**ISSN: 0975-766X**
**CODEN: IJPTFI**
***Available Online through*** ***Research Article***
**www.ijptonline.com**

# SECURITY IN SENSOR NETWORKS AND NECESSITY FOR SELF DESTRUCTION

**Sumangali K***
Department of Information Technology, School of Information Technology and Engineering,
VIT University, Vellore – 632 014, Tamilnadu, India.
*Email: sumivenkata@gmail.com*

*Received on 22-07-2016* *Accepted on 26-08-2016*

**Abstract**

The main objective of this paper is security in sensor networks. In general sensor networks are launched at high sensitive areas such as war fares where apart from the performance of the network, the security constraint is to be seriously considered. Though there are some security protocols like µ TESLA, SNEPdeveloped with concern to the sensor nodes that possess limited resources like memory, computation power, battery, and network bandwidth, these security protocols only provide authentication and confidentiality to the messages passed by the sensor nodes but the real toughest of the attacks that is hard to deal. Hence, node capturing effect where an enemy acquires full control over sensor nodes through direct physical contact and can obtain confidential information such as the network topology, encryption passwords and may also use the captured node to mislead the other nodes as well as the server node. Since we cannot practically demand an efficient access control to thousands of nodes distributed in a large territory and as it is very difficult to assure tamper-resistance requirements as the sensor nodes frequently need to be inexpensive to justify their use, there is more chance for the enemies to attack using this technique. In order to handle this type of issue, we introduce a self destruction mechanism to the sensor nodes where if a node is identified to be under this attack, self destructs itself and protects the valuable information from getting into the hands of enemies.

**Keywords**: µTESLA, SNEP, Selfdestruction, Security, Wireless sensor network.

## 1. Introduction

In recent times the incredible progress in micro-electromechanical systems and radio technologies has lead to the new concept wireless sensor networks[1]. A wireless sensor network, being a collection of tiny sensor nodes, proves to be a viable solution to many challenging civil and military applications despite with limited resources (limited coverage, low power, smaller memory sizes and low bandwidth), Their exploitation, sometimes in hostile environments, can be dangerously troubled by any type of sensor failure or, more harmful, by malicious attacks from an opponent. Sensor networks because of their specific limitations are vulnerable to various kinds of attacks that cannot be prevented only by traditional methods. Eavesdropping, traffic analysis, selective forwarding, spoofing, wormhole attack,  ink hole attack, Sybil attack and Hello flood attack are the most significant. But, almost certainly the most important danger, due to the inherent unattended characteristic of wireless sensor networks, is represented by node-capturing attack, where an enemy acquires full control over sensor nodes through direct physical contact[2]. A node capturing attack is very feasible because of at least two reasons: a) practically, we cannot demand an efficient access control to thousands of nodes distributed in a large territory; and b) it is very difficult to assure tamper-resistance requirements because sensor nodes frequently need to be inexpensive to justify their use[3]. After an attacker gains the physical control over a sensor node he can extract secret information such as cryptographic keys to achieve unrestricted entrance to higher network levels, or by using reverse engineering techniques he can find security holes to compromise the entire sensor network.

The proposed solution is based on the fact that a corrupted node is better to be excluded from the network as soon as its malicious activity is started. Even if more sensors are expelled, the WSN will function as designed because of one inherent feature, spatial redundancy. In order to identify a corrupted sensor node, we presumed that even if it may still send authenticated messages, it might not operate according to its original specifications sending incorrect readings to the base station. We will identify these sensors by using an autoregression technique and will eliminate them starting a self-destruction node procedure [4]

We organize the paper as follows. In section 2, we recall few basics related to wireless sensor networks, and some security schemes for wireless sensor networks related to proposed work. In section 3, the background of the proposed work is outlined to use them in our proposed work. Next, in section 4, we present the proposed work in a detailed manner with pseudo code.  Finally, we conclude this article with possible extensions and scope.

## 2. Related works

### 2.1 Wireless Sensor Networks

A wireless sensor network is an ad hoc network mainly comprising sensor nodes, which are normally used to monitor and observe a phenomenon or a scene. The sensor nodes are physically deployed within or close to the phenomenon or the scene. The collected data will be sent back to a base station from time to time through routes dynamically discovered and formed by sensor nodes. Sensors in wireless sensor networks are normally small network nodes with very limited computation power, limited communication capacity, and limited power supply. Thus a sensor may perform only simple computation and can communicate with sensors and other nodes within a short range. Also, the life spans of sensors are also limited by the power supply. Wireless sensor networks can be self-organizing, since sensors can be randomly deployed in some inaccessible areas. The randomly deployed sensors can cooperate with other sensors within their range to implement the task of monitoring or observing the target scene or the target phenomenon and to communicate with the base station that collects data from all sensor nodes[5]. The cooperation might involve finding a route to transmit data to a specific destination, relaying data from one neighbour to another neighbour when the two neighbours are not within reach of each other, and so on.

### 2.2 SPINS: Security Protocols for Sensor Networks

Sensor nodes in sensor networks are normally low-end devices with very limited resources, such as memory, computation power, battery, and network bandwidth. Perrig et al. in [6], proposed a family of security protocols named SPINS, which were specially designed for low end devices with severely limited resources, such as sensor nodes in sensor networks. SPINS consists of two building blocks: Secure Network Encryption Protocol (SNEP) and the “micro” version of the Timed, Efficient, Streaming, Loss-tolerant Authentication Protocol (μ TESLA). SNEP uses symmetry encryption to provide data confidentiality, two-party data authentication, and data freshness. μTESLA provides authentication over broadcast streams. SPINS assumes that each sensor node shares a master key with the base station. The master key serves as the base of trust and is used to derive all other keys.

### 2.3SNEP

From Figure 1we understand that, SNEP uses a block cipher to provide data confidentiality and message authentication code (MAC) to provide authentication. SNEP assumes a shared counter *C* between the sender and the receiver and two keys, the encryption key *K encr* and the authentication key *K mac*. For an outgoing message *D*, SNEP processes it as follows:

- The message *D* is first encrypted using a block cipher in counter mode with the key *K encr* and the counter *C*, forming the encrypted text *E* { *D* } ! *Kencr*, *C* "

- A message authentication code is produced for the encrypted text $E$ with the key $K\,mac$ and the counter $C$, forming the MAC $M.MAC(K\,mac\ , C \mid E\ )$ where $MAC()$is a one-way function and $C \mid E$ stands for the concatenation of $C$ and $E$ .
- SNEP increments the counter $C$.

To send the message $D$ to the recipient, SNEP actually sends out $E$ and $M$ . In other words, SNEP encrypts $D$ to $E$ using the shared key $K\,encr$ between the sender and the receiver to prevent unauthorized disclosure of the data, and it uses the shared key $K\,mac$, known only to the sender and the receiver, to provide message authentication. Thus data confidentiality and message authentication can both be implemented. The message $D$ is encrypted with the counter $C$, which will be different in each message. The same message $D$ will be encrypted differently even it is sent multiple times. Thus semantic security is implemented in SNEP. The MAC is also produced using the counter $C$; thus it enables SNEP to prevent replying to old messages.

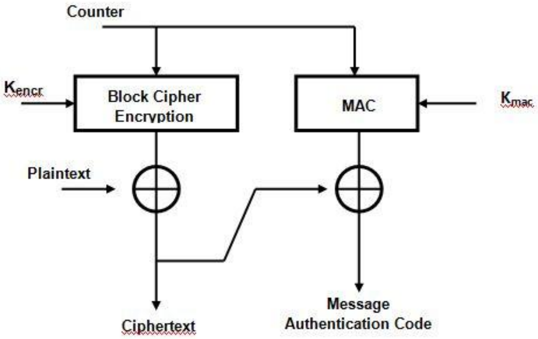

**Fig.1: SNEP Architecture.**

### 2.4 µ TESLA

TESLA [7] was proposed to provide message authentication for multicast. TESLA does not use any asymmetry cryptography, which makes it lightweight in terms of computation and overhead of bandwidth. µTESLA is a modified version of TESLA, aiming to provide message authentication for multicasting in sensor networks. The general idea of µ TESLA is that the sender splits the sending time into intervals. Packets sent out in different intervals are authenticated with different keys. Keys to authenticate packets will be disclosed after a short delay, when the keys are no longer used to send out messages. Thus packets can be authenticated when the authentication keys have been disclosed. Packets will not be tampered with while they are in transit since the keys have not been disclosed yet. The disclosed authentication keys can be verified using previous known keys to prevent malicious nodes from forging authentication keys. µ TESLA has four phases: sender setup, sending authenticated packets, bootstrapping new receivers, and authenticating packets. In the sender setup phase, a sender generates a chain of keys, K i (0 <= i <= n ). The keychain is a one-way chain such that K i can be derived from K j if i <= j , such as a keychain K i (i = 0, … ,n), Ki = F(Ki + 1 ), where F is a one-way function. The sender also decides on the starting time T0 , the interval duration Tint , and the disclosure delay d (unit is interval), as shown in Figure[figure no] . To send out authenticated packets, the sender attaches a MAC with each packet, where the MAC is produced using a key from the keychain and the data in the network packet. µ TESLA has specific requirements on the Sensor Network Encryption Protocol (SNEP) namely; Sequences of intervals, key usages, and key disclosure. Use of keys for producing MACs. Keys are used in the same order as the key sequence of the keychain. Each of the keys is used in one interval only. For the interval Ti = T0 + i x Tint, the key Ki is used to produce the MACs for the messages sent out in the interval Ti. Keys are disclosed with a fixed delay d such that the key Ki used in interval Ti will be disclosed in the interval Ti + d. The sequence of key usage and the sequence of key disclosure are demonstrated in Figure. To bootstrap a new receiver, the sender needs to synchronize the time with the receiver and needs to inform the new receiver of a key Kj that is used in a past interval Tj , the interval duration T int , and the disclosure delay d . With a previous key Kj, the receiver will be able to verify any key K p where j <= p using the one-way key chain's property. After this, the new receiver will be able to receive and verify data in the same way as other receivers that join the communication prior to the new receiver. To receive and authenticate messages, a receiver will check all incoming messages if they have been delayed for more than d. Messages with a delay greater than d will be discarded, since they are suspect as fake messages constructed after the key has been disclosed. The receiver will buffer the remaining messages for at least d intervals until the corresponding keys are disclosed. When a key K i is disclosed at the moment T i + d, the receiver will verify Ki using K i - 1 by checking if K i-1 = F ( Ki ). Once the key K i is verified, K i will be used to authenticate those messages sent in the interval Ti.

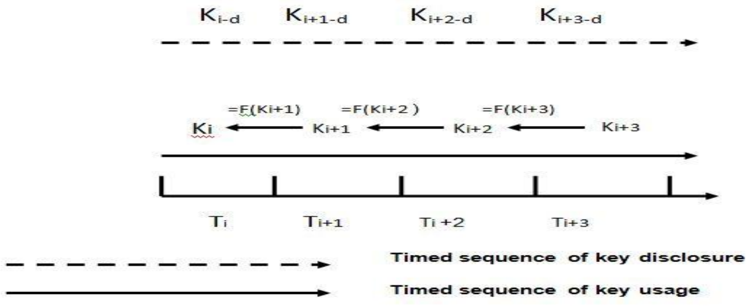

**Fig. 2: µ TESLA Architecture.**

### 2.5 Node capturing attack

Node capture attacks result from the combination of passive, active, and physical attacks by an intelligent adversary[8]. In order to initialize or set up an attack, the adversary will collect information about the WSN by eavesdropping on message exchanges, either local to a single adversarial device or throughout the network with the aid of a number of adversarial devices deployed throughout the network.

Even if message payloads are encrypted, the adversary can extract information about the network operation and state, effectively learning about the network structure and function. In addition to passive learning, the adversary can actively participate in network protocols, probing the network for information and maliciously injecting information into the network. Once a sufficient amount of passive and active learning has taken place, the adversary can physically capture nodes[7].

## 3. The Self Destruction Architecture with Problem Statement and Proposed Solution

### 3.1 Problem Statement

The above security protocols μ TESLA, SNEP are developed with concern to the sensor nodes which possess limited resources like memory, computation power, battery, and network bandwidth [7]. But the problem with these security protocols only provide authentication and confidentiality to the messages passed by the sensor nodes but the real toughest of the attacks that is hard to deal with is the NODE CAPTURING effect where an enemy acquires full control over sensor nodes through direct physical contact and can obtain confidential information like the network topology, encryption passwords and may also use the captured node to mislead the other nodes as well as the server node. Since we cannot practically demand an efficient access control to thousands of nodes distributed in a large territory and as it is very difficult to assure tamper-resistance requirements as the sensor nodes frequently need to be inexpensive to justify their use, there is more chance for the enemies to attack using this technique.

### 3.2 Proposed solution

In this section, we discuss [discussed] the design of our self-destructing system that analyzes the results given by sensor nodes, compares them with the predicted results and in case if the difference between them exceeds a certain value for a specific number of times then the node is subjected to destruct itself[9].

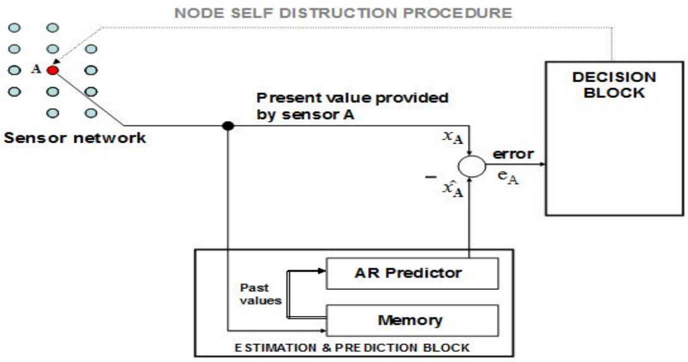
NODE SELF DISTRUCTION PROCEDURE
A
Sensor network
Present value provided by sensor A
$x_A$
error
$e_A$
$-\hat{x}_A$
DECISION BLOCK
AR Predictor
Past values
Memory
ESTIMATION & PREDICTION BLOCK

**Fig.3: The Proposed Architecture.**

for every node A in the sensor network, the output resulted by the node is stored in the memory of estimation and prediction block, based on the values stored in the memory an auto regression predictor predicts the value from the node A. now, the real value is compared to the predicted value, if the error rate is more than a certain level, decision block gets activate, up on repetition of few such errors the node is identified as malicious node as a result of which a self destruction method gets started where the node destructs itself [10].

**3.3 Pseudo code for mechanism**

```
FOR (every sensor node)
{
…
SET $b_A$,$\varepsilon_A$, k; //node trust indicator, threshold and the
//approximation of length of predictor’s transitory regime
}
WHILE (network is active)
{
…
FOR (every sensor node)
{
$x_A$=READ sensor A; //get sensor actual value
…
$\hat{x}_A$=PREDICT (prior $x$Avalues); //call prediction method
$e_A$= $x_A$- $\hat{x}_A$; //calculate the error
IF (ABS ($e_A$) is greater than$\varepsilon_A$ )
IF (predictor is not in transitory regime)
{
$b_A$= $b_A$+ 1; //increment node trust indicator
START thread TRANSITORY_REGIME;
//a counter set on k and will be decremented
//every instant until it becomes zero
DECISION_BLOCK (sensor A); //call decision method
} }
```

**Prediction method**

```
float PREDICT (prior $x_A$values)
{
CALCULATE auto regression coefficients $a_i$ ;
// an estimation using QRD-RLS method
CALCULATE predicted value $\hat{x}_A$;
// compute sensor predicted value as a result of
RETURN $\hat{x}_A$;
}
```

**Decision box**

```
void DECISION_BLOCK (sensor A)
{
IF ( bA greater or equal than $\gamma$ ) //sensor is corrupted
{
SELF_DESTRUCTION (sensor A) ;
//call self-destruction mechanism for sensor A
        }
        }
```

**Self destruction method**

```
        void SELF_DESTRUCTION (sensor A)
        {
        WHILE (sensor A is in network)
        {
        START thread
        CONSUME battery energy //broadcast specific messages;
        START thread
        {
        ERASE node memory; //erase RAM and flash memory
        DISABLE auto-organization property; //delete node id from all neighbour's lists
        DESTROY node radio device;
        MASK node measurement nature;   //for hiding the type of the sensor
        }
        }
```

## 4. Conclusion

Thus implementing the above mechanism an effected node can be self destructed and there by the sensitive information namely, the network topology and Cryptographic keys can be safe guarded from the attackers. But the real problem in this mechanism is improper implementation of this mechanism may result in unnecessary destruction of the node which is not desirable though destruction of single node do not affect the performance of the network as a whole. As a part of future enhancement, the values from nodes that are physically nearer to the malicious node can be given higher priority in predicting the results as a result of which unnecessary node destruction can be eliminated in extreme environments where it is likely for the node to result values that vary from predicted values.

## References

1. S.Movassaghi, J.Lipman, Wireless Body Area Networks: A Survey, IEEE communications surveys & tutorials, 2014, Vol. 16, No. 3, PP1658-1686.

2. C. Karlof and D. Wagner, "Secure routing in wireless sensor networks: attacks and countermeasures", In Proceedings of the 1st IEEE International Workshop SNPA2003, Anchorage, USA, PP113-127 (2003).

3. D.I. Curiac, C. Volosencu, A. Doboli, O. Dranga and T. Bednarz, Discovery of Malicious Nodes in Wireless Sensor Networks using Neural Predictors, WSEAS Transactions on Computer Research,2007, Vol. 2, No.1, PP38-43.

4. I. M. Atakli, H. Hu, Y. Chen,W. S. Ku, Z. Su , "Malicious Node Detection in Wireless Sensor Networks using Weighted Trust Evaluation", In Symposium on Simulation of Systems Security (SSSS'08), Ottawa, Canada, PP836-843 (2008).

5. S.Chitnis, N.Deshpande, A.Shaligram, An Investigative Study for Smart Home Security: Issues, Challenges and Countermeasures, Wireless Sensor Network, 2016, Vol.8, PP61-68.

6. A. Perrig, J. Stankovic, D.Wagner, Security in wireless sensor networks, Communications of the ACM, 2004,Vol. 47, No. 6, PP53-57.

7. A. K.Pathan, H.W. Lee, C. S. Hong, Security in Wireless Sensor Networks: Issues and Challenges, In 8th International Conference Advanced Communication Technology, Phoenix Park, Korea, PP1043-1048 (2006).

8. D.I.Curiac, M.Plastoi, A.Doboli "Combined Malicious Node Discovery and Self-Destruction Technique for Wireless Sensor Networks", Inthird International Conference on Sensor Technologies and Applications, Athens, Glyfada, PP436 – 441 (2009).

9. M.Plastoi, D.I.Curiac, "Energy-Driven Methodology for Node Self- Destruction in Wireless Sensor Networks", In 5th International Symposium on Applied Computational Intelligence and Informatics, Timişoara, Romania, (2009).

10. M.Plastoi, O.Banias, D.I.Curiac, C.Volosencu, R.Tudoroiu, A.Doboli, "Integrted system for malicious node discovery and self-destruction in wireless sensor networks", International Journal on advances in networks and services,2009, Vol. 2, No.4 , PP241-250.

**Corresponding Author:**

**Sumangali K*,**

**Email:** sumivenkata@gmail.com