Performance Evaluation on Secure Transaction Protocols in WSN

K. Rajasekaran, Dr.Kannan Balasubramanian

Abstract— Wireless Sensor Networks(WSN) are extremely vulnerable against any kind of internal or external attacks, due to several factors such as resource-constrained nodes and lack of tamper-resistant packages. As a result, security must be an important factor to have in mind when designing the infrastructure and protocols of sensor networks. In this paper we survey the \state-of-the-art" security issues in sensor networks and highlight the open areas of research. Wireless Sensor networks have great potential to be employed in mission critical situations like battlefields but also in more everyday security and commercial applications such as building and traffic surveillance, habitat monitoring and smart homes etc. However, Wireless sensor networks pose unique challenges. While the deployment of sensor nodes in an unattended environment makes the networks vulnerable to a variety of attacks, the inherent power and memory limitations of sensor nodes makes conventional security solutions unfeasible. Though there has been some development in the field of sensor network security, the solutions presented thus far address only some of the problems faced. This research presents a security framework to provide a comprehensive security solution in sensor networks. The proposed framework consists of four components: a secure triple-key scheme, secure routing algorithms, secure localization technique and a malicious node detection mechanism. Singly each of the components can achieve certain level of security. However when deployed as a framework, high degree of security is achievable. Experimental results show that the proposed framework can achieve high degree of security with negligible overheads.

Index Terms — Secure Transaction, secure triple-key, Adhoc, malicious node detection, secure routing algorithms, DDoS, Antnet

1. Introduction

Wireless sensor networks are consisting of large number of tiny sensors and actuators with limited energy, computations and transmission power. Sensor nodes are randomly deployed in an environment where they are prone to physical interaction and most likely left unattended after deployment. Although nodes have many limitations but they report to a single destination called base station which is believed to be a powerful computer safely located with large computation resources.

We consider a hierarchical topology of sensor networks where sensor nodes form a parent child relationship in clusters when deployed. In this topology, nodes broadcast their IDs and listen to the neighbors, add the neighbors IDs in its routing table and count the number of neighbors it could listen to. Hence these connected neighbors become a cluster. Each cluster elects a sensor node as a leader. All inter-cluster communication is routed through cluster leaders. Cluster leaders also serve as fusion nodes to aggregate packets and send them to the base station.

• *Research Scholar,CSE, Manonmaniam Sundaranar University, Tirunelveli, Tamilnadu, India, PH:+91-9841358088, tenza79@gmail.com*
• *Associate Professor, CSE, Mepco Schlenk Engineering College,Sivakasi, Tamilnadu, India, PH: +91-9442378500,kannanbala@mepcoeng.ac.in*

A cluster leader receives highest number of messages, this role changes after reaching an energy threshold, hence giving opportunity to all nodes becoming a cluster leader when nodes move around in a dynamic environment. Coverage of cluster depends on the signal strength of the cluster leader. Cluster leader and its neighbor nodes form a parent-child relationship in a tree-based network topology. In this multi hop cluster model, data is collected by the sensor nodes, aggregated by the cluster leader and forwarded to the next level of cluster leader, eventually reaching the base station. Due to the deployment nature, nodes are highly vulnerable to localization attacks from compromised networks and malicious nodes. In this paper we argue that using four triangulation methods and a secure set of keys, impact of localization attacks can be reduced or eliminated in optimum scenarios.

Routing messages in a network is an essential component of Internet communication, as each packet in the Internet must be passed quickly through each network (or autonomous system) that it must traverse to go from its source to its destination. It should come as no surprise, then, that most methods currently deployed in the Internet for routing in a network are designed to forward packets along shortest paths. Indeed, current interior routing protocols, such as OSPF, RIP, and IEGP, are based on this premise, as are many exterior routing protocols, such as BGP and EGP. The algorithms that form the basis of these protocols are not secure, however, and have even been compromised by routers that did not follow the respective protocols correctly. Fortunately, all network malfunctions resulting from faulty routers have to date been shown to be the result of misconfigured routers, not malicious attacks. Nevertheless, these failures show the feasibility of malicious router attacks, for they demonstrate that compromising a single router can undermine the

performance of an entire network.
Adhoc networks have enormous impact on many aspects such as emergency medical care and military services where security of data is very important. The secure path establishment procedure plays a vital role in the MANET security mechanism. The efficiency of secure route discovery will be evaluated by the factors like prevention of DoS attacks on data traffic, high speed, low energy overhead and successful secure link establishment among neighbors. The flat network layout is good for small networks but does not scale well with increase in network size since the nodes in the neighborhood of the base station are flooded by route requests and replies. Also in large networks the average number of hops to the base station increases, which means the energy consumption for route requests and replies, increases drastically. Due to larger distances, the end-to-end data latency also increases. In order to overcome these problems, ECC security mechanism and a prediction based proactive hierarchical network structure has been considered. This research focuses on establishing efficient secure routing in clustered based adhoc networks by a two fold process viz.,. Estimation of trust values of neighbours. Secure end-to-end route discovery using Antnet routing mechanism and mutual authentication using ECC.
In this secure routing mechanism, each MN in the cluster maintains the trust value of its one-hop neighbors. Trust is nothing but the measure of uncertainty about the node (trust value is associated to successful packet forwarding) and it can be measured by entropy. In this research the trust relationship of neighborhood node is evaluated by the recommendation of third party. That is, by observing the trust value of the third party for the particular packet transmission, the trust value of neighbors can be predicted. The secure path will be evaluated and established using ECC since it offers an excellent level of security with lower key sizes.

Recent advancements in micro-electro-mechanical systems (MEMS) and low power and highly integrated electronic devices have led to the development and wide application of wireless sensor networks. Wireless sensor networks consist of very small devices, called sensor nodes, that are battery powered and are equipped with integrated sensors, a data-processing unit, a small storage memory, and short-range radio communication. Typically, these sensors are randomly deployed in the field. They form an unattended wireless network, collect data from the field, partially aggregate them, and send them to a sink that is responsible for data fusion. Sensor networks have applications in emergency-response networks, energy management, medical monitoring, logistics and inventory management, and battlefield management.
In contrast to traditional wireless networks, special security and performance issues have to be carefully considered for sensor networks. For example, due to the unattended nature of sensor networks, an attacker could launch various attacks and even compromise sensor devices without being detected. Therefore, a sensor network should be robust against attacks, and if an attack succeeds, its impact should be minimized. In other words, compromising a single sensor node or few sensor nodes should not crash the entire network.
Another concern is about energy efficiency. In a WSN, each sensor node may need to support multiple communication models including unicast, multicast, and broadcast. Therefore, due to the limited battery lifetime, security mechanisms for sensor networks must be energy efficient. Especially, the number of message transmissions and the amount of expensive computation should be as few as possible. In fact, there are a numbers of attacks an attacker can launch against a wireless sensor network once a certain number of sensor nodes have been compromised. In literature, for instance, HELLO flooding attacks, sink hole attacks, Sybil attack, black hole attack, worm hole attacks, or DDoS attacks are options for an attacker. These attacks lead to anomalies in network behaviors that are detectable in general. There are some reported solutions to detect these attacks by monitoring the anomalies.
The rest of the paper is structured as follows. In section 2, we briefly review the Security Goals. Section 3 describes the Securing the node location. The network framework and the flooding algorithm is presented in Section 4. Section 5 describes the Securing the flooding algorithm on General networks. The Securing setup for the Distance vector routing algorithm is discussed in Section 6. Section 7 presents the Secure Routing algorithm. The Malicious nodes detection is given n Section 8. Finally the conclusions are discussed in Section 9.

1. Security Goals

The Objective of this paper is to design the Security methods that achieve the following properties:
Fault detection. The algorithm should run correctly and, in addition, should detect any computational steps that would compromise the correctness of the algorithm.
Damage containment. The algorithm should contain the damage caused by an incorrect router to as small an area of the network as possible.
Authentication. The algorithm should confirm that each message is sent from the host or router that the message identifies as its source.
Data integrity. The algorithm should confirm that the contents of received messages are the same as when they are sent, and that all components of a message are as intended by the algorithm (even those message portions added by routers other than the original sender).
Timeliness. The algorithm should confirm that all messages interacting to perform this algorithm are current up-to-date messages, thereby preventing replay attacks.

2. Securing the Node Location

Nodes change its position if they move in a dynamic network or if an adversary has compromised the node. In the event of compromise a node is considered as a malicious node. The localization process described here is protected by a secure triple-key management scheme which consists of three keys: two pre-deployed keys in all nodes and one in-network generated cluster key for a cluster to address the hierarchical nature of sensor network.
Kn (network key) – Generated by the base station, pre-deployed in each sensor node, and shared by the entire sensor network. Nodes use this key to encrypt the data and pass onto next hop.
Ks (sensor key) – Generated by the base station, pre-deployed in each sensor node, and shared by the entire sensor network. Base station uses this key to decrypt and process the data and cluster leader uses this key to decrypt the data and send to base station.
Kc (cluster key) – Generated by the cluster leader, and shared by the nodes in that particular cluster. Nodes from a cluster use this key to decrypt the data and forward to the cluster leader.
Base station broadcasts a beacon message to the sensor network; this message is encrypted by the Kn. If the receiving node is a cluster leader it decrypts the message using Ks and encrypts it again with its Kn and forwards it to the nodes in its cluster. Nodes in the cluster use its Kc to decrypt the message, adds its location and reply back to the cluster leader with its location encrypted with Kn. Cluster leader receives the locations from all nodes in the cluster and encrypts it with Kn and sends it to the base station. Base station uses its Ks to decrypt the message and becomes aware of the nodes location in the entire network. The process of base station to cluster leader and nodes and vice versa is described in the following steps:
Step 1: To establish the secure communication, base station builds a packet which contains:
IDBS, Kn, TS, MAC, S (message)
Step 2: Cluster leader builds a packet containing following information:
IDCL, Kn, TS, MAC, S (message)
Step 3: Nodes to cluster leader packet consists of:
IDsn, kn, TS, MAC, S (message)
Step 4: Cluster leader aggregates the messages received from the nodes in its cluster and forwards it to the base station using the packet: IDCL, Kn, TS, MAC, S (Aggr message). Fig. 1 below illustrates the key calculation process among the nodes, cluster leaders and the base station.

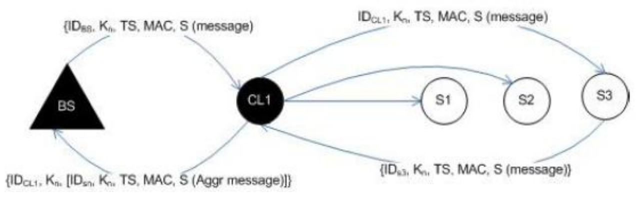
{ID_{BS}, K_n, TS, MAC, S (message)
ID_{CL1}, K_n, TS, MAC, S (message)
BS
CL1
S1
S2
S3
{ID_{CL1}, K_n, [ID_{sn}, K_n, TS, MAC, S (Aggr message)]}
{ID_{s3}, K_n, TS, MAC, S (message)}

Fig. 1 Key Calculation using the Secure triple Keys

IDBS: Base station ID
Kn: Network Key
Ks: Sensor Key
Kc: Cluster Key
IDCL: Cluster leader ID
IDsn: Sensor node ID
TS: An encrypted time stamp for beacon authentication
S: Seed value randomly generated by the base station
Aggr message: Aggregated message by a cluster leader
MAC: Message authentication code for message m, generated using key k
BS: Base station – a node assumed to be very powerful with extra ordinary computation resources

3. The Network Framework and the Flooding Algorithm

Let G = (V;E) be a network whose vertices in V are routers and whose edges in E are direct connections between routers. We assume that the routers have some convenient addressing mechanism that allows us without loss of generality to assume that the routers are numbered 1 to n. Furthermore, we assume that G is biconnected, that is, that it would take at least two routers to fail in order to disconnect the network. This assumption is made both for fault tolerance, as single points of failure should be avoided in computer networks, and also for security reasons, for a router at an articulation point can fail to route packets from one side of the network to the other without there being any immediate way of discovering this abuse.
The flooding algorithm is initiated by some router s creating a message M that it wishes to send to every other router in G. The typical way the flooding algorithm is implemented is that s incrementally assigns sequence numbers to the messages it sends. So that if the previous message that s sent had sequence number j, then the message M is sent with sequence number j + 1 and an identification of the message source, that is, as the message (s; j +1;M). Likewise, every router x in G maintains a table Sx that stores the largest sequence number encountered so far from each possible source router in G. Thus, any time a router x receives a message (s; j + 1;M) from an adjacent router y the router x first checks if Sx[s] < j +1. If so, then x assigns Sx[s] = j +1 and x sends the message (s; j +1;M) to all of its adjacent routers, except for y. If the test fails, however, then x assumes it has handled this message before and it discards the message. If all routers perform their respective

tasks correctly, then the flooding algorithm will send the message M to all the nodes in G. Indeed, if the communication steps are synchronized and done in parallel, then the message M propagates out from s is a breadth-first fashion. If the security of one or more routers is compromised, however, then the flooding algorithm can be successfully attacked. For example, a router t could spoof the router s and send its own message (s; j + 1;M0). If this router reaches a router x before the correct message, then x will propagate this imposter message and throw away the correct one when it finally arrives. Likewise, a corrupted router can modify the message itself, the source identification, and/or the sequence number of the full message in transit. Each such modification has its own obvious bad effects on the network. For example, incrementing the sequence number to j+m for some large number m will effectively block the next m messages from s. Indeed, such failures have been documented (e.g., [13, 12]), although many such failures can be considered router mis configuration not malicious intent. Of course, from the standpoint of the source router s the effect is the same independent of any malicious intent all flooding attempts will fail until s completes m attempted flooding messages or s sends a sequence number reset command (but note that the existence of unauthenticated reset commands itself presents the possibility for abuse).

4. SECURING THE FLOODING ALGORITHM ON GENERAL NETWORKS

On possible way of avoiding the possible failures that compromised or mis configured routers can inflict on the flooding algorithm is to take advantage of a public-key infrastructure defined for the routers. In this case, we would have s digitally sign every flooding message it transmits, and have every router authenticate a message before sending it on. Unfortunately, this approach is computationally expensive. It is particularly expensive for overall network performance, for, as we discuss later in this paper, flooding is often an important substep in general network administration and setup tasks.

Our scheme is based on a light-weight strategy, which we call the leap-frog strategy. The initial setup for our scheme involves the use of a public-key infrastructure, but the day-to-day operation of our strategy takes advantage of much faster cryptographic methodologies. Specifically, we define for each router x the set N(x), which contains the vertices (routers) in G that are neighbors of x (which does not include the vertex x itself). That is,

N(x) = fy: (x; y) 2 E and y 6= xg:

The security of our scheme is derived from a secret key k(x) that is shared by all the vertices in N(x), but not by x itself. This key is created in a setup phase and distributed securely using the public-key infrastructure to all the members of N(x). Note, in addition, that y 2 N(x) if and only if y 2 N(y). Now, when s wishes to send the message M as a flooding message to a neighboring router, x, it sends (s; j + 1;M; h(sjj + 1jMjk(x)); 0), where h is a cryptographic hash function that is collision resistant (e.g., see [10]. Any router x adjacent to s in G can immediately verify the authenticity of this message (except for the value of this application of h), for this message is coming to x along the direct connection from s. But nodes at distances greater than 1 from s cannot authenticate this message so easily when it is coming from a router other than s. Fortunately, the propagation protocol will allow for all of these routers to authenticate the message from s, under the assumption that at most one router is compromised during the computation.

Let (s; j +1;M; h1; h2) be the message that is received by a router x on its link from a router y. If y = s, then x is directly connected to s, and h2 = 0. But in this case x can directly authenticate the message, since it came directly from s. In general, for a router x that just received this message from a neighbor y with y 6= s, we inductively assume that h2 is the hash value h(sjj + 1jMjk(y)). Since x is in N(y), it shares the key k(y) with y's other neighbors; hence, x can authenticate the message from y by using h2. This authentication is sufficient to guarantee correctness, assuming no more than one router is corrupted at present, even though x has no way of verifying the value of h1. So to continue the propagation assuming that flooding should continue from x, the router x sends out to each w that is its neighbor the message (s; j + 1;M; h(Mjj + 1jk(w)); h1). Note that this message is in the correct format for each such w, for h1 should be the hash value h(sjj+1jMjk(x)), which w can immediately verify, since it knows k(x). Note further that, just as in the insecure version of the flooding algorithm, the _rst time a router w receives this message, it can process it, updating the sequence number for s and so on. This simple protocol has a number of performance advantages. First, from a security standpoint, inverting or finding collisions for a cryptographic hash function is computationally difficult. Thus, it is considered infeasible for a router to fake a hash authentication value without knowing the shared key of its neighbors, should it attempt to alter the contents of the message M. Likewise, should a router choose to not send the message, then the message will still arrive, by an alternate route, since the graph G is biconnected. The message will be correctly processed in this case as well, since a router is not expecting messages from s to arrive from any particular direction. That is, a router x does not have to wait for any other messages or verifications before sending in turn a message M on to x's neighbors. Another advantage of this protocol is its computational efficiency. The only additional work needed for a router x to complete its processing for a flooding message is for x to perform one hash computation for each of the edges of G that are incident on x. That is, x need only perform degree(x) hash computations, where degree(x) denotes the degree of x. Typically, for communication networks, the degree of a router is kept bounded by a constant. Thus, this work compares quite favorably in practice to the computations that would be required to verify a full-blown digital signature from a message's source. The leap-frog routing process can detect a router malfunction in the flooding algorithm, for any router y that does not follow the protocol will be discovered by one of its neighbors x. Assuming that x and y do not collude to suppress the discovery of y's mistake in this case, then x can report to s or even a network administrator that something is potentially wrong with y. For in this case, y has clearly not followed the protocol. In addition, note that this discovery will occur in just one message hop from y.

6. Securing the Setup for Distance Vector Routing Algorithm

Since the main algorithmic portion in testing the correctness of a round of the distance-vector algorithm involves validating the computation of a minimum of a collection of values, let us focus more specifically on this problem. Suppose, then, that we have a node x that is adjacent to a collection of nodes y0, y1, : : :, yd−1, and each node yi sends $m = \min_{i=0,1,\ldots,d-1} \{a_i\},$ ask x is to perform is to compute

in a way that all the yi's are assured that the computation was done correctly. As in the previous sections, we will assume that at most one router will be corrupted during the computation (but we have to prevent and/or detect any fallout from this corruption). In this case, the router that we consider as possibly corrupted is x itself. The neighbors of x must be able therefore to verify every computation that x is to perform. To aid in this verification, we assume a preprocessing step has shared a key k(x) with all d of the neighbors of x, that is, the members of N(x), but is not known by x.

The algorithm that x will use to compute m is the trivial minimum-finding algorithm, where x iteratively computes $m_j = \min_{i=0,\ldots,j} \{a_i\},$ ıimum values

for j = 0; : : : ; d−1. Thus, the output from this algorithm is simply m = md−1. The secure version of this algorithm proceeds in four communication rounds:

1. Each router yi sends its value ai to x, as Ai = (ai; h(aijk(x)), for i = 0; 1; : : : ; d − 1.
2. The router x computes the mi values and sends the message (mi−1;mi;Ai−1 mod d;Ai+1 mod d) to each yi. The validity of Ai−1 mod d and Ai+1 mod d) is checked by each such yi using the secret key k(x). Likewise, each yi checks that mi =minfmi−1; aig.
3. If the check succeeds, each router yi sends its verification of this computation to x as Bi = (\yes00; i;mi; h(\yes00jmijijk(x))). (For added security yi can seed this otherwise short message with a random number.)
4. The router x sends the message (Bi−1 mod d;Bi+1 mod d) to each yi. Each such yi checks the validity of these messages and that they all indicated \yes" as their answer to the check on x's computation. This completes the computation.

In essence, the above algorithm is checking each step of x's iterative computation of the mi's. But rather than do this checking sequentially, which would take O(d) rounds, we do this check in parallel, in O(1) rounds.

7. Secure Routing Algorithm

Trust estimation: This section deals with the trust estimation of the neighborhood nodes. The basic understanding of the trust is summarized as follows:

 Trust is a relationship established between two entities for a specific action. In particular, one entity trusts the other entity to perform an action. In this study, the first entity is called the subject, the second entity is called the agent. So, the notation used to describe a trust relationship is {subject: agent; action} Let T {subject: agent; action} denote the trust value of the trust relationship {subject: agent; action} and P{subject: agent; action} denote the probability that the agent will perform the action in the subject’s point of view. Information theory states that entropy is a measure of uncertainty; thus, the entropy-based trust value as

Error!

$$T\{\text{trust : agent, action}\} = \begin{cases} I - H(p), \text{for } 0.5 \le p \le 1 \\ H(p) - 1, \text{for } 0 \le p < 0.5 \end{cases}$$

H(p) = ˜ . p log 2 p˜ . (1̃ p) log 2 (1̃ p)
P = P {subject: agent; action}

In this work, the trust value is a continuous real number [-1, 1]. This definition satisfies the following properties.

 When P = 1, the subject trusts the agent the most and the trust value is 1

 When P = 0, the subject distrusts the agent the most and the trust value is -1

 When P = 0.5, the subject has no idea about the agent and the trust value is 0

In general, trust value is negative for 0 P 0.5 and positive for 0.5 P 1. Trust value is **a** increasing function with P. It is a one-to-one mapping between T {subject: agent; action} and P {subject: agent; action}. One typical example, A wants to establish the trust relationship with B (A and B are two nodes) based on A’s previous observation about B. In this action, A asked B to forward N-number of packets and B in fact forwarded K-number of packets. Let V (i) be the performance action of the B at the ith trial. That is, if V(i) = 1, B correctly performs the action at the ith trial; Otherwise V(i) = 0. n (N) =

$$\sum_{i=1}^{N} V_i$$ -> Number of actions successfully performed

by B out of totally N trials. For the N trials of transmission between two nodes, K trials are success. The probability of successfulness of (N+1)th trial will be predicted by Bayesian Theorem given below

$$P(v(N+1)=1/n(N)=k=\frac{P(v(N+1)=1,n(N)=k)}{P(n(N)=k)}$$

Here the probability of all trials will be calculated by the Bernoulli's distribution given below

$$P\{n(N)=K\}=\binom{N}{K}p^{K}(1-p)^{(N-K)}$$

where,
P = Average Probability of success transmission of each packets
N = Total number of packets transmitted by source
K = Number of packets successfully transmitted by neighborhood nodes

Secure route establishment using antnet: When a node (source) wants to establish a route to the other node (destination), the source first tries to find multiple routes to the destination. Then the source tries to find the packet-forwarding trustworthiness of the nodes on the routes from its own trust record or through requesting recommendations. Finally the source selects the trustworthy route to transmit data. After the transmission, the source node updates the trust records based on its observation of route quality. Using Antnet algorithm the following sequence of steps leads to discovery of route:

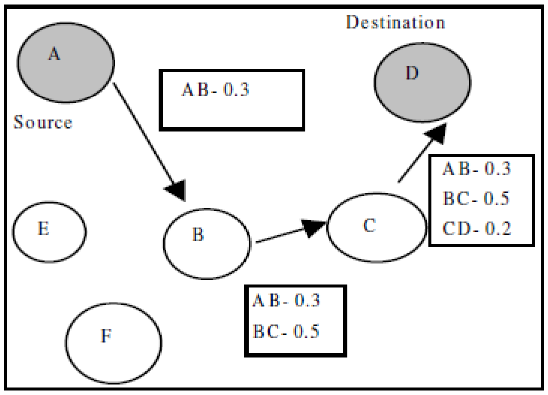

Fig. 2 Forward route discovery

• Each source launches some forward agent packets to destination through multi hop propagation. The path will be selected randomly based on the current routing table

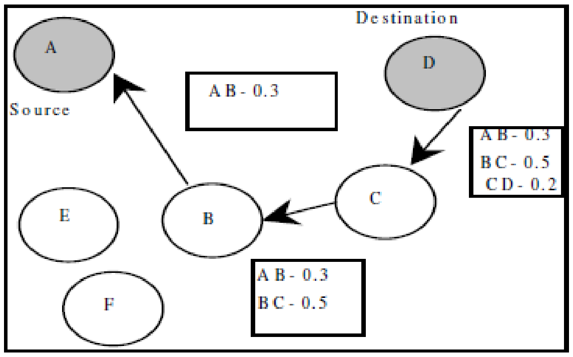

Fig. 3 Backward route discovery

• The forward agent packets create a stack, pushing in trip times, trust values and traffic intensities of every node it visits during transmission as shown in Fig. 2.

• When the packets reach the destination, some backward agent packets will be sent to the source. During this time, the backward agent packets inherit the stacks parameters. That is pop the parameter and verify once again as shown in Fig. 3.

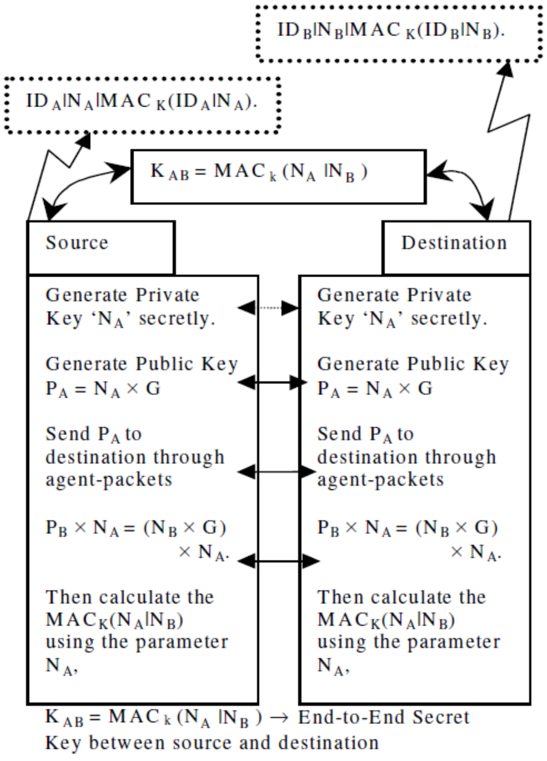
$ID_B|N_B|MAC_K(ID_B|N_B)$.
$ID_A|N_A|MAC_K(ID_A|N_A)$.
$K_{AB}=MAC_k(N_A|N_B)$
Source
Destination
Generate Private Key ‘N_A’ secretly.
Generate Public Key $P_A=N_A\times G$
Send P_A to destination through agent-packets
$P_B\times N_A=(N_B\times G)\times N_A$.
Then calculate the $MAC_K(N_A|N_B)$ using the parameter N_A.
Generate Private Key ‘N_A’ secretly.
Generate Public Key $P_A=N_A\times G$
Send P_A to destination through agent-packets
$P_B\times N_A=(N_B\times G)\times N_A$.
Then calculate the $MAC_K(N_A|N_B)$ using the parameter N_A.
$K_{AB}=MAC_k(N_A|N_B)\rightarrow$ End-to-End Secret Key between source and destination

Fig. 4 Session secret key negotiation Algorithm

G - Point on Elliptic Curve whose order is ‘n’
N - Total number of points in Elliptic Curve including point on infinity
NA, NB - Secret Keys
PA, PB - Public Keys
The backward agent packets deliver the parameters of trust values, traffic intensities and delays of discovered routes to

source. Finally the source will select the optimum path to destination. The secret-key exchange between the source and destination followed by ECC is shown in Fig. 4

8. Malicious Nodes Detection

Wireless Sensor nodes in sensor networks are usually deployed in hostile environments such as battlefields. Consequently a sensor node may be compromised or out of function and then provides wrong information that may mislead the whole network. This problem is called as the Byzantine problem. For example, a compromised sensor node (malicious node) can constantly report incorrect information to higher layers. The aggregator (FN or AP) in higher layer may make a wrong aggregation result due to the effect of the malicious node. It is therefore an important issue in sensor networks to detect malicious nodes in spite of such Byzantine problem.

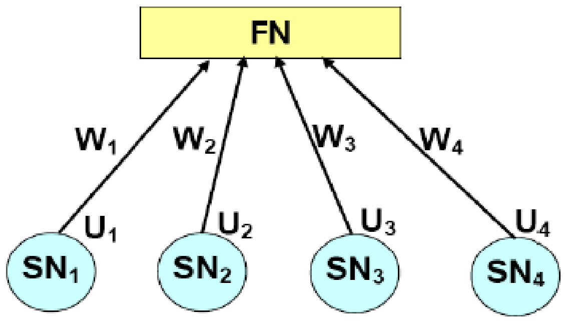

Fig. 5 A weight based network for hierarchical sensor network

As the first step toward the solution to the problem, we model it into a weight-based network as shown in Figure 5. The network is adapted in the architecture between a group of sensor nodes and their forwarding node. As shown in the figure, a weight W is assigned to each sensor node. The FN collects all information provided by SNs and calculates an aggregation result using the weight assigned to each SN:

$$E = \sum_{n=1}^{N} W_n \times U_n$$

Where E is the aggregation result and Wn is the weight ranging from 0 to 1. An essential concern is about the definition of sensor node's output Un. In practice, the output information Un may be "false" or "true" information or continues numbers such as temperature reading. Thus the definition of output Un is usually depending on the application where the sensor network is used. The following issue is to update the weight of each sensor node based on the correctness of information reported. Updating the weight of each sensor node has two purposes. First, if a sensor node is compromised (becomes a malicious node) and frequently sends its report inconsistent with the final decision, its weight is likely to be decreased. Then if a sensor node's weight is lower than a specific threshold, we can identify it as a malicious node. Second, the weight also decides how much a report may contribute to the final decision. This is reasonable since if the report from a sensor node tends to be incorrect, it should be counted less in the final decision. This logic is reflected in the following equation.

$$W_n = \begin{cases} W_n - \theta \times r_n & if (U_n \neq E) \\ W_n & elsewise \end{cases}$$

Where θ is a weighted penalty ratio. When the output of a sensor node s is not consistent with the final result, its weight is reduced by the weight penalty θ multiplying rn. The number rn is defined as:

$$r_n = m/s$$

Where m is the number of nodes in the cluster sending different report to the FN, and s is the total number of nodes in the cluster under the same FN. An optimal θ value is essential in our WTE mechanism since it affects the detection time and the accuracy of the algorithm. In addition, due to various definitions of output information (Un) as mentioned above, the consistence determination, which decides whether a node's output is consistent with the final result, is also application dependent. For example, it is easy to determine the consistence for a "false" or "true" output. However, for a continuous number of Un like temperature reading, the probability distribution function could be used to determine the consistency of the output information from all sensor nodes. Furthermore, a normalization operation as described in the following equation is used to guarantee the weight kept in the range from 0 to 1.

$$W_n = W_n / \max(W_{1,} \cdots W_N)$$

Based on updated weights, the forwarding node is able to detect a node as a malicious node if its weight is lower than a specific threshold. This detection algorithm can be widely used in different types of sensor networks. For example, the number of sensor nodes can vary in the algorithm, which makes it suitable for very large and very small networks. However, the description of sensor node output and updating scaling factor which are dependent on the applied application need to be determined carefully in order to achieve efficient and high accuracy detection.

9. Conclusions

Security in wireless sensor networks is a field of research that is growing rapidly and achieving tangible results applicable to real-life scenarios. Nevertheless, there is still room for more improvements in this area. Fields such as public key cryptography and intrusion detection systems over sensor networks are fairly new. It is necessary to develop secure routing algorithms while complying with essential design properties, such as connectivity, coverage and fault tolerance. Also, secure data aggregation algorithms should be more optimal, and the privacy of the information now should be taken into account. Other open areas of research include tolerating the lack of physical security, optimizing the security infrastructures in terms of resources (energy and computation), detecting and reacting over denial of service attacks, and raising the question of the social privacy problems that sensor networks might create. Finally there are some areas, such as the management and protection of mobile nodes and base stations, and the secure administration of multiple base stations with delegation of privileges that are yet developed.

10. References

[1] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy- Efficient Communication Protocol for Wireless Microsensor Networks," IEEE Proceedings of the 33rd Hawaii International Conference on System Sciences, Jan. 2000, pp. 1–10.

[2] M. Tubaishat, J Yin, B. Panja, S. madria, "A Secure Hierarchical Model for Sensor Networks", ACM SIGMOD, Volume 33 , Issue 1 March 2004, ACM Press, NY

[3] I. Khalil, S. Baghi, and C. Rotaru, "DICAS: Detection, Diagnosis and Isolation of Control Attacks in Sensor Networks", IEEE Securecomm 2005, 5-9 September 2005, Athens Greece.

[4] C.Y. Chong, and S.P. Kumar, "Sensor Networks: Evolution, Opportunities, and Challenges", IEEE 2003

[5] S. Capkun and JP Hubaux, "Secure Positiong of Wireless Devices with Application to Sensor Networks", In the proceedings of IEEE INFOCOM 2005, Miami, FL, USA

[6] A. Savvides, C.C. Han, and M.B. Strivastava, "Dynamic Fine-Grained Localization in Ad-Hoc Networks of Sensor", In the proceedings of ACM SIGCOM, July 2001, Rome, Italy

[7] C. Karlof and D. Wagner, "Secure Routing in Wireless Sensor Networks: Attacks and Countermeasures", University of California at Berkeley, USA 2003

[8] N. Sastry, U. Shankar, D. Wagner, "Secure Verification of Location Claims", proceedings of the 2003 ACM workshop on Wireless Security, 2003. pp. 1-10.

[9] L. Lazos and R. Poovendran, "SeRLoc: Secure range-Indpendent Localization for Wireless Sensor Networks", In the proceedings of ACM WiSe'04, October 1, 2004, Philadelphia, Pennsylvania, USA.

[10] F. Anjum, S. Pandey, and P. Agrawal, "Secure Localization in Sensor Networks using Transmission Range Variation", In the proceedings of IEEE MASS 2005 Workshop, November 7-11, 2005, Washington DC, USA.

[11] T. A. Zia, and A. Y. Zomaya, "A Secure Triple-Key Management Scheme for Wireless Sensor Networks", In the proceedings of the IEEE INFOCOM 2006 Students Workshop, April 23-24, 2006, Barcelona, Spain

[12] K. A. Bradley, S. Cheung, N. Puketza, B. Mukherjee, and R. A. Olsson. Detecting disruptive routers: A distributed network monitoring approach. In *IEEE Symposium on Security and Privacy*, pages 115{124, 1998.

[13] S. Cheung. An e_cient message authentication scheme for link state routing. In *13th Annual Computer Security Applications Conference*, pages 90{98, 1997.

[14] T. H. Cormen, C. E. Leiserson, and R. L. Rivest. *Introduction to Algorithms*. MIT Press, Cambridge, MA, 1990.

[15] R. Hauser, T. Przygienda, and G. Tsudik. Reducing the cost of security in link-state routing. *Computer Networks and ISDN Systems*, 1999.

[16] C. Kaufman, R. Perlman, and M. Speciner. *Network Security: Private Communication in a Public World*. Prentice-Hall, Englewood Cli_s, NJ, 1995.

[17] S. Murphy, M. Badger, and B. Wellington. RFC 2154: OSPF with digital signatures, June 1997. Status: EXPERIMENTAL.

[18] S. L. Murphy and M. R. Badger. Digital signature protection of OSPF routing protocol. In *Proceedings of the 1996 Internet Society Symposium on Network and Distributed System Security*, pages 93{102, 1996.

[19] R. Perlman. *Network Layer Protocol with Byzantine Agreement*. PhD thesis, The MIT Press, Oct. 1988. LCS TR-429.

[20] R. Perlman. *Interconnections, Second Edition: Bridges, Routers, Switches, and Internetworking Protocols*. Addison-Wesley, Reading, MA, USA, 2000.

[21] B. Schneier. *Applied cryptography: protocols, algorithms, and sourcecode in C*. John Wiley and Sons, Inc., New York, 1994.

[22] B. R. Smith, S. Murthy, and J. Garcia-Luna-Aceves. Securing distance-vector routing protocols. In *Symposium on Network and Distributed Systems Security (NDSS '97)*, 1997.

[23] B. Vetter, F.-Y. Wang, and S. F. Wu. An experimental study of insider attacks for the OSPF routing protocol. In *5th IEEE International Conference on Network Protocols*, 1997.

[24] S. F. Wu, F.-Y. Wang, Y. F. Jou, and F. Gong. Intrusion detection for link-state routing protocols. In *IEEE Symposium on Security and Privacy*, 1997.

[25] K. Zhang. E_cient protocols for signing routing messages. In *Symposium on Network and Distributed Systems Security (NDSS '98)*, San Diego, California, 1998. Internet Society.

[26] B. Sun, K. Wu, and U. Pooch, "Secure Routing against Black-hole Attack in Mobile Ad Hoc Networks," in *Proceedings of Communications and Computer Networks*, 2002.

[27] M. Tubaishat and S. Madria., "Sensor Networks: an Overview," *IEEE Potentials*, 22, 2, 20-23, April 2003.

[28] M.A.M. Vieira, D.C. da Silva Jr., C.N. Coelho Jr., and J.M. da Mata., "Survey on Wireless Sensor Network Devices," *Emerging Technologies and Factory Automation* (ETFA03), September 2003.

[29] W. Ye, F. Silva, and J. Heidemann, "Ultra-Low Duty Cycle MAC with Scheduled Channel Polling", in *Proceedings of the 4th ACM Conference on Embedded Networked Sensor Systems* (SenSys), Boulder, Colorado, USA, November, 2006.

[30] Y. Yu, B. Krishnamachari, and V.K. Prasanna, "Energy-Latency Tradeoffs for Data Gathering in Wireless Sensor Networks," *IEEE Infocom'04*

[31] S. Zhao, K. Tepe, I. Seskar and D. Raychaudhuri, "Routing Protocols for Self-Organizing Hierarchical Ad-Hoc Wireless Networks," *Proceedings of the IEEE Sarnoff Symposium*, Trenton, NJ, March 2003.

[32] L. Zhou and Z. Haas, "Securing Ad Hoc Networks," *IEEE Network Special Issue on Network Security*, 13, 6, 24-30, November 1999.

[33] S. Zhu, S. Setia, and S. Jajodia, "LEAP: Efficient Security Mechanisms for Large-Scale Distributed Sensor Networks," *CCS'03*,October2003.