International Journal of Applied Engineering Research ISSN 0973-4562 Volume 11, Number 6 (2016) pp 4280-4287

Authenticated key exchange protocol for Wireless Sensor Networks

Ayaz Hassan Moon
National Institute of Electronics and Information Technology, Srinagar, J & K, India.

Ummer Iqbal
National Institute of Electronics and Information Technology, Srinagar, J & K, India.

G. Mohiuddin Bhat
Department of Electronics and Instrumentation Technology, University of Kashmir, Srinagar, J & K, India.

Abstract
Wireless Sensor Network (WSN) suffer from many constraints including lower processing power, low battery life and small memory. Security turns into the primary concern to manage in such sort of systems as customary Public key cryptography (PKC) algorithms are computionaly expensive and hence not feasible for WSN. As key exchange is significant in every cryptographic primitive, requirement for a proficient and secure key exchange gets to be imperative. In this paper, an authenticated key exchange algorithm based on Elliptic Curve Cryptography (ECC) has been presented for wireless sensor networks. The algorithm builds up a mutual key between two communicating nodes in an authenticated manner. The protocol is resistant against Man in the Middle Attack. The Shared Key can further be utilized for achieving any cryptographic service like authentication, Confidentiality etc. The developed protocol has been simulated in TinyOS using Tossim simulator. The protocol has also been run on MicaZ motes. The performance benchmarking of the proposed protocol against ECDH has also been carried out.

Keywords: WSN, TinyOS, TinyECC, Tossim

Introduction

Wireless sensor systems involves battery-operated sensor nodes having restricted assets in terms of power, processing and energy. These nodes need to work in a complex and harsh environment. The unattended operation of WSN makes them vulnerable against various types of attacks. Typical security protocols can't be specifically ported to WSN for their asset limitations. Key establishment plays a pivot role in ensuring core security services like authentication, integrity etc[1]. An uncomplicated yet efficient method to share secret keys in a WSN is based on the idea of key pre-distribution. The key pre-distribution includes stacking every node with a set of keys randomly chosen from a large key pool such that two nodes will share one key. While this basic scheme is easy to implement and entails only little overhead since no costly key agreement must be carried out, it has some disadvantages in terms of scalability and resilience to node capture.

A totally distinctive methodology for key distribution in WSNs is to utilize a trusted third party (e. g. the base station) that acts as a key dissemination center (KDC) and produces, upon solicitation,, a unique secret key for two nodes wishing to communicate securely with each other. Each node of the network shares a long-term secret key with the KDC, which enables the nodes to verify that messages from the KDC are authentic. The long-term key that each sensor node shares with the KDC is pre-deployed, but, contrary to approaches with a single network-wide key, each node has a unique key. Therefore, compromised nodes do not jeopardize the security of the rest of the WSN. The trusted third party schemes using BS have several limitations:

- Base station becomes a single point of failure
- It results in heavy communication over heads in the network as each sensor node has to communicate with the BS to get pairwise keys. This method will result in power exhaustion as transmitting 1-bit consumes energy nearly equal to executing 800-1000 instructions [2] Karlof et al.
- Nodes in the neighborhood of BS shall likely exhaust their energy reserves at a high pace in comparison to other nodes located distantly.

Zhu et al proposed Localized Encryption and Authentication Protocol (LEAP) [3], a key management Scheme which underpins the foundation of four sorts of keys for every sensor node. It incorporates an individual key imparted to 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. The disadvantages are in terms high cost of memory to store the four unique keys for every node, when the number of nodes are few.

Key establishment in WSNs can also be performed with protocols that use public-key cryptography to generate a secret key shared between two nodes. In traditional networks, Public key cryptography (PKC) has been judiciously used with Symmetric Key Cryptography for achieving major security services. PKC based key management schemes provide resilience, scalability and flexibility to a network. There down side being that they are computationally expensive, slow with high energy demands. The general trend is to use Symmetric Keys for encryption of data and adopt PKC for key distribution. This advantage has not been leveraged in case of WSN due to its resource constraint nature. PKC is yet to find its foothold while implementing various security services in WSN. But with the successful implementation of PKC in WSN using Elliptic Curve Cryptography, PKC operations have now become a viable option if used sparingly. Achieving

authenticated broadcast is one of the gains of adopting PKC based key management.

Due to ECC advantages, PKC is achievable even on resource constraint wireless sensor nodes as was demonstrated by N. Gura et al (2004)[4] and W. Du et al (2005) [5]. In WSN, ECC is a natural cries among various PKC options due to its efficient execution, fast computation, small key size signatures in comparison to other PKC schemes like RSA. An ECC protocol needs 160 bit keys to provide security equivalent to 1024-bit RSA. Smaller key size brings in other significant advantages in terms of smaller memory (both ROM as well as RAM), faster execution and efficient storage. The implementations by A. Liu et al (2008)[6] on TinyECC, D. J. Malan et al (2008) [7][8] in implementing PKI for Sensor Networks did not provide the expected efficient solutions for WSN. However with ECC optimization techniques such as Barret Reductions, Sliding Windows, Shamir's Trick resulted in overcoming various Sensor network limitations in terms of memory, size. Such optimizations at times resulted in 10 fold performance enhancement. It takes 1. 61 seconds to verify an ECC based signature on ATmega 128 while consuming 45. 09 mj energy. On Imote2 the energy is reduced to 3. 51 mj for the same operation. (2005)[9]. ECC provides an asymmetric key exchange mechanism known as Elliptical Curve Diffe Hellman. The main advantages of using ECDH key exchange in WSNs are perfect resilience to node capture, excellent scalability, and low memory as well as communication overhead. However, the big drawback of ECDH [10] is that it suffers from Man –in-the-Middle Attack [11] which occurs due to lack of authentication between two parties. The drawbacks of the existing techniques are shown in the Table 1.

Table 1: Limitations of Existing Schemes

S. No	Key Establishment Technique	Drawbacks
1	Key Pre-distribution	Node Capture, Scalability
2	Key Distribution Centre	High Communication Cost
3	LEAP	High Memory Consumption
4	ECDH	Man-in-the-Middle Attack

The paper primarily presents a light weight Authenticated key exchange protocol using ECC for establishing a shared key to overcome the issue of Man –in –the-Middle Attack. Rest of the paper is organized as :Section 2 provides feasibility of using ECC in WSN Section 3 provides specifics about Proposed Protocol,. Section 4 presents the Implementation and analysis and Section 5 discusses the performance of the developed protocol.

Elliptical Curve Cryptography & WSN

In general elliptic curves (ECC) combine number theory and algebraic geometry. These curves can be defined over any field of numbers (i. e., real, integer, complex and evenFp). An elliptic curve consists of the set of numbers (x, y), also known as points on that curve that satisfies the equation:

$$y^2 = x^3 + ax + b$$

The set of all of the solutions to the equation forms the elliptic curve. Changing a and b directly changes the shape of the curve. We generally see elliptic curves used over finite fields in cryptography applications where the points (x, y) form an additive group. Elliptic curves can be used to define a"hard" to solve problem: Given two points, P and Q, on an elliptic curve, find the integer k, if it exists, such that P = kQ..

The well-known public crypto system RSA is based upon modular exponentiation. Its security is derived from the difficulty of factorizing large integers. The solution of integer factorization lies in sub-exponential algorithm [10].

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

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

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[11]. 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 as shown in Table 2. Smaller key size often brings the advantage of faster computation efficiency and saving of bandwidth, memory and energy. Therefore ECC is better suited for resource constrained devices like WSN.

Proposed Authenticated Key Exchange Mechanism

Typically Asymmetric cryptography is used for key exchange between 2 parties. The exchanged key can then be used for achieving various cryptographic primitives like Authentication, MAC, Integrity using symmetric cryptography. ECDH, which is a asymmetric technique for exchanging key is more feasible for low constraint devices used in Wireless Sensor Networks. However ECDH suffers from Man –in –the middle attack primarily due to lack of authentication between two parties. This section proposes an authenticated key exchange protocol based on ECC which exchanges a key between two parties by in-cooperating authentication.

ECDH

Using ECDH two nodes having the same elliptic curve domain parameters can build up a common key over a unsecured channel without exchanging their secrets. In ECC usage, the hardness is gotten from ECDLP [10][11]. The sequence for establishing the shared key is shown in figure 1. Suppose

Alice needs to set up a mutual key with Bob, however the channel accessible to them might be eavesdropped by a third party. G is the generator point on the choosen ECC curve. Initially Alice computes the Scalar Multiplication x. G where x is the chosen secret of Alice. x. G is transmitted to BOB. As this is an ECDLP problem the Secret x cannot be extracted from x. G. Simillary BOB computes the y. G and sends it to Alice. The key established is x. y. G as both parties multiply the received expression with their Secrets. ECDH is the efficient way of exchanging keys in WSN [12][13].

Man in the Middle Attack in ECDH

A sort of attack, where in a attacker node sets in himself between two sensor node to capture their dynamic communication. It is a sort of eves dropping which can prompt handing-off of wrong messages to both communicating sensor nodes thus leading to breach of confidentiality, authentication and data integrity and is therefore perceived as a serious threat to any network. MIMA attacks lead to session hijacking and is an attack on mutual authentication. In MIM as shown in figure 2 eve chooses two random numbers p and q and manipulates the sequences in such a way that a separate pair of key is established between Alice and Eve where in Alice believes it has established a key with BOB and Similarly between BOB and EVE where in BOB thinks that it has established a key with Alice [10][11][14].

Proposed key exchange Protocol

The major problem in ECDH is that no authentication happens prior to exchange. Based on this need, an improved protocol has been suggested in the figure 3. The algorithm developed performs a sequence of steps to establish a shared key between two parties while in co-operating a mechanism to authenticate. Before initiating the protocol an asymmetric signature key pair for each party must be generated. The asymmetric pair based on ECC can be pre deployed in each node before deployment. The notations used in the protocol are shown in the table 3.

Table 3: Symbol Table

Symbol	Description
X	Random Number Chosen By Alice
Y	Random Number Chosen By BOB
G	Generator Point of Elliptical Curve
X. G	Secret at Alice
Y. G	Secret at BOB
Ka	Private Key of Alice
Kb	Private Key of BOB
PUa=Ka. G	Public Key of Alice
Pub=Kb. G	Public Key of BOB

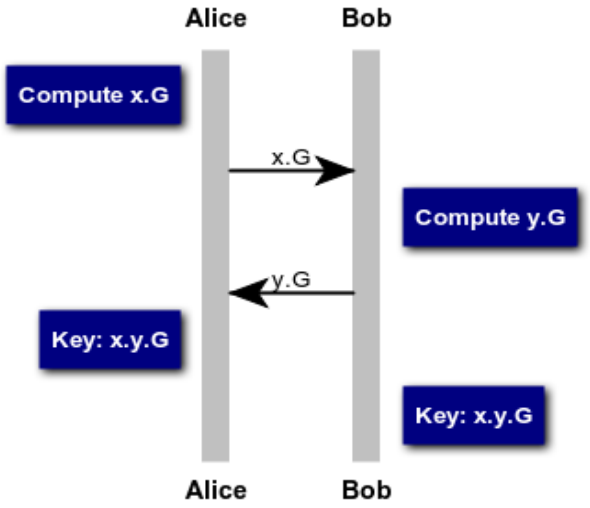

Figure 1: Elliptical Curve Diffe-Hellman(ECDH)

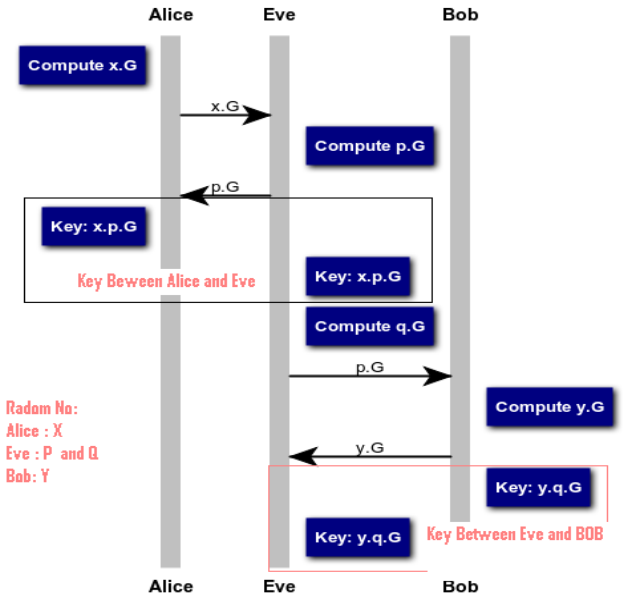

Figure 2: Man-in-The-Middle Attack in ECDH

In this protocol, Alice and BOB choose the Radom numbers x and y respectively. The secret chosen by Alice is X. G and by BOB is Y. G. The purpose of the key exchange algorithm is that Alice and BOB should be able exchange there secrets with each other in an authenticated manner.. BOB receives (Ka+x). Pub. BOB multiplies (Ka+x). Pub with inv(Kb) i. e inverse of the private key of the BOB. Multipliying (Ka+x). Pub with inv(Kb) results in (Ka+x). G i. e (Ka+x). Kb. G. inv(Kb). The intermediate value(Ka+x). G is added with inverse of Pua which yields x. G. The sequence of exchanging Y. G from BOB to Alice is shown in the figure 3. After the secrets have been exchanged the keys established at Alice and BOB is X. G + Y. G. ie the point addition of secrets

The defense against MIMA[4] can be understood by considering the following illustration. Let us say eve intercepts the first exchange i. e (Ka+x). Pub. In order to establish a fraudulent key with Alice, eve should be able to derive x. G from it. Deriving x. G is not possible for eve as she doesn't know the private key kb which is required for computing inv(Kb). Using PUa in deriving X. G from (Ka+x). Pub by BOB provides data source authentication of Alice to BOB.

Implementation

Simulation

The protocols were implemented in TinyOS [15] operating system using NesC Language and simulated on TOSSIM [16]. TinyOS is a open-source lightweight operating system specifically designed for low-power wireless sensors. NesC is a dialect with features to reduce RAM and code size, enable significant optimizations, and help prevent low-level bugs like race conditions. The NesC programs developed were enabled with a highly optimized ECC implementation, TinyECC. TinyECC [6] is a code packet provided by North Carolina State University develops team. It provides a base arithmetic operation of ECC on TinyOS. It provides all ECC operations on domain, including the point add, double and scalar multiplication. Fp. The Main modules of TinyECC are shown in the table 4.

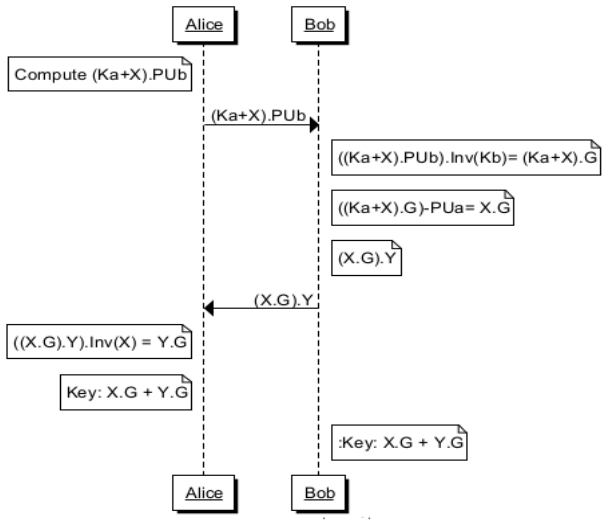

Figure 3: Proposed Protocol

Table 4: Major Components of TinyECC

S. NO	Module	Description
1	Natural Number Module	Provides the realization of Large operations in Sensor nodes
2	ECC Module	Provides many basic operations on Elliptical Curve which includes initialization of an elliptical curve, point adding, point doubling, scalar multiplication and some more operations based on sliding window
3	ECDSA Module	Implements signature protocol based on ECC
4	ECDH Module	Implements Elliptical Curve Diffe-Hellman Protocol
5	ECIES Module	Implements Elliptical Curve Integrated Encryption Scheme

The component graph of TinyECC and the developed protocol generated using nesdoc is shown in the figure 4 and 5. TinyECC provides 3 elliptical curve standards secp128 secp160, secp192. These curve standards are chosen by passing a global argument in the make system of Security framework being developed. The elliptical curve domain parameters (p, a, b, G, n, h) are initialized during node init state. The values initialized depend upon the formal curve argument passed through the make system. The program uses TimerC component of TinyOS library for synchronising the operation within the programme. GenericComm component is used for sending and transmitting radio packets using AM Stack.

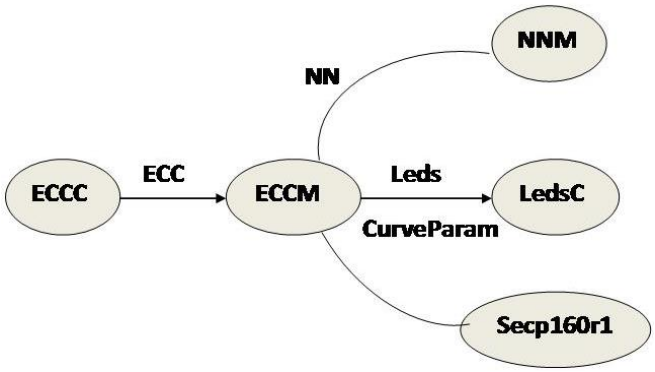

Figure 4: Wiring Diagram of TinyECC

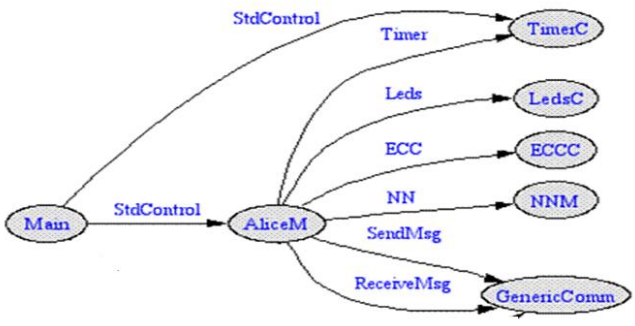

Figure 5: Component Graph of Proposed Protocol

The simulation has been carried out in TOSSIM Simulator. Using DBG flags, the output of various calculation and transmissions in the protocols has been captured. TOSSIM provides configuration of debugging output at run-time.. Much of the TinyOS source contains debugging statements. Each debugging statement is accompanied by one or more modal flags. When the simulator starts, it reads in the DBG environment variable to determine which modes should be enabled. In our simulations DBG variable has been set to

USR1 (user defined) LEDS, AM(Messages). The TinyVIZ snapshot is shown in the figure 6 The simulation output is shown in the Table 5.

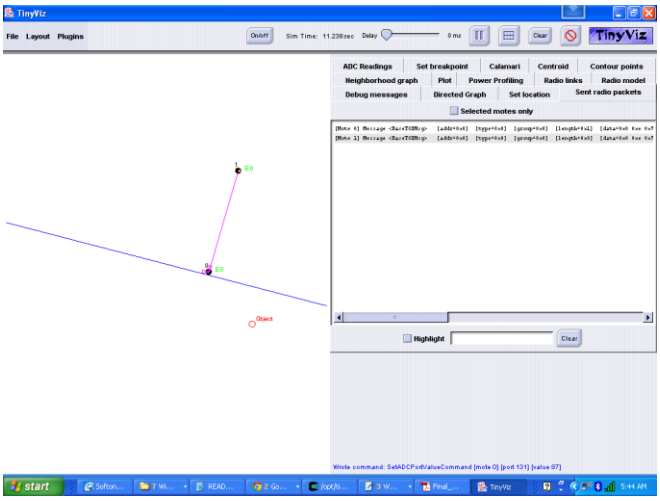

Figure 6: TinyVIZ Snapshot of proposed protocol

Hardware setup

The NesC program was fused on MicaZ Motes using a MIB 520 programmer. MicaZ[14] is a sensor node with IEEE 802. 15. 4 compliant RF transceiver. The typical characteristics of a MicaZ mote are 8-Bit micro-controller, 4 KB of RAM, 128 KB of ROM and bandwidth of 250kbpsThe MIB520 [17] provides USB connectivity to the MICA family of Motes for communication and in-system programming. The Fusing Snapshot of the proposed protocol is shown in the figure 7.

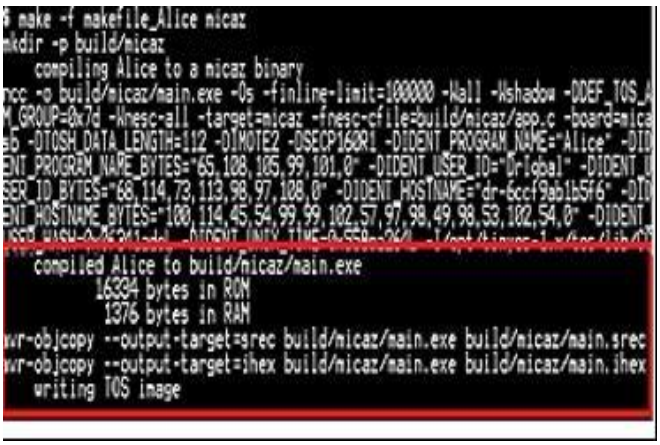

Figure 7: TinyOS Snapshot of proposed protocol being fused on Mica

Table 5: Packet captured using Tossim

Exchange	Source	Destination	Operation	DBG Message
01	Alice	BOB	(Ka+X). Pub	The ((Ka+X).PUb).X is as follows 24 23 67 83 55 d4 0 9f 31 d8 4b 59 49 1e ca d5 7 f6 34 af 0 The ((Ka+X).PUb).Y is as follows 2f 95 24 7b e2 8e f5 42 c8 d b2 d9 c5 69 e2 a2 62 ea 9e 85 0
02	BOB	Alice	(X. G). Y	The (X.G).Y).X is as follows b9 71 7d f3 5c 63 32 f4 7f 88 3a e6 13 d8 34 d3 f7 ac 7f b7 0 The (X.G).Y).Y is as follows 28 95 c4 d6 75 d 39 8f cd dc 18 3e 29 78 6c ba b2 f6 90 50 0
Key Generated at Alice			X. G + Y. G	The Key X.G + Y.G Generated at Alice is The X Part is as follows 91 91 37 16 dc a2 cb dd 47 eb 1f 39 63 95 44 be e5 73 28 ce 0 The Y Part is as follows f6 40 b4 c2 e5 66 ad 2f 27 15 c8 d1 4a a8 b4 27 a4 67 fd c5 0
Key Generated at BOB			X. G + Y. G	The Key X.G + Y.G Generated at BOB is The X Part is as follows 91 91 37 16 dc a2 cb dd 47 eb 1f 39 63 95 44 be e5 73 28 ce 0 The Y Part is as follows f6 40 b4 c2 e5 66 ad 2f 27 15 c8 d1 4a a8 b4 27 a4 67 fd c5 0

Performance Analysis

The performance benchmarking of the proposed key exchange protocol with ECDH is based on the following parameters[4][17]:

- ✓ Memory Consumption
- ✓ Computational Time

While Fusing the Motes with Actual Code RAM and ROM consumption as calculated by TinyOS were captured. These requirements were recorded for various SECP curve which include Secp128r1, Secp160r1, Secp190r1. The RAM requirements of the proposed protocol and ECDH are shown in the figure 8

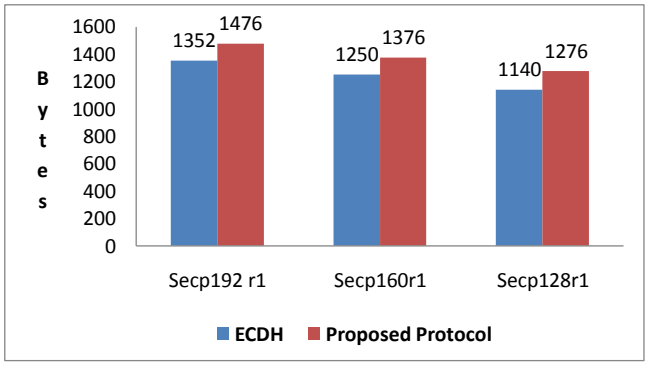

Figure 8: RAM Requirement of ECDH and Proposed Protocol

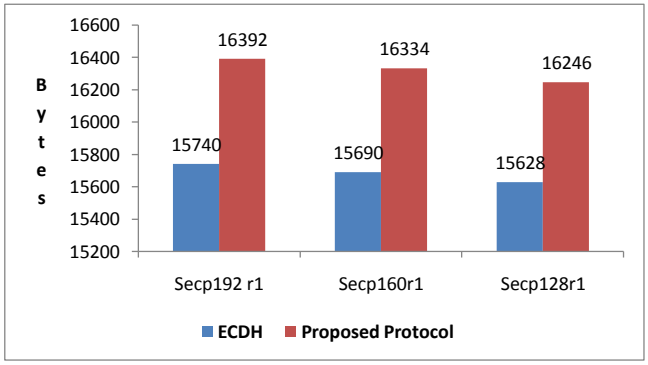

Figure 9: ROM Requirement of ECDH and Proposed Protocol

The ROM Requirements are shown in the figure 9. 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 through MIB520. The TinyECC Switches enabled in the program are shown in the Table 6. The time in the time message comprised of the time taken in the execution of the specific operation. This was achieved by starting and stopping the timer interface and recording the difference of the two. The difference was sent as a payload to the serial forwarder as shown in the figure 10.

Table 6: Optimization Switches in TinyECC

S. No	Name	Description
1	BarrettReduction (BARRETT)	Barrett reduction is an alternative method for modular reduction
2	Projective Coordinate Systems (PROJECTIVE)	Jacobian representation in TinyECC to speed up point addition, point doubling and scalar point multiplication.
3	Sliding Window Method (SLIDING_WIN)	sliding window method to speed up scalar point multiplication
4	Hybrid Multiplication (HYBRID_MULT, HYBRID_SQR)	Optimized Multilplication and Squaring for MicaZ

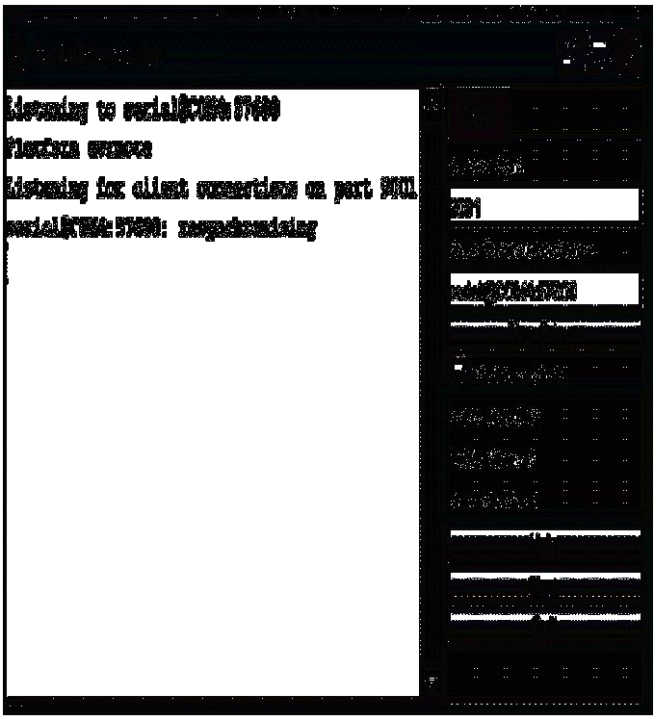

Figure 10: Serial Forwarder In TinyOS

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. The Serial Forwarder is an inbuilt functionality in TinyOS which is used to read packet data from MIB 520 and forward it over an Internet connection, so that other programs can be written to communicate with the sensor network over the Internet. A java program written connected with the serial forwarder to read these packets. The no of crtical operations and Time taken by key operations is shown in the table 7 and table 8.

Table 7: Time Taken by Key Operations

Operation	Time Taken (Seconds)		
	Secp192r1	Secp160r1	Secp128r1
Scalar Multiplication	6. 07	3. 90	2. 23
Point Addition	0. 21	0. 14	0. 098
Inverse Operation	0. 16	0. 12	0. 069

Table 8: Number of Crtical Opertions

Protocol	Point Addition	Point Multiplication	Inverse Operation
ECDH	NIL	4	NIL
Proposed Protocol	3	4	2

Comparison graphs for total time taken by ECDH and Proposed protocol is shown in the figure 11

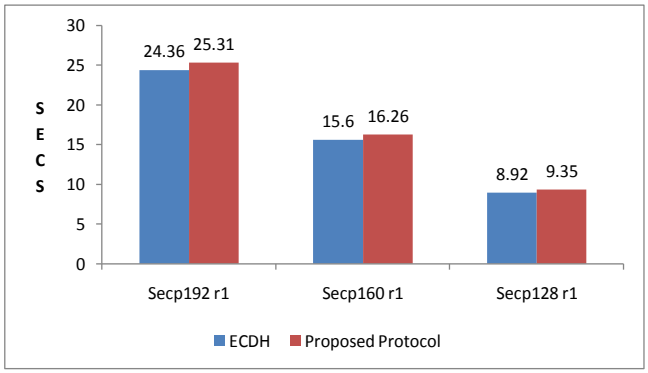

Figure 11: Comparison of Time taken by ECDH and proposed protocol

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. With a voltage of 3 V for 02 AA batteries, and a maximum load current of 19. 7 mA, the energy comparison is shown in the figure 12.

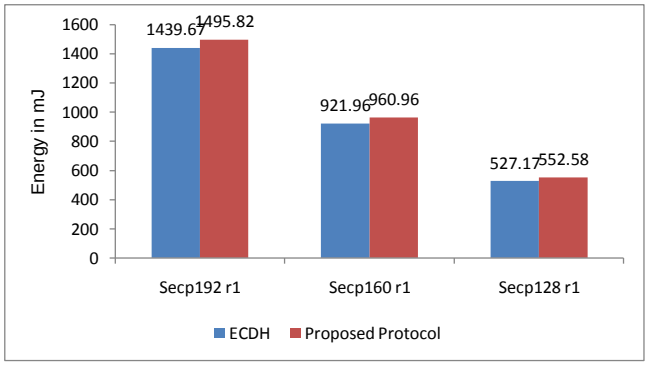

Figure 12: Comparison of Energy consumed by ECDH and proposed protocol

Conclusion

In this paper, a key exchange protocol based on ECC has been proposed and implemented in TinyOS platform. Existing key exchange mechanism in WSN were discussed. These techniques mainly suffer from the issues like node capture, scalablity, communication overhead and Man-in-the Middle-Attack. The existing techniques discussed included pre-deployment of keys, key distribution centre (KDC), LEAP and ECDH. It was pointed out that ECDH being a asymmetric technique is more prudent than others. Work has been done to implement ECDH using ECC for WSN as ECC offers much better performance than RSA. However the ECDH key exchange suffers from Man-in-the-Middle-Attack. The proposed key exchange protocol based on 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 of protocol discussed in the paper clearly indicates that resource utilization of proposed protocol is comparable to that of ECDH with an added advantage of offering protection against Man-in-The-Middle attack. With a small increase in computational time and memory the protocol does provide a strong protection against MIMA and thus can be utilized for practical purposes. The concept can be further exploited in developing energy efficient security application for low power devices used in smart cities. As authentication is the key cryptographic service in WSN, the use of the Proposed protocol must be evaluated to solve the key exchange problem of exiting authentication protocols in WSN like TinySec, SNEP etc as most of them exchange key in a symmetric manner.

References

[1] Xiao, Y., Ravi, V. K. and Sun, B., 2007 " A Survey of Key Management Schemes in Wireless Sensor Networks, " Elsevier Journal of Computer Communications., Vol 30, pp 2314-2341.

[2] Karlof, C., Sastry, N. and Wagner, D. 2004" TinySec: Link Layer Security Architecture for Wireless Sensor Networks, " Sensys., Baltimore, MD.

[3] Setia, S., and Jajochia, S., 2003 "LEAP: Energy efficient security mechanism for large-scale distributed sensor networks, "ACM Press, Washigton DC, pp 62-72. 76.

[4] Gura, N., Patel, A., et. al 2004 " Comparing Elliptic Curve Cryptography and RSA on 8 bit CPU, " Workshop on cryptographic hardware and embedded systems.

[5] Du, W., Wang, R. and Ning, P. 2005 " An Efficient Scheme for Authenticating Public Keys In Sensor Networks, " 6th. ACM, MobiHoc-05, pp 58-67

[6] Liu, A. and Ning, P. et al., 2008 "Tiny ECC: A Configurable Library for Elliptical Curve Cryptography in Wireless Sensor Networks", IPSN – vol 00, pp. 245-256, 2008

[7] Mallan, D. J., Welish, M. and Smith, D. M, 2008, " Implementing Public Key Infrastructure for Sensor Networks, " Transactions on Sensor Networks, vol. 4

[8] Mallan, D. J., Welish, M. and Smith, D. M, 2004 " A Public Key Infrastructure for Key Distribution in TinyOS based on Elliptic Curve Cryptography, " First Annual IEEE Communications Society Conference on Sensor and Adhoc Communications and Networks,. pp. 71-80.

[9] Wander, A. S, Gura, N., Eberle, H., Gupta, V. and Shantz, S. C. " Energy analysis of Public Key cryptography on Small Wireless Devices, " 3rd. IEEE International Conference on Pervasive Computing and Communications, pp 324-328.

[10] Menzes, B., "Network Security and Cryptography", Cengage Learning
[11] Hankerson, D. et al. "Guide to Elliptic Curve Cryptography" Springer.
[12] Gura, N., Patel, A., Wander, A. S, Eberle, H. and Chang Shantz, S., 2004 "Comparing elliptic curve cryptography and RSA on 8-bit CPUs". Cryptographic Hardware and Embedded Systems, vol. 3156, pp. 119–132. Springer [13] Energy-Efficient Implementation of ECDH Key Exchange for Wireless Sensor Networks:Crtien etal at IFIP International Federation for Information Processing 2009
[14] Huang, X. Shah, P. G and Sharma, D. 2010 "Protecting from attacking the man-in-middle in wireless sensor networks with elliptic curve cryptography key exchange, " 4th IEEE International Conference on Network and System Security, pp. 588–593.
[15] Levis, P., and Gay. D., 2009 "TinyOS Programming. Cambridge University Press"
[16] Levis, P., Lee, N., Welsh, M. and Culler, D. E. et al "TOSSIM : Accurate and Stable Simulation of Entire TinyOS Applications". SenSys
[17] Memsic Mote Manual