Volume 1, No. 3, October 2010

Journal of Global Research in Computer Science

ISSN-2229-371X

RESEARCH PAPER

Available Online at www.jgrcs.info

A NEW ALGORITHM FOR END TO END SECURITY OF DATA IN A SECURE SELF ORGANIZED WIRELESS SENSOR NETWORK

Hemanta Kumar Kalita[*1] and Avijit Kar[2]

*[1]Dept of Computer Sc & Engineering, Jadavpur University, Kolkata, India
hemanta91@yahoo.co.in1
[2]Department of Computer Sc & Engineering, Jadavpur University, Kolkata, India
dr.avijit.kar@gmail.com2

Abstract: We consider a self-organized wireless sensor network where sensor nodes are multi-hop away from base station. This means data communication happens through several intermediate nodes before it reaches the base station from source. End to end security of data ensures data sent from the source to destination through several intermediate nodes preserve confidentiality, integrity, authentication and non-repudiation of data. Due to resource-constrained sensor nodes it was considered that implementing confidentiality, integrity, authentication and non repudiation of data using public key infrastructure based scheme would be a challenge in Wireless Sensor Network. Consequently we have seen a lot of secure data communication schemes based on symmetric key in the literature. Although symmetric key based schemes are easy to implement, yet it has a single point of failure, i.e. the key. If a node's key falls in the hand of attacker, then the whole network is compromised. In this paper we propose a secure data communication scheme for wireless sensor network. Our scheme takes advantage of both asymmetric and symmetric key based schemes and preserves confidentiality, integrity, authentication and non-repudiation of data.

Keywords: wireless sensor network; security; data communication; public key infrastructure; encryption; authentication

INTRODUCTION

Recent works has showed it in security that strong cryptography can be equated with strong security or even usable security is a myth. While strong cryptography may be necessary, it definitely is not sufficient to realize a system with the required security properties. End-to-End security ensures confidentiality of data from source to destination, while any intermediary node knows from where the data is coming. This essentially implies layer-wise encryption and authentication of data packets. Data from higher layer, such as transport layer will be encrypted and network layer PDU will only be authenticated. This makes any intermediary node to open up any packet and know from where it is coming. However, it can't see the data due to encryption. In this paper we propose our scheme to provide End-to-end security of data in wireless sensor network. Our scheme is a certificate less PKI based scheme suitable for resource constrained wireless sensor nodes to implement.

In [1] and [2] we discuss how key distribution and self-organization happens for our scheme of data communication. Note that before data communication happens every node is equipped with public key of the base station and its nearest neighbor. Also, every node has public keys of those sensors nodes for which it falls in their key path. Our scheme is very particular about public key (of base station or of a neighbor) making it public to other nodes. A new node is given public key of its neighbor and the base station after passing through two level of authentication called neighbor authentication and base station authentication. Also, with proper authentication only public key of a new node is accepted. This is one time activity; once this is done third party or Certification Authority does not verify public key.

Remainder of this chapter is divided into four sections. In section 2 we discuss background or related work. In section 3 we discuss our scheme to provide end-to-end security of data. In section 4 we compare and analyze our scheme and finally in section 5 conclude the paper.

BACKGROUND

In 1985, [3] proposed an Identity based Cryptosystem and Signature Schemes. In this scheme any pair of users can communicate securely and verify each other's signatures without exchanging private or public keys, without keeping key directories and without using the services of a third parties. The scheme assumes the existence of trusted key generation center, whose sole purpose is to give each other a personalized smart card when he first joins the network. The information embedded in this card enables the user to sign and encrypt the messages he sends and to decrypt and verify the messages he receives in a totally independent way, regardless of the identity of the other party. Previously issued cards do not have to be updated when new users joins the network.

[4] States the implementation of a PKI requires an analysis of business objectives and the trust relationships that exist in their environment. The awareness of these trust relationships leads to the establishment of an overall trust model that the PKI enforces.

[5] Describes a security protocol for sensor network called SPINS. SPINS consists of two building blocks: SNEP and µTESLA. SNEP (Sensor Network Encryption Protocol) is for pair-wise communication and provide confidentiality, integrity, authentication and freshness of data. If A wants to send data to B, then SNEP works as follows:

Let us assume M is the message to send.

1) A encrypts M using encryption key, K_{encr} and counter, CTR. Then

$$e: K_{encr} \times CTR \times M \rightarrow C$$

Where C is the cipher text and e is the encryption function.

2) CTR and C are concatenated and encrypted using K_{mac} to generate the MAC. Thus,

$$h: CTR \times C \rightarrow H$$

$$e: K_{mac} \times H \rightarrow \Delta$$

H is the fixed size 'message digest' generated using h. Δ is the MAC generated.

3) Finally, A sends C and Δ to B.

$$A \rightarrow B : C + \Delta$$

Both the keys used in SNEP are generated from the master key. Authors claim that SNEP provides semantic security, authentication, replay protection, weak freshness and low communication overhead.

As mentioned, plain SNEP provides weak freshness of data. This is because, the SNEP enforces a sending order on the messages from A to B; but no absolute assurance to node A that a message was created by B in response to an event in node A. Node A achieves strong data freshness for a response from node B through a nonce N_A. Node A generates N_A randomly and sends it along with a request message R_A to node B. The simplest way to achieve strong freshness is for B to return the nonce with the response message R_B in an authenticated protocol. However, instead of returning the nonce to the sender, the process can be optimized by using the nonce implicitly in the MAC computation. The entire SNEP protocol providing strong freshness for B's response is as follows:

$$A \rightarrow B : N_A + R_A$$
$$e: K_{encr} \times CTR \times R_B \rightarrow C$$
$$h: N_A \times CTR \times C \rightarrow H$$
$$e: K_{mac} \times H \rightarrow \Delta$$
$$B \rightarrow A: C + \Delta$$

[5] Argues that pure TESLA (Timed, Efficient, Streaming, and Loss-tolerant Authentication Protocol) is not feasible in wireless sensor network for several reasons. For example, TESLA authenticates the initial packet with a digital signature, which is too expensive to compute on the sensor nodes. TESLA has an overhead of 24 bytes per packet, which is at the higher end. Finally, the one-way key chain does not fit into the memory of the sensor node. µTESLA is designed to overcome these inadequacies of TESLA in sensor networks. For example, for authentication of initial packet, µTESLA uses symmetric mechanism. Key disclosure is done once per epoch. And µTESLA restricts the number of authenticated users.

[6] Defines CCM Mode, a symmetric key block cipher algorithm. CCM may be used to provide assurance of the confidentiality and the authenticity of computer data by combining the techniques of the Counter (CTR) mode and the Cipher Block Chaining-Message Authentication Code (CBCMAC) algorithm.

[7] Quantifies the energy cost of authentication and key exchange based on public-key cryptography on an 8-bit microcontroller platform. Authors present a comparison of two public-key algorithms, RSA and Elliptic Curve Cryptography (ECC), and consider mutual authentication and key exchange between two not trusted parties such as two nodes in a wireless sensor network. Their measurements on an Atmel ATmega128L low-power microcontroller indicate that public key cryptography is very viable on 8-bit energy constrained platforms even if implemented in software. Authors found ECC to have a significant advantage over RSA as it reduces computation time and also the amount of data transmitted and stored.

[8] Describes secure solutions for collecting and processing data in Wireless Sensor Networks (WSNs), introduces a toolbox concept to support such a framework and gives an overview on security and reliability challenges for WSNs.

[9] Announces that with hardware support and software optimization, public key cryptography (PKC) is feasible on micro sensors. A number of experiments proved that the elliptic curve cryptography (ECC) is more suitable for resource constraint motes compared with RSA, but even ECC based protocols still cost too much energy. In this paper, authors propose C4W, an identity-based public key infrastructure specially designed for wireless sensor networks (WSNs), in which all nodes can generate other's ECC public keys directly from their identities. Without certificates, no energy will be consumed for certificates communication and verification, which makes C4W especially energy efficient. C4W uses a protocol without certificates to realize mutual authentication and key agreement. Compared with a simplified SSL (SSSL) protocol using an abbreviated certificate, C4W consumes lower than 35% energy, and the communication consumption of C4W is only 28.5% of that consumed by SSSL. Furthermore, the energy analysis of C4W illuminates that the expensive public key computational cost is almost neglect able compared with the heavy communication consumption in a large-scale WSNs, which gives the asymmetric key management in WSNs a bright future.

[10] Analyzes a number of the security architectures employed, and proposed, to date, such that the various characteristics of each protocol are easily identifiable to potential network designers, allowing a more informed decision to be made when implementing a security protocol for their intended application. Authentication is the primary focus, as the most malicious attacks on a network are the work of imposters, such as DOS attacks, packet insertion etc. Authentication can be defined as a security mechanism, whereby, the identity of a node in the network can be identified as a valid node of the network. Subsequently, data authenticity can be achieved; once the integrity of the message sender/receiver has been established.

[11] Analyzes the ZigBee 2006 standard, identify security issues and offer recommendations. Issues with authentication, initialization vectors, group keys, MAC layer acknowledgements, cipher modes, security guarantees, key protection, node compromise, revocation and denial of service attacks are examined. Suggestions for ZigBee compatible improvements when deploying the framework are offered. Some changes to ZigBee and IEEE 802.15.4 are also discussed. ZigBee is suitable as a foundation for a secure network, but care has to be taken in the usage of the numerous settings. Many settings provide tradeoffs and the security impacts are not always obvious. The main vulnerabilities are in the area of denial of service attacks while the confidentiality, integrity and authenticity are better protected. ZigBee networks are vulnerable to insider attacks, facilitated by the physical exposure of the hardware. Sensor node capture is a large threat to the networks.

The IEEE 802.15.4 standard defines 8 different security suites. The first of these is the Null suite and provides no security. The next is encryption only (AES–CTR), followed by authentication only (AES–CBC–MAC), and finally encryption and authentication (AES–CCM). In Table I we mention the security suites.

Table 1: IEEE 802.15.4 Security Suites

Null	No Security
AES-CTR	Encryption Only, CTR Mode
AES-CBC-MAC-128 AES-CBC-MAC-64 AES-CBC-MAC-32	128 bit MAC 64 bit MAC 32 bit MAC
AES-CCM-128 AES-CCM-64 AES-CCM-32	Encryption & 128 bit MAC Encryption & 64 bit MAC Encryption & 32 bit MAC

CCM Mode: CCM mode (Counter with CBC-MAC) is a mode of operation for cryptographic block ciphers. It is an authenticated encryption algorithm designed to provide both authentication and privacy. In other words, the CCM mode ensures confidentiality, integrity and freshness of the transferred packets. In RFC 3610, it is defined for use with AES. As the name suggests, CCM mode combines the well known counter mode of encryption with the well-known CBC-MAC mode of authentication. In a way this mode comprises two-step process.

- First, using a Cipher Block Chaining–Message Authentication Code (CBC–MAC) to calculate the MAC of the message.
- Second, AES–CTR to encrypt the data and the MAC.

The key insight is that the same encryption key can be used for both, provided that the counter values used in the encryption do not collide with the (pre-) initialization vector used in the authentication. CCM requires two block cipher encryption operations per each block of encrypted and authenticated message and one encryption per each block of associated authenticated data. ZigBee uses a modified version of this cipher mode called CCM*. In the following we describe AES-CTR and CBC-MAC Algorithm.

AES-CTR: Advanced Encryption Standard (AES) was developed by Joan Daemen and Vincent Rijmen and was standardized by NIST in 2002. The algorithm operates in fixed block sizes for data of 128, 192 and 256 Bits. AES is a symmetric algorithm that uses a shared key with 128, 192 or 256 Bits. AES-CTR uses AES in counter mode, which turns AES into a stream cipher. A counter (the nonce) is encrypted using the key associated with the link. This results in a set of blocks. These blocks are then used just like an encryption stream in a stream cipher and XORed with the message to be encrypted. Performing the same action again decrypts the data. In Figure 1 we describe the AEC–CTR process.

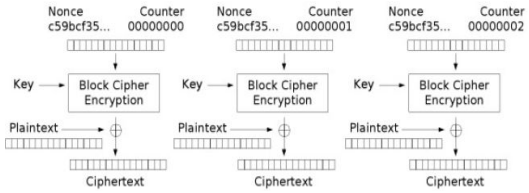

Figure 1: AES-CTR Process

According to the standard, AES-CTR offers encryption and freshness of data. However, the freshness protection is very limited. No integrity checking of the messages is built into this mode. Without integrity checking this encryption mode is weak and might expose other weaknesses in the protocol. It is trivial to manipulate packets and bypass the freshness counter, creating a DoS situation. For example by flipping a high bit in the freshness counter and force the receiving node to discard following traffic.

CBC-MAC: In this method the MAC is calculated covering the whole packet (header also included), by using CBC. The method can be summarized as below:

1) Start with an Initialization vector of 0.
2) XOR the vector with the first message block.
3) Encrypt the result with the current key.
4) The result of previous operation is XORed with the next message block.
5) Encrypt the result of Step 4.
6) Repeat Step 4 and Step 5 until all message blocks are considered.

In Figure 2, CBC-MAC process is described. The result is the MAC, which is added to the end of the original message. AES-CBC-MAC offers integrity and authentication of the messages.

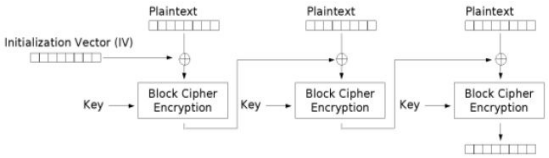

Figure 2: CBC-MAC Process

[12] Proposes a new IP Multimedia Subsystem (IMS) Service Authentication scheme using Identity Based Cryptography. [13] discusses security issues relevant to wireless sensor network such as key management, secure time synchronization, secure location discovery etc.

PROPOSED END TO END SECURITY APPROACH

In this section, we discuss our scheme to provide end-to-end security of data transmitted in a self-organizing wireless sensor network. End to end security means data sent between two communicating devices can't be seen by any third party or router. For a router (forwarding node), however it is important to know the packets it is forwarding has come from an authentic source. In such a situation transport layer PDU encryption and network layer PDU authentication is the best solution.

THREAT MODEL

Typical functions in a WSN include sensing and collecting data, processing and transmitting sensed data, possibly storing data for some time, and providing processed data as information e.g. to a so called sink node. A particular kind of processing that is essential is aggregation of data in the sensor nodes. Securing such functions turns out to be very challenging.

Communication network, in general has a threat model where two communicating parties A (Alice) and B (Bob) communicate over an insecure channel. If an intruder Trudy gains control over the communication network, she/he can overhear messages between the partners, intercept them and prevent their delivery to the intended recipient. But this threat model also assumes that the end-points, Alice and Bob, are not themselves subject to attack.

A WSN threat model is similar to the communication network threat model. Additionally, it is assumed that the endpoints cannot in general be trusted. An attacker may physically pick up sensor nodes and extract sensitive information.

TRUST MODEL

Trust model as suggested in our security framework is as follows: Trusted Node is a node which has successfully joined the network by supplying all the key credentials. Any new node can join the network with valid key credentials through a

trusted node. Without authentication, a trusted node does not start communication with a new node. Key Distribution Center pre-distributes certain Keys for joining the network initially. Public key of trusted a new node does not know node and base station till it authenticates itself with the pre-distributed key credentials. For communication, base station gives its public key to the newly joined network and announces public key of the newly joined node to the network.

ALGORITHM

Our proposed scheme to provide end-to-end security of data from source to destination is discussed in this section. As stated our scheme is a certificate less PKI based scheme where public key of each node is not known to a new node a-priori. Instead, only after successful joining of a new node public keys of a trusted node and the base station are given to it. Discussion of secure joining of a new node and key distribution are beyond the scope of this paper [1] [2].

Let us assume A needs to send data to B. Our scheme provides end-to-end security by encrypting Transport layer PDU (TPDU) with the public key of the receiver. And, authenticating the DL HEADER and the network layer PDU (NPDU) using some hashing function such as SHA-1 or MD-5 and private key of A. In Algorithm 1 we describe the steps performed on the sender side. Note that K_A^{Pub} is not transmitted with each packet for two reasons:

- Our scheme is based on certificate less public key. For this reason public keys are distributed to the nodes who have a symmetric key called 'One hop key' K_1. In other words public keys are made 'public' only to some authentic nodes [1] [2].
- Sending public key with packet each time will increase the packet size.

Figure 3 explains the Algorithm 1 diagrammatically.

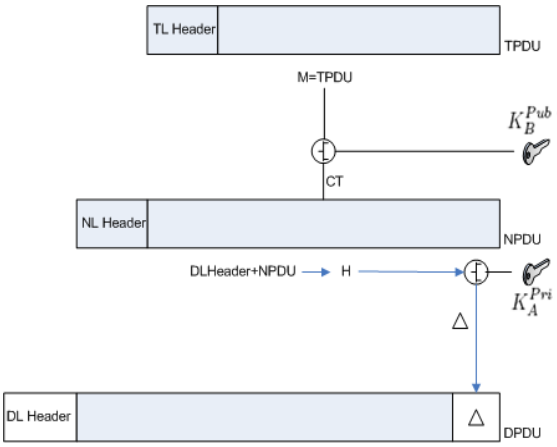

Figure 3: 'A' side protocol stack with encryption and signing

Input : Message/Data TPDU to send to B.
Output: TPDU is encrypted to have NPDU. Then, DL HEADER and NPDU is authenticated.

1 **begin**
2 A:
3 **begin**
4 **begin** Encryption of TPDU
   // A encrypts the TPDU using B's public key.
5 $M \leftarrow$ TPDU
6 $e : K_B^{Pub} \times M \rightarrow C$
7 NPDU $\leftarrow$ NL HEADER $+ C$
8 **end**
9 **begin** Authentication using DL HEADER and NPDU
10 $M' \leftarrow$ DL HEADER $+$ NPDU
   // Using h, where h can be SHA-1 or MD5, A generates H, the fixed size 'message digest'.
11 $h : M' \rightarrow H$
   // A generates signed digest Δ from H
12 $e : K_A^{Pri} \times H \rightarrow \Delta$
13 **end**
   // A sends DPDU to B
14 DPDU $\leftarrow$ DL HEADER $+$ NPDU $+ \Delta$
15 $A \rightarrow B$: DPDU
16 **end**
17 **end**

Algorithm 1: End-to-End Security Scheme: Sender A

Once A sends the packet to B there may be many intermediary nodes requiring forwarding the packet so as to reach its destination B finally. However, if an intermediary node does not verify the integrity and authenticity of the packet, and blindly forwards it, then we can't prevent flooding attack. In our scheme we take care of this problem by having any intermediary node verifying the integrity and authenticity of the packet first and then forward the packet. Additionally we ensure confidentiality of data. Algorithm 2 describes the steps. Note that–D can't see the message as it does not have the private key of B.

Finally, when B receives the packet first of all it verifies the integrity and authenticity of the packet and then decrypts the message using its private key. Steps are described in Algorithm 3.

RESULT ANALYSIS

We analyze our proposed scheme with the help of a wireless sensor network simulator in this section.

ESTIMATION OF COST

In this section we estimate cost of energy incurred by a sensor for encrypting and authenticating each packet sent. Similarly, we also estimate the cost of energy incurred by a sensor node for verifying a signature.

[7] compares energy consumed by RSA and ECC for generating and verifying signatures. In Table II we observe that while the cost of RSA verify is small, it is overshadowed by the more expensive sign operation, both of which required for authentication. In comparison ECDSA signatures are

significantly cheaper than RSA signatures and ECDSA verification are within reasonable range of RSA verification.

Input : Intermediate node D receives the packet from A for forwarding to B
Output: D verifies integrity of the $DPDU$ and forwards/drops the packet.

```
1  begin
2      D:
3      begin
           // D separates DL HEADER, NPDU, and Δ from DPDU.
           // Integrity and Authenticity check
4          M ← DL HEADER + NPDU
           // Using h, where h can be SHA-1 or MD5, D generates H', the fixed size 'message digest'.
5          h : M → H'
           // D generates H from Δ
6          d : K_A^Pub × Δ → H
7          if H = H' then
               // D forwards DPDU to B.
8              D → B : DPDU
9          else
10             D drops the DPDU
11     end
12 end
```

Algorithm 2: End-to-End Security Scheme: Intermediate node D

[7] mentions that one RSA sign operation is equivalent to transmitting 5,132 bytes, compared to 385 bytes for an ECDSA-160 sign operation. In other words if the size of a packet is X bytes then energy cost of one RSA sign is equivalent to transmitting 5132/X packets. Similarly, energy cost of one ECDSA sign is equivalent to transmitting 385/X packets.

Table 2: Energy Cost of Digital Signature

Algorithm	Sign	Verify
RSA-1024	304	11.9
ECDSA-160	22.82	45.09
RSA-2048	2302.7	53.7
ECDSA-224	61.54	121.98

In our proposed algorithm, we need to encrypt TPDU and then authenticate DPDU. Assuming encryption and authentication takes same amount energy, and our proposed authentication scheme consumes same amount of energy to ECDSA-160, the energy consumed by a sensor node for encryption and signing is equivalent to transmitting $2 \times \frac{385}{X}$ packets. If T is the cost of sending a packet by a sensor node, then cost of encrypting and authenticating a packet by our proposed method will be

$$\left(2 \times \frac{385}{X}\right)T$$

From Table II we see that ECDSA-160 verification cost is almost double to signing cost. Therefore we conclude that for our proposed algorithm, verification (checking for integrity and decryption) cost for each packet is

$$\left(4 \times \frac{385}{X}\right)T$$

We consider X = 41 bytes and average signing and verifying cost for each packet as

$$\left(3 \times \frac{385}{41}\right)T = 28.17T \approx 28T$$

Input : Receiver node B receives the packet from A.
Output: B verifies integrity of the $DPDU$ and then decrypts/drops the packet.

```
1  begin
2      B:
3      begin
           // B separates DL HEADER, NPDU, and Δ from DPDU.
           // Integrity and Authenticity check
4          M' ← DL HEADER + NPDU
           // Using h, where h can be SHA-1 or MD5, B generates H', the fixed size 'message digest'.
5          h : M' → H'
           // B generates H from Δ
6          d : K_A^Pub × Δ → H
7          if H = H' then
               // B accepts the DPDU and sends the packet to upper layer.
               // NPDU is separated to NL HEADER and C.
               // Using decryption function and private key of B TPDU is obtained.
8              d : K_B^Pri × C → TPDU
9          else
10             B drops the DPDU
11     end
12 end
```

Algorithm 3: End-to-End Security Scheme: Receiver B

SIMULATION

WSN Simulator: For simulation we modify and enhance the Wireless Sensor Network simulator v1.1 designed by [14]. The enhancements are done implementing our proposed algorithms in C# and integrating it with the WSN simulator. The simulation consists of two stages: deploying the network and running simulations. Before deploying the network, the properties of the network should be set using the configuration sliders. The GUI for the WSN Simulator is shown in Figure 4. Following properties are required to be set:

- Network Configuration
  - Network Size: Number of nodes in the network.
  - Sensor Radius: Proximity range of a sensor in the network.

- Sensor Period: Delay period between sensor detection events.
- Sensor Cost: The energy cost in detecting a vector and generating a packet.
- Transmission Radius: The maximum distance within which two network nodes can communicate.
- Transmitter Period: The amount of time required to send a packet.
- Transmit Cost: The energy cost in sending a packet.
- Receive Cost: The energy cost in receiving a packet.

- Routing Parameters
  - Random: Each node selects a downstream connection randomly for each packet.
  - Directed: The network routes packets based on the algorithm designed by [15]. Henceforth, we refer this algorithm as *AllPath* Algorithm.

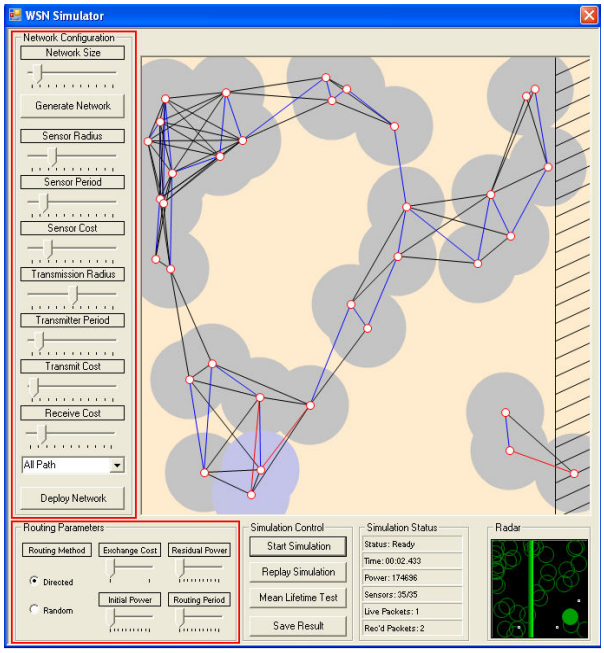

Figure 4: WSN Simulator GUI

Result Analysis: We simulate a WSN on the WSN simulator twice. In the first case data communication is done in plain text, where there is no encryption and authentications of packets are involved. In second case we use our algorithm for encryption and authentication of data packets and see their effect on overall performance of the network. In Figure 5 we compare residual energy of the WSN when data communication is done in plain text and when data communication is done with end to end security proposed by our algorithm. We observe that end to end security takes more energy as compared to plaintext. Similarly, In Figure 6 we compare the live nodes of the WSN. We found as the time passes more number of node dies in case of end to end security scheme as compared to data communication in plain text. This is due to depletion of energy for encryption and authentication of data by each sensor node.

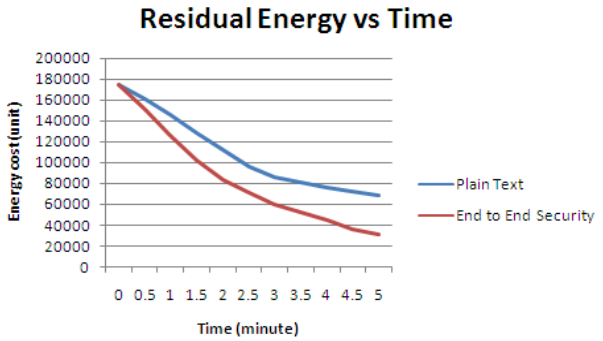

Figure 5: Residual Energy vs Time

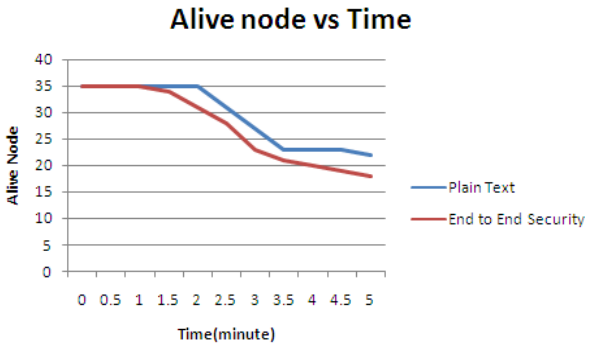

Figure 6: Live Node vs Time

COMPARISON

CCM uses symmetric key and hence generates only MIC (or, MAC) for checking integrity of the message. Using MIC integrity of the message can be checked; but authentication of the message can't be established. However, in case of CCM the MIC is generated using Key (Keyed Hash). Therefore, CCM Mode claims authentication is also provided, since the symmetric key is given only to the authenticated user. But in CCM mode non-repudiation property is not preserved as only, 'nonce' can give 'freshness' of data. However, it is possible for an attacker to modify bit of the nonce and cause DoS attack. CCM has been heavily criticized by [15]. A summary of those are –

- Efficiency: CCM is not efficient as compared to other cipher modes. It is not possible to process streaming data without knowing the full length of the data being processed using the CCM algorithm. In addition to this the algorithm disrupts word alignment in the encrypted data and does not allow pre-computation of the key stream.
- Parameterizations: One of the parameters that is set when employing CCM has strange effects. There is a parameter making tradeoffs between the maximum message length and the size of the nonce, two conceptually unrelated parameters, forcing shorter nonce when the maximum amount of data to be transferred grows. The user of a cipher mode should not have to make such a choice.
- Complexity: The cipher mode has a very high complexity, which in itself is an issue when it increases the risk of faulty implementations. Some seemingly innocent changes to the input parameters are enough to break the security of CCM. For example, the encoding does not allow two byte tags. This is due to the fact that one of bit 3,4 or 5 in the

first block must be non-zero or the security proof will not hold. There is also a large amount of bit manipulation in the cipher mode, making it much harder to understand the mode.

In our proposed method we preserve confidentiality, integrity, authenticity and non-repudiation of the message sent by any sensor node to the base station. Confidentiality of the message is preserved in the transport layer with the help of encryption of TPDU using the public key of the base station. The public key of the base station is known to the wireless sensor node after it successfully joins the network. No other node (adversary) can decrypt the cipher text as the secret key (private key) is only with the base station. In our method we use 'signed digest' for authenticating the message. Signed digest is equivalent to the digital signature and as we know a digital signature preserves authentication, integrity and non repudiation property of a message (but not confidentiality). Hence, in our case integrity, authenticity and non-repudiation property of the message is preserved in the network layer with the help of the signed digest (that is, signature on the message digest). Signed digest is a fixed length hash (generated using any known hashing algorithm like SHA-1 or MD5) and encrypted with the private key of the sensor node. By using verification algorithm any intermediate node or base station can verify integrity and authenticity of the message sent by a sensor node by using the public key of the sensor, which is known to the intermediate node. Also, non-repudiation property of the message is preserved, since similar signed digest can't be generated by other public key.

To summarize, we compare our method with CCM Mode as in Table III.

Table 3: CCM Mode vs Our Method

CCM Mode	**Our Method**
Symmetric	Asymmetric
No repudiation: No	Non repudiation: Yes
Complex	Simple
Key distribution: difficult	Key distribution: easy

CONCLUSION

In this paper we have proposed our end-to-end security scheme for secure transmission of data in wireless sensor network (Algorithm 1, 2 and 3). Our scheme is a certificate less PKI based scheme and provides better security than CCM Mode algorithm as it is based on public key cryptography. Due to encryption of data at higher layer and authentication at lower layer in our scheme, any intermediary node can't see the message but can verify the authenticity and integrity of the message. This prevents flooding attack by dropping malicious packets while on the route to the base station. However, as observed from simulation results providing end-to-end security is more costly in terms of energy as compared to data communication in WSN in plaintext.

REFERENCES

[1] H. K. Kalita and A. Kar, "Key management in secure self organized wireless sensor network: A new approach," 2010.

[2] A. Shamir, "Identity-based cryptosystems and signature schemes," CRYPTO 1984, Springer-Verlag, vol. LNCS 196, pp. pp. 47–53, 1985.

[3] J. Weise, "Public key infrastructure overview," 2001.

[4] A. Perrig, R. Szewczyk, V. Wen, D. Culler, and J. Tygar, "Spins: Security protocols for sensor networks," Mobile Computing and Networking, 2001.

[5] M. Dworkin, "Recommendation for block cipher modes of operation: The ccm mode for authentication and confidentiality," Natl. Inst. Stand. Technol. Spec. Publ. 800-38C 25 pages (May 2004), 2004.

[6] A. S. Wander, N. Gura, H. Eberle, V. Gupta, and S. C. Shantz, "Energy analysis of public-key cryptography for wireless sensor networks," 2005.

[7] F. Armknecht, A. Hessler, J. Girao, A. Sarma, and D. Westhoff, "Security solutions for wireless sensor networks," 2006.

[8] Q. Jing, J. Hu, and Z. Chen, "C4w: An energy efficient public key cryptosystem for large-scale wireless sensor networks," vol. Oct. 2006 Page(s):827 - 832. Mobile Adhoc and Sensor Systems (MASS), 2006 IEEE International Conference on, 2006.

[9] D. Boyle and T. Newe, "Securing wireless sensor networks: Security architectures," Journal of Networks, vol.3 (1), 2008.

[10] P. Soderman, "An analysis of wsn security managemant," Master's thesis, School of Computer Science and Engineering, Royal Institute of Technology, 2008.

[11] M. Abid, S. Song, H. Moustafa, and H. Afifi, "Integrating identity-based cryptography in ims service authentication," International Journal of Network Security & Its Applications (IJNSA), vol. Vol.1, No. 3, 2009.

[12] T. Kavitha and D. Sridharan, "Security vulnerabilities in wireless sensor networks: A survey," Journal of Information Assurance and Security 5 (2010) 031-044, 2010.

[13] D. J. Stein and Esq., Wireless Sensor Network Simulator v1.1, 2005.

[14] J.-H. Chang and L. Tassiulas, "Maximum lifetime routing in wireless sensor networks," IEEE/ACM Transactions on Networking (TON), vol. Volume 12 , Issue 4, 2004.

[15] P. Rogaway and D. Wagner, "A critique of ccm," 2003.

