TNC Transactions on Networks and Communications
SOCIETY FOR SCIENCE AND EDUCATION UNITED KINGDOM
VOLUME 3, ISSUE 4
ISSN: 2054 - 7420

Light Weight Secure Key Generation protocol with Hidden Generator point using ECC

[1]Ayaz Hassan Moon, [2]Ummer Iqbal
National Institute of Electronics and Information Technology (NIELIT), J&K, INDIA
moonah@rediffmail.com, ummer@nielit.gov.in

ABSTRACT

Key generation and distribution is one of the most important primitive of any security framework. This is irrespective of using a symmetric or asymmetric cryptosystem. However, while securing a WSN, its resource constraint nature cannot be ignored. Therefore Elliptical Curve Cryptography (ECC) based solutions like Elliptical Curve Digital Signature algorithm (ECDSA), Elliptical Curve Diffie-Hellman (ECDH) are becoming more and more popular in comparison to other Public crypto system like RSA. In ECC, the Generator point is treated as a public parameter along with other domain parameters. This can make communication within the WSN vulnerable to man-in-the-middle attack. The attack can be thwarted by keeping the Generator point Private and still be able to establish a common Generator Point across communication parties .It will result in establishing a light weight secure key between a sender and a receiver and achieve other security primitives like generation of MAC and Node identification. This paper discusses and analyses the generation of Shared keys using 1 hidden generator point in comparison to 2- hidden point generator and the conventional ECDH method.

Keywords: Key establishment, Hidden generator, Node Identification, Authentication, WSN, ECC

1 Introduction

Wireless communication being broadcast in nature is more prone to different kind of attacks like eavesdropping, intercept, inject and alter transmitted data. Traditional security solutions based upon public key cryptography are not suitable for wireless sensor networks [1-2]. In conventional networks, message authentication, data integrity and confidentiality are usually achieved by end-to-end security mechanism like SSH, SSL, IP-Sec etc. In end-to-end communication, it is neither necessary nor desirable for the contents of the message (beyond the necessary headers) to be made available to the intermediate routers [3].

The most common security services to be considered for WSN include Confidentiality, Authentication, Integrity, Freshness, Availability, Intrusion detection. In realizing the objectives of the most of the security primitives, Key Management is rightly regarded as the linchpin of Cryptographic mechanism.

Adoption of ECC as an alternative cryptosystem to popular public algorithm like RSA has emerged very strongly in WSN based applications. In ECC, the generator point is to be advertised publically along with other domain parameters. This can increase the vulnerability of the node-to-node communication to be subject to man-in-the-middle-attack. One of the possible solutions to this problem is to keep Generator point Hidden and still arrive at a common shared key between communicating parties.

DOI: 10.14738/tnc.34.1295
Publication Date: 15th July 2015
URL: http://dx.doi.org/10.14738/tnc.34.1295

To establish authenticated communication between sensor nodes, secure key distribution and sharing is imperative. Secure key distribution and sharing in WSN is a research challenge. Most sensor node key exchange requires key distribution before deployment. According to [4], easiest key distribution method is to equip all nodes with same key for establishing communication. But in the event of node capture, entire network is comprised.

The rest of the paper is organized as following:

Related Work, Man-in-the-middle-attack, Suitability of ECC, Suggested protocol, Performance Benchmarking, Conclusion.

2 Related Work

Key generation could be either probabilistic, deterministic or hybrid. Zhu et al proposed Localized Encryption and Authentication Protocol (LEAP) a key management protocol which supports the establishment of four types of keys for each sensor node[5]. It includes an individual key shared with base station, a pair wise key shared with another node, a cluster key shared with multiple neighbouring nodes and a group key which is shared by all the nodes. LEAP provides efficient protocol mechanism for inter-node traffic authentication. LEAP also provides schemes for sensor nodes to establish and update individual keys, pair wise shared keys, cluster keys and group keys, revocation and subsequent rekeying mechanism.

Eschenauer & Gligor proposed random-key pre-distribution scheme that relies on probabilistic key sharing among nodes within the sensor networks. It allots several keys to nodes during Initialization chain [6]. Perrig & Song [7] improved upon security of Esch & Gilgor [6] design by requiring at least two common shared keys for authenticated communication and updating communication keys for subsequent communications.

Trusted server schemes depend on a trusted and secure server such as the base station for key agreements among nodes. The server can be treated as the key distribution centre KDC. The base station is the most appropriate choice for the server and each sensor node stores only an embedded key such that a compromising/captured node cannot reveal much security information about sensor network.

The TinyPK systems described by [8] are designed specifically to allow authentication and key agreement between resource constrained sensors. The agreed upon keys may then be used in conjunction with existing cryptosystems TinySec using Diffie-Hellman key exchange algorithm [9].

Many hybrid broadcast authentication protocols have been proposed which use digital signature in base station or cluster head and use improved MAC in sensor nodes. ZHAO Xin et al have proposed hybrid broadcast authentication protocols (HBA) in wireless sensor networks by selecting Tiny ECC and GBA which is an improved version of μTESLA[10].

Public key cryptography techniques like RSA and elliptic curve cryptography (ECC) were traditionally thought to be impractical for WSN. However recently, several groups have successfully implemented public-key cryptography in WSN. In Gura Etal[4] report both RSA and elliptic curve cryptography is possible using 8 bit CPU with ECC, demonstrating a performance advantage over RSA. ECC's 160 bit keys result in shorter messages compared to 1024 RSA keys.

Xu Huang et al [11] and Ravi Kishore et al [12] demonstrated that the efficiency of ECC implementation is highly dependent on the performance of scalar multiplication. Ravi Kishore et al [13]. Proposed different algorithms based on Hidden generator to overcome man-in-the-middle attack He also suggested 2-point Hidden generator method for arriving at a shared key.

3 Man-In-the Middle Attack

A kind of attack, where in a malicious user/attacker inserts himself between two parties to intercept their active communication. It is a kind of eves dropping which can lead to interception of messages and relaying of wrong messages to both the parties [15]. It can lead to breach of confidentiality, authentication and data integrity and is therefore perceived as a serious threat to any network.

The attacker gains a vantage position by inserting himself between two communicating parties [17]. He can therefore intrude into the communication and inject undesirable communication leading to falsification of data. MIMA attacks lead to session hijacking and is an attack on mutual authentication.

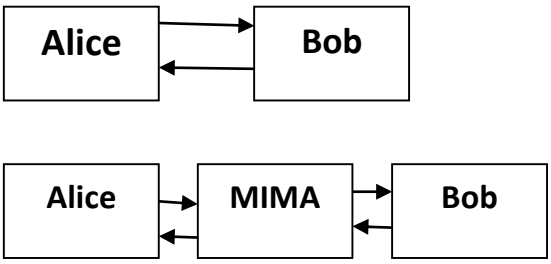

Figure 1: Depicting Man-in-the-middle attack

4 Suitability of ECC

The well-known public crypto systems RSA is based upon modular exponentiation in the integer rings. Its security is derived from the difficulty of factorizing large integers. The solution of integer factorization lies in sub-exponential algorithm [14].

Elliptical Curve Cryptography operates on groups of points over elliptic curve. Its security stems from hardness of elliptic curve discrete logarithmic problem ECDLP. The best known algorithm for solving ECDLP is expo-nential. This implies that attacking ECC is more difficult than attacking RSA. ECC can achieve same level of security as RSA with smaller key size e.g. 160 Bit ECC can provide comparable security to the conventional 1024 Bit RSA. Smaller key size often brings the advantage of faster computation efficiency and saving of bandwidth, memory and energy [14-15]. Therefore ECC is better suited for resource constrained devices like WSN. ECC based ECDSA is used to authenticate new sensor nodes when they join the networks and ECDH (ECC based Diffie-hellmin) algorithm is used to establish shared keys between sensor nodes.

Key length of RSA	Key length of ECC	Ratio of RSA/ECC
512	106	5:1
768	132	6:1
1024	160	7:1
2048	210	10:1

Figure 2: Key comparison between RSA and ECC in terms of security equivalence

4.1 Elliptical Curve Illustration

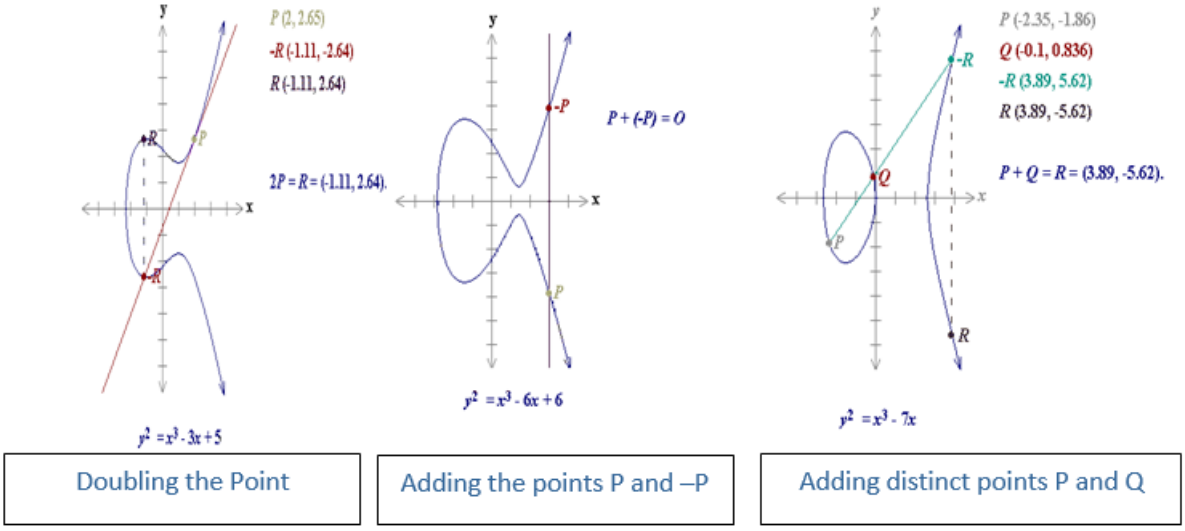

Figure 3: ECC references from www.Certicom.com

4.2 Elliptic Curve Diffie-Hellman (Ecdh) Algorithm

Two parties sharing the same elliptic curve domain parameters can establish a shared secret over an insecure channel without exchanging their respective secret keys. In ECC implementation, the hardness is derived from ECDLP [18]. The flow for establishing the shared secret is as following;

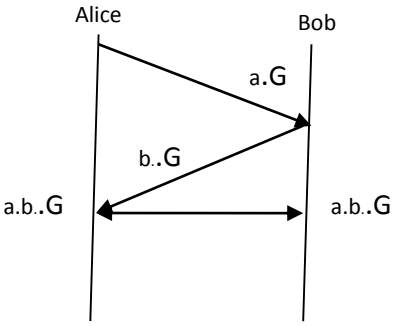

Figure 4: Elliptical curve Diffie-Hellman key generation

5 Suggested Protocols for Key Generation

Two protocols are suggested to implement the key generation with hidden generator points using ECC.

- Key generation using 2-Hidden Generator Points.
- Key generation using 1-Hidden Generator Point.

5.1 Key generation using 2-Hidden Generator Points.

This is similar to the protocol proposed by Ravi[13]. In this protocol, 2 communicating parties i.e Alice and Bob each have their respective hidden generator points Ga and Gb. After selecting their private keys X and Y , they undergo a scalar multiplication with their respective generator points resulting in a X.Ga and Y.Gb both being points on the curve. The parties under take exchanges indicated in the figure 5. and also perform operations based on scalar multiplication and multiplicative inverse . After

6 exchanges, both Alice and Bob are in possession of common generator point G = Ga + Gb , a point on the curve .

This method thwarts the man–in–middle attack as the intruder would not have any access to the either of the generator points lying with bob and Alice, since algorithm leverages the hardness of ECDLP. Extracting Generator points from the scalar multiplicative terms becomes a discrete logarithmic problem which has exponential time complexity.

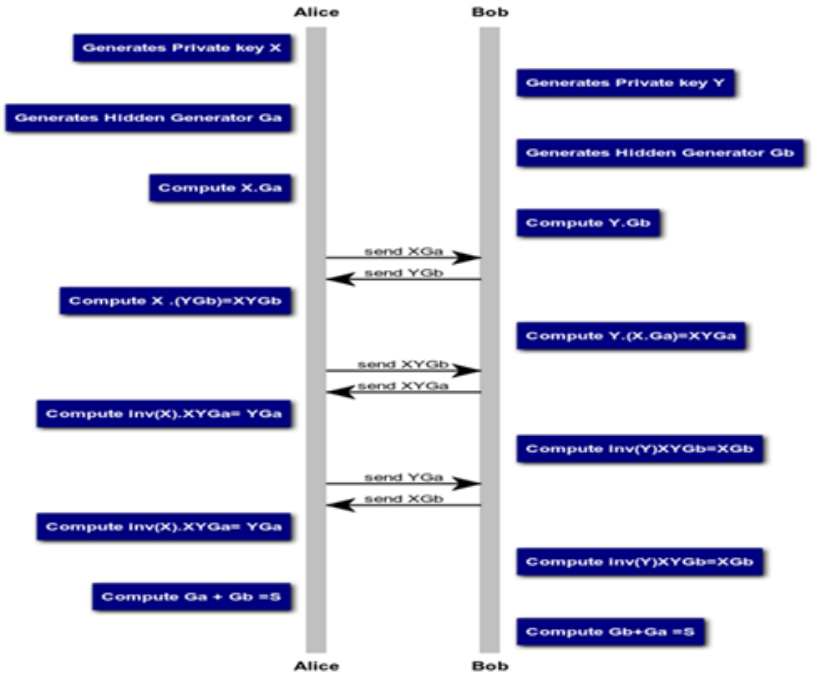

Fig 5: Shared Key generation using 2-hidden generator points

5.1.1 Generation of Shared Key:

After both the communicating parties are in know of G =Ga + Gb , a point on the curve, the following method can be adopted for adoption of a shared key:

G being a point on the curve will have x and y coordinates. Depending upon the curve choosen, the size of these coordinates can be 120, 160 , 192 bits etc.This being a scalar number , can act as a symmetric key between two parties, which can be used as a sessions key for various purposes including distribution of public keys or for encrypting a session. Message Authentication Code (MAC) which is key dependent ,

can also be generated using say x co-ordinate of the common generated point G. The shared key S can be used with any light weight symmetric cipher for achieving authentication.

5.2 Key generation using 1-Hidden Generator Point.

In this protocol, either of the communicating parties i.e Alice or Bob is supposed to have a hidden Generator point. Both the parties choose their respective private keys X and Y in the form of Scalar numbers. After performing scalar multiplication and inverse operations in a series of exchanges as illustrated in the figure 6., both the parties establish a common shared point i.e Ga .

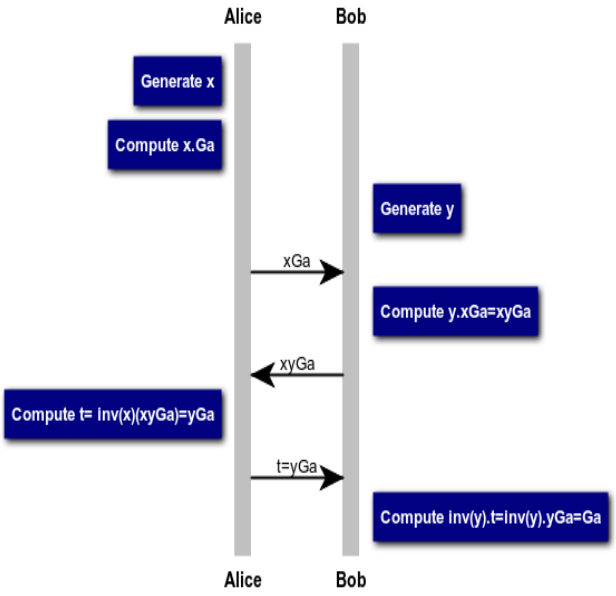

Fig 6 : Shared Key generation using 1-hidden generator point

5.3 Authentication of nodes:

After arriving at a common generator point G and shared key K, the following protocol steps can be adopted for authentication of nodes:

1. Node A calculates hash of its ID : Hash (IdA)
2. Node B calculates hash of its ID : Hash (IdB)
3. Node A calculates G.Hash (IdA), EK (IdA) and sends it to B
4. Node B decripts IdA by performing DK (IdA) and calculates G.Hash (IdA)
5. If G.Hash(IdA) calculated by node B at step 4 is same as G.Hash(IdA) of step 3 then node B authenticates node A .

The protocol can use simple Encryption (EK) and Decryption (DK) symmetric functions. Similar mechanism can be adopted for mutual authentication of nodes.

5.4 Simulation Outputs

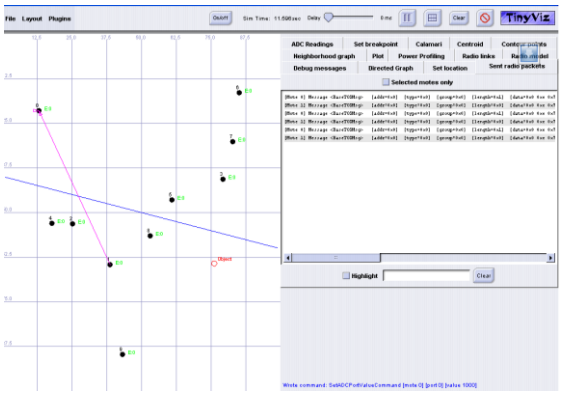

Fig.7: Tinyviz simulation of first protocol with 2-hidden generator point

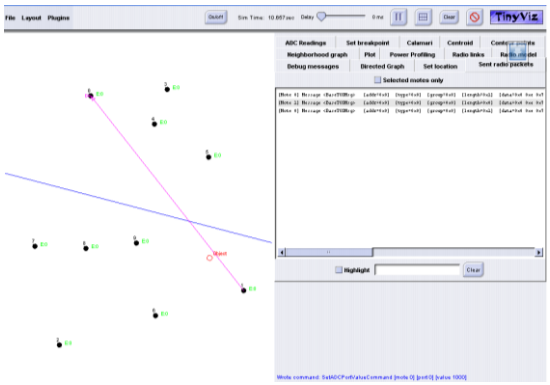

Fig.8: Tinyviz simulation of second protocol with 1- hidden generator point

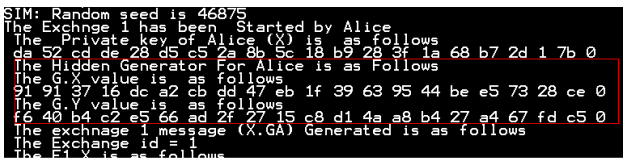

Fig.7(a): Hidden generator point at Alice

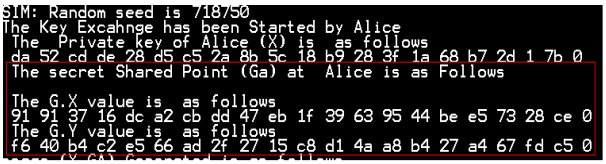

Fig.8(a): Hidden generator point of Alice

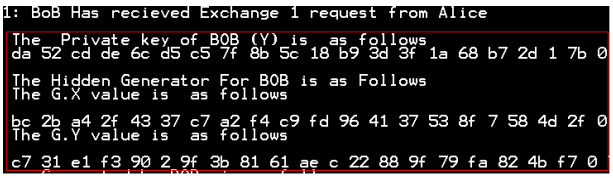

Fig.7(b): Hidden generator point at Bob

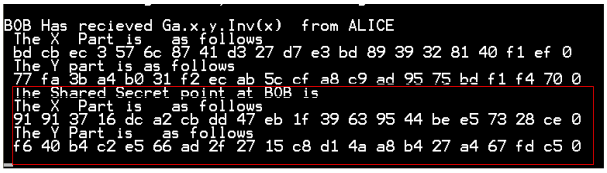

Fig.8(b): Shared hidden generator point with Bob

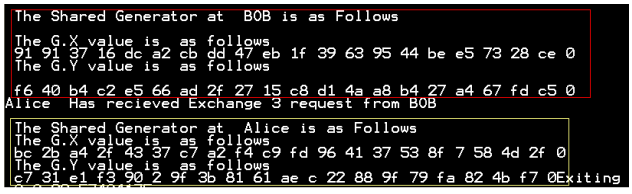

Fig.7(c): Exchange of hidden generator points between Alice & Bob

Both the protocols i.e 1-hidden generator point and 2-hidden generator point were implemented and simulated in Tiny OS[19]. A discrete event simulator TOSSIM[20] was used for simulating the NesC applications developed using TinyECC[21] for the concerned protocols. The applications were also ported on MICAZ hardware. A graphical user interface of TOSSIM, Tinyviz was used for capturing the exchanges between Alice and Bob. The simulation outputs for 2-Hidden Generator Point and 1-Hidden Generator are shown in Fig 7 and Fig 8 respectively.

6 Performance Benchmarking

The performance benchmarking of the key exchange protocol involving hidden generator points would be based on the following parameters:

1. Energy Consumption
2. Memory Consumption

3. Computational Time

In comparison to 2 -hidden Generator Points, the 1-hidden Generator Point algorithm has better performance in terms of Memory Consumption and defense against MIM as indicated in the Table 1.

Table 1: Comparative statement

S.No	Protocol	No of Exchanges	Scalar Multiplication	Point Addition	Inverse Operation	ROM	RAM	Defense against MIM
1	2 -hidden Generator Points	6	8	1	2	16482 Bytes	1890 Bytes	Yes
2	1-hidden Generator Point	3	4	-	2	15482 Bytes	1337 Bytes	Yes
3	ECDH	02	04	-	-	1487 bytes	1208 bytes	No

6.1 Energy Calculations

Energy Calculations would primarily depend on computational time taken for core ECC operations like Point Addition, Scalar Multiplication in addition to the voltage and current requirements. For calculation of energy we use E= V*i*t (joules) where V and i stand for voltage and current drawn respectively, t is the execution time for each operation . MicaZ node using Atmel AT Mega 128 L is powered by 02 AA batteries. Assuming voltage of 3 V for 02 AA batteries, and a maximum load current of 19.7 mA, the energy calculations for each operation are indicated in the Table 2.

For the purpose of capturing computational time of various key ECC operations like Point addition , Scalar Multiplication a basic setup was established using MicaZ, MIB520(programming board). A nesC program was developed for sending the time message to a TinyOS Serial Forwarder. These packets were sent on serial port through a MIB 520 programming board. The packets captured by the serial forwarder were transported to a java application.

Table 2: Energy calculations

Operations	Avg Time Taken (Seconds)	1-hidden Generator Point	Energy Consumption (1-Hidden Generator) (milli Joules)	2-hidden Generator Point	Energy Consumption (2-Hidden Generator) milli Joules	ECDH	Energy Consumption (ECDH) milli Joules
Scalar Multiplication	1.78	4*1.78 = 7.14 secs	422.38	8*1.78 14.29 sec	844.75	4*1.78 = 7.14 secs	422.38
Inverse Operation	0.11	2*0.11 =0.23 secs	14.06	2*0.11 0.22sec	13.49	nil	nil
Point Addition	1.787	NIL	nil	1.78sec	105.61	nil	nil
TOTAL		7.37 secs	436.44	16.29	963.85	7.14secs	422.38

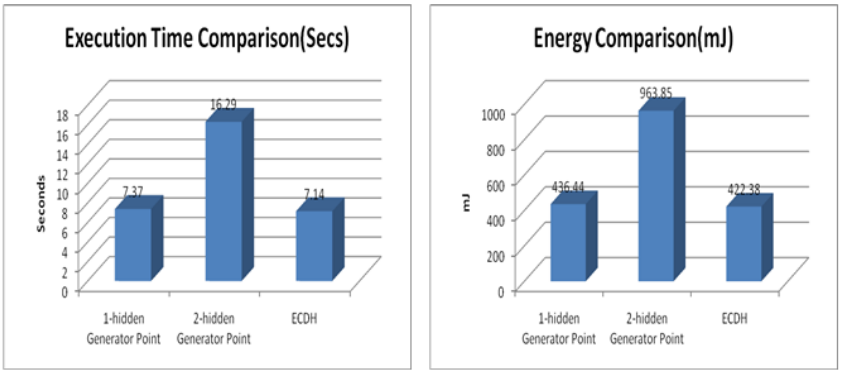

Figure 9: Execution time and energy comparison

7 Conclusion

A Hidden Generator Point in ECC can be useful to thwart Man-In-The-Middle Attack in a resource constraint WSN Network. The conventional ECDH used for generating shared keys does not offer such an advantage. The performance Benchmarking matrix of protocols discussed in the paper based upon hidden generator concepts clearly indicates that resource utilization of 1-hidden generator point protocol is comparable to that of ECDH with an added advantage of offering protection against Man-in-The-Middle attack. The energy consumption which inter alia depends upon computational time of each operation was found to be 436.44mJ in case of 1-hidden generator point as against 422.38mJ of ECDH protocol and 936.85mJ of 2-hidden generator points. The paper illustrates the generation of shared keys along with a simple authentication protocol based upon hidden generator. The concept can be further exploited in developing energy efficient security application for low power devices used in smart cities.

REFERENCES

[1] Adrian Perrig, John Stankovic, David Wagner.," *Security in wireless sensor networks",* Communications of the ACM, vol 47,no. 6, pp 53-57, June 2004.

[2] Adrian Perrig, Robert Szewczyk, J.D. Tygar, Victor Wen and David E.Culler., "*SPINS: Security protocol for sensor networks*", in proceedings of 7th International conference on mobile networking and computing, 2001, vol 8, no.5, pp 189-199, 2001.

[3] Xiaojiang Du, Hsiao-Hwa chen. , "Security in Wireless Sensor Networks", IEEE Wireless Communications, August 2008.

[4] N.Gura, A.Patel, A. Wander, H.Eberele and S. Shantz.," *Comparing Elliptic Curve Cryptography and RSA on 8 bit CPU*".,in 2004 workshop on cryptographic hardware and embedded systems, August 2004.

[5] Zhu, S.,Setia,S., and Jajochia, S., "*LEAP: Energy efficient security mechanism for large-scale distributed sensor networks*", In the proceedings of the conference on computer and communications security ,03,ACM Press, Washigton DC 2003, pp 62-72.

[6] Escheanauer, L., and Gilgor, U.D., "*A Key management scheme for distributed sensor networks*.", in the proceedings of the conference on computer and communications security "02", Washington DC 2002 pp 41-47.

[7] Chan, H,Perrig, A., and Song,D," *Predistribution schemes for sensor networks*", in the proceedings of IEEE security and privacy symposium ,IEEE Computer society press, Loss Alanos 2003, pp 197-213.

[8] R. Watro, D. Kong, S.Cuti, C.Gardiner, C.Lynn and P. Kruus., " *TinyPK: Securing sensor networks with public key technology*." , in the proceedings of 2nd ACM workshop on security of adhoc sensor networks (SASN 04), pp 59-64, New York, ACM press.

[9] Q.Huang, J.Cukier, H.Kobayashi, B.Liu and J.Zhang., " *Fast authenticated key establishment protocols for self-organizing sensor networks*", in the proceedings of the 2nd ACM international conference on WSN and applications , pp 141-150,ACM Press, 2003.

[10] ZHAO Xin, EANG Xia-dong., "*Design and implementation of the Hybrid broadcast authentication protocols in WSN*", published in 2nd international conference on future generation communication and networking, 2008.

[11] Xu Huang, et al. "Fast Scalar multiplication for Elliptic curve cryptography in Sensor Networks with Hidden Generator point", 2010 International conference on Cyber-enabled distributed Computed and knowledge Discovery.

[12] Ravi Kishore et al. "*High Performance Scalar Multiplication for ECC*. In 2013 International Conference on Computed Communication and Informatics (ICCCI-2013, Jan 04-06, 2013 Coimbatore, INDIA)

[13] Ravi Kishore Kodali et al. "Implementation of ECC with Hidden Generator Point in *Wireless Sensor Network*". 978-1-4799-3635-9/14 @ 2014 IEEE.

[14] D. Hankerson et al. "*Guide to Elliptic Curve Cryptography*" Springer, 2004

[15] Bernard Menzes "*Network Security and Cryptography*", Cengage Learning

[16] Ioannis Chatzigiannakis et al. "*Elliptic Curve Based Zero Knowledge Proofs and their Applicability on Resource Constraint Devices*". Ict-2010-258885(SPITFIRE)

[17] Yi Jiang et al. "*Cluster Based Strategies for Public Key Authentication in Wireless Sensor Networks*". Chinese journal of Sensors and Actuators, Volume 20,6,2007.

[18] www.certicom.com

[19] TinyOS. http:// www.tinyos.net

[20] P. Levis, N. Lee, M. Welsh and D. E. Culler. et al *TOSSIM : "Accurate and Stable Simulation of Entire TinyOS Applications"*. SenSys 2003

[21] A. Liu and P. Ning et al. Tiny ECC: "*A Configurable Library for Elliptical Curve Cryptography in Wireless Sensor Networks"* IPSN 2008

