input fuzzy may result in further efficiencies. This possibility will be discussed in another paper.

VI. CONCLUSION

The positive integer in point multiplication may be recoded with *one's complement subtraction* to reduce the computational cost involved in this heavy mathematical operation for wireless sensor network platforms. The window size may be the subject of trade-off between the available RAM and ROM at a particular instance on a sensor node. As the NAF method involves modular inversion operation to get the NAF of binary number, the one's complement subtraction can provide a very simple way of recoding the integer.

REFERENCES

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: a survey," *Computer Networks,* vol. 38, pp. 393-422, 2002.

[2] C. Chung-Kuo, J. M. Overhage, and J. Huang, "An application of sensor networks for syndromic surveillance," 2005, pp. 191-196.

[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," *IEEE Internet Computing,* vol. 10, pp. 18-25, 2006.

[4] B. Sinopoli, C. Sharp, L. Schenato, S. Schaffert, and S. S. Sastry, "Distributed control applications within sensor networks," *Proceedings of the IEEE,* vol. 91, pp. 1235-1246, 2003.

[5] D. L. Stephens, Jr. and A. J. Peurrung, "Detection of moving radioactive sources using sensor networks," *Nuclear Science, IEEE Transactions on,* vol. 51, pp. 2273-2278, 2004.

[6] P. Sikka, P. Corke, P. Valencia, C. Crossman, D. Swain, and G. Bishop-Hurley, "Wireless ad hoc sensor and actuator networks on the farm," 2006, pp. 492-499.

[7] Z. Feng, "Wireless sensor networks: a new computing platform for tomorrow's Internet," 2004, pp. I-27 Vol.1.

[8] I. F. Akyildiz, Weilian Su, Yogesh Sankarasubramaniam, and E. Cayirci, "A Survey on Sensor Networks" in *IEEE Communication Magazine*. vol. 40, August 2002, pp. 102-116.

[9] V. S. Miller, "Use of Elliptic Curves in Cryptography," in *Advances in Cryptology - CRYPTO '85: Proceedings*. vol. 218: Springer-Verlag, 1986, pp. 417-426.

[10] N.Koblitz, "Elliptic Curve Cryptosystems," *Mathematics of Computation,* vol. 48, pp. 203-209, 1987.

[11] J. Lopez and R. Dahab., " An overview of elliptic curve cryptography," Technical report ,Institute of Computing, Sate University of Campinas, Sao Paulo, Brazil, May 2000.

[12] K. Lauter, "The advantages of elliptic curve cryptography for wireless security," *Wireless Communications, IEEE [see also IEEE Personal Communications],* vol. 11, pp. 62-67, 2004.

[13] H. Wang, B. Sheng, and Q. Li, "Elliptic curve cryptography-based access control in sensor networks," *Int. J. Security and Networks,,* vol. 1, pp. 127-137, 2006.

[14] N. Gura, A. Patel, and A. Wander, "Comparing elliptic curve cryptography and RSA on 8-bit CPUs," in *Proceedings of the 2004 Workshop on Cryptographic Hardware and Embedded Systems (CHES)* August 2004.

[15] http://csrc.nist.gov/CryptoToolkit/dss/ecdsa/NISTReCur.pdf."

[16] D. J. Malan, M. Welsh, and M. D. Smith, "A public-key infrastructure for key distribution in TinyOS based on elliptic curve cryptography," in *2nd IEEE International Conference on Sensor and Ad Hoc Communications and Networks (SECON 2004)2nd IEEE International Conference on Sensor and Ad Hoc Communications and Networks (SECON 2004)*, 2004, pp. 71-80.

[17] I. Blake, G. Seroussi, and N. Smart, *Elliptic Curves in Cryptography* vol. 265, 1999.

[18] D. Hankerson, J. L. Hernandez, and A. Menezes, "Software Implementation of Elliptic Curve Cryptography over Binary Fields, CHES," 2000.

[19] Angelo C Gillie, "Binary Arithmetic and Boolean algebra," McGRAW-HILL Book Company, 1965. pp53.

[20] H R. Bellman and L.A. Zadeh, Decision-making in a fuzzy environment, Management Science 17 (1970),141-164.

[21] Xu Huang, Shirantha Wijesekera, and Dharmendra Sharma, "Fuzzy Dynamic Switching in Quantum Key Distribution for Wi-Fi Networks," 6th International Conference on Fuzzy Systems and Knowledge Discovery, 14-16 August 2009, Tianjin, China. Proceeding pp302-306.