Securing Wireless Sensor Networks against Broadcast Service Attacks

Seonho Choi
Department of Computer Science, Bowie State University
Bowie, MD 20715, U.S.A.
and
Hyeonsang Eom
School of Computer Science and Engineering, Seoul National University
Seoul, 151-744, Korea

Abstract— **Attacks against broadcast services in wireless sensor networks will have the most catastrophic effects on power and other resource consumptions. As bogus broadcast packets are propagated through sensor nodes without being filtered out, sensor nodes will waste their resources such as power and memory on transmitting and buffering those false packets. In networks where complex Denial-of- Service (DoS) attacks may be launched, each node should be able to limit its effects on resource consumptions. In addition, it is strongly desired to identify compromised sensor node(s) which has injected invalid packets (intrusion detection) and to isolate the nodes (remediation) such that no more DoS attack is possible through the compromised sensor node(s) in the future. A new secure broadcast authentication scheme for sensor networks is proposed in this paper based upon predictive -hasing and µTESLA techniques. This new scheme may be used to detect invalid packets as quickly as possible and to isolate links that are from the compromised nodes, which results in enhanced resistance to various security attacks including DoS attacks. We present the details of the scheme along with the simulation results.**

Index Terms— **authentication, wireless sensor network, broadcast, intrusion detection, remediation**

I. Introduction and Related Works

Broadcast (or multicast) plays an important role in wireless sensor networks though its invocation frequency may vary from application to application. For example, several routing protocols were proposed based on periodic broadcasting (e.g., flooding) of routing (or beacon) messages. These include TinyOS beaconing [3], directed diffusion and its multi-path variants [4], etc. Also, several location discovery schemes were proposed, which utilize broadcasting capabilities to estimate node locations [5]. Even though more advanced broadcast techniques may be utilized in the network, in many cases in sensor networks, simple flooding is preferred or required due to the simplicity or instability of network connections. Our proposed approach may be applied in both cases.

Most of the wireless sensor network protocols are susceptible to various types of attacks such as spoofing, altering, replaying, sinkhole attacks, Sybil attacks, Wormholes, HELLO flood attacks, and Acknowledgement spoofing, etc. [3]. Especially, broadcasting is a very attractive service for an adversary to consider for launching various types of attacks due to the significant amount of bandwidth and power consumption involved in broadcasting. If an adversary succeeds in injecting invalid broadcast packets into the network, all the nodes in the network need to forward them unless there is a way for immediate authentication of the broadcast packet.

However, authenticating broadcast (or multicast) traffic in resource constrained environment such as sensor networks is a hard problem since the traditional approaches like digital signatures may not be adequate due to the heavy resource requirements. µTESLA approach [6] was proposed as a viable solution to the authentication problem in such networks. µTESLA utilizes delayed key disclosure and one-way key chain technique. However, µTESLA is susceptible to Denial-of-Service (DoS) attacks due to delayed authentication. If an adversary succeeds in spoofing/injecting one invalid broadcast packet into the network, e.g., before the key is disclosed in µTESLA, all the nodes in the network need to forward the invalid packet since it may not be authenticated until the corresponding authentic key is disclosed later. In this way, DoS attacks may be launched against broadcast service even in the network with µTESLA protocol running unless there is a mechanism that enables each node to *immediately* decide whether (at least significant portion of) the arriving packet is authentic or not.

Such DoS attacks against broadcasting capability in sensor network will have the most catastrophic effects on power consumption in sensor nodes. As bogus broadcast packets are propagated through sensor nodes without being filtered out, sensor nodes will waste their power on transmitting those invalid packets. In case there is such a DoS attack launched against the network, the network should be able to limit its effects on resource consumptions in the network nodes. In addition to that, it is strongly required to identify the compromised sensor node which has injected invalid packets (***intrusion detection***) and to isolate the nodes (***remediation***) such that no more DoS attack is possible through the compromised sensor node(s) in the future.

Multi-level µTESLA [7] was proposed as a solution to provide a DoS resistance. However, this approach has several limitations. First, an adversary may inject false broadcast packets by just modifying the last MAC field value in each

This work was supported by the U.S. Army Research Office (ARO) under Grant 50849CIISP.

packet after obtaining the authentic message contents and the MAC field values via Wormhole attack. In this scenario, all the nodes still need to forward different broadcast packets (authentic plus false) through the network, which will result in the same degree of resource waste compared to the cases where original TESLA is used. Second, due to the increased number of MAC fields in each packet, bandwidth and power overhead will increased. Third, the buffer space needed in each node may not be reduced in the cases mentioned above.

A new secure broadcast scheme for sensor networks is proposed in this paper with the following features:

a) Broadcast packet authentication
b) Prevention of spoofing attacks from arbitrary nodes
c) Resistance to various types of DoS attacks
d) Identification/Isolation of the compromised node(s)
e) Minimal usage of buffer space in each node

Section II presents the background work. The proposed solution approach is given in Section III. Simulation design / results are explained in Section IV, and the conclusion follows in Section V.

II. Background

A protocol named µTESLA [6] has been proposed for broadcast authentication in distributed sensor networks, which is adapted from a stream authentication protocol called TESLA. µTESLA employs a chain of authentication keys linked to each other by a pseudo random function, which is by definition a one way function. Each key in the key chain is the image of the next key under the pseudo random function. µTESLA achieves broadcast authentication through delayed disclosure of authentication keys in the key chain. The efficiency of µTESLA is based on the fact that only pseudo random function and secret key based cryptographic operations are needed to authenticate a broadcast message. Time line is divided into intervals and different keys are assigned to the intervals from the one-way key chain(OKC), and they will be used to generate MACs from the messages. However, a key used to generate a MAC from a message in an interval will be disclosed after a fixed number, *d*, of intervals in such a way that the probability of message delivery to most of the recipients in the network before that delay is high. This scheme is shown in Figure 1.

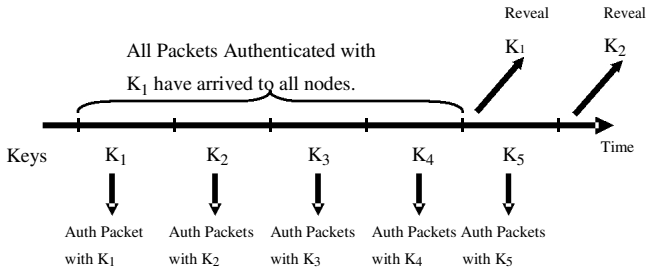

Fig. 1. µTESLA [8] approach for authenticating broadcast packets in sensor networks. Note that the key disclosure delay is set to 4 TESLA intervals. That is, *d=4*.

However, µTESLA approach is not resistant to some type of security attacks such as DoS attacks due to the authentication delay. With a simple flooding attack, each node will suffer from buffer overflows and the communication bandwidth will be wasted in the network during the delay.

Predictive Hashing

To enhance the resistance to security attacks including DoS attacks, a new approach named as *Predictive Hashing* was proposed [2] for stream authentication in multicasting environment. In this approach, instead of delaying disclosure of only the key portion as in TESLA, both message and key portions are disclosed after a fixed delay (in one of the future block period). The MAC portion which is disclosed without any delay was named as Predictive Hash (PH). In this scheme, message portion is immediately authenticated upon the receipt and doesn't have to be buffered. Since only PH value needs to be buffered at each node in the network, buffering overhead is greatly reduced even in the presence of DoS attacks. The idea of this scheme is shown in Figure 2.

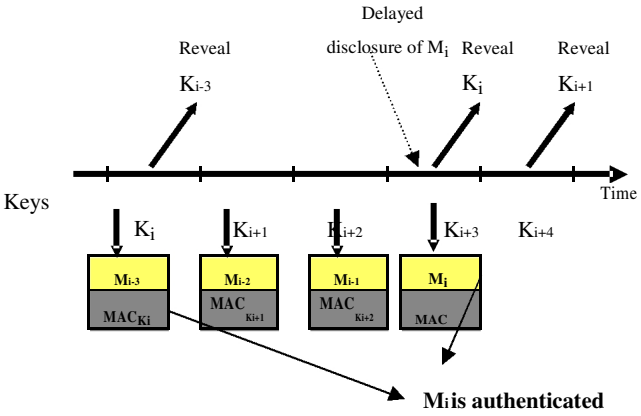

Fig. 2. Predictive Hashing technique [2] is shown with *d=3*. Note that not only the key is disclosed later, but also the corresponding message portion is disclosed later along with the key.

III. Proposed Approach

Our proposed approach consists of two components: Lightweight Neighbor Authentication Protocol (LNAP) and Predictive Hash Based Broadcast Protocol (PHBBP).

A. *Lightweight Neighbor Authentication Protocol (LNAP)*

LNAP is used for authenticating whether the received packet is really from the neighbor whose source identity is claimed inside the packet. The approach presented in [1] for mobile ad-hoc network is adapted here for sensor network application. LNAP protocol allows each sensor node to authenticate whether a received packet is really from its neighbor. In other words, one-hop authentication may be carried out with this protocol. Due to the space limitation, the details are not given here, but can be found in [1].

LNAP protocol is mainly useful for preventing simple spoofing attacks, and, for more complex attacks with compromised nodes in adversary's hand, this protocol may not be effective.

B. *Predictive Hashing Based Broadcast Protocol (PHBBP)*

For further protection against complex attacks and for isolation/remediation of compromised nodes, PHBBP protocol is proposed. This protocol is based on Predictive Hashing (PH) technique and Size-Selective Forwarding (SSF) technique. PH technique was explained in the previous section, and it is assumed that a one-way key chain (OKC) is obtained and the last key of the chain is pre-installed to all sensor nodes in the network. If OKC keys are exhausted, another OKC may be obtained by the broadcast source (such as BS) and distributed by using the broadcast services. The OKC keys used in our protocol is represented as K_i for $i=1, 2, \ldots$. To limit the forwarding level at each node, thus to put a upper bound on resource waste, forwarding limit will be introduced and denoted as *f_level*. This means a node may forward up to *f_level* broadcast packets (either full-length or shortened) in one time interval.

Packet types:

(1) full-length packet:

A full-length broadcast packet, P_{i+d}, which is generated in the i-th TESLA interval and sent out in the $(i+d)$-th interval is assumed to consist of the following fields:

a) Message Portion (M_i) – contains application data produced at the source (BS) in an i-th TESLA time interval. This part was buffered at the BS until the $(i+d)$-th time interval.

b) Key Portion (K_i) – contains a TESLA key that was used to obtain PH_i. PH_i was included and sent in P_i. Upon the receipt of P_{i+d}, the receiver verifies the validity of K_i using the one-way key chain knowledge. If the key is verified to be authentic, the node will apply a MAC operation to M_i with K_i as its key, and compare the result to PH_i that was already delivered and buffered at the node.

c) Predictive Hash (PH_{i+d}) – contains the result from the MAC operation applied to M_{i+d} with a key K_{i+d}. M_{i+d} and K_{i+d} will be sent out to the receivers in the $(i+2d)$-th TESLA time interval.

(2) shortened packet:

A shortened broadcast packet, P_{i+d}, is assumed to consist of the following fields:

a) Time interval index $i+d$ denoting the corresponding TESLA time interval.

b) Predictive Hash (PH_{i+d}) – contains the result from the MAC operation applied to M_{i+d} with a key K_{i+d}.

Protocol Outline:

The basic idea is to infuse Predictive Hashing technique into µ-TESLA approach to provide authentication at each sensor node. The following are the key features of this new protocol:

(i) Only the MAC portion is pre-distributed by using predictive hashing technique while the message and key portions are distributed in a later block period.

(ii) Each receiver can authenticate message portions immediately upon their receipt by utilizing PHs stored in PH buffer array.

(iii) Only the PHs and hop_count information need to be stored into PH buffer while message portions may be discarded after applying authentication operation.

(iv) When a node finds out that a message portion is already forwarded previously, it formulates a shortened packet instead of a regular packet for forwarding different PHs.

(v) When a node detects any invalid message, or key, or PH received along any of its links, then it blocks them either immediately or with some delay (hop count based blocking)

Assumptions:

- Time Synchronization: times in sensor nodes are loosely synchronized with an error bound. Any technique may be used for this purpose such as those in [8, 9] with a reasonable overhead.
- At each node, hop_count field in the regular or shortened packet will be incremented with modulo-HC_{max} where HC_{max} is the maximum value for this field.

Attack Types:

Even though various attack types may be considered combining different attack mechanisms, we consider the following attack model because they may cause the most catastrophic results in terms of resource waste against our protocol (LNAP+ PHBBP) . The simulation results against these attacks are presented in the next section to show the relative strengths of our approach compared to the traditional approach.

Simple Spoofing Attack: an attacker will compromise one or more sensor nodes and inject randomly generated packets through the compromised nodes to the network. At least one of M_i, K_i, or PH_{i+d} doesn't contain valid contents, and they are named as message portion modification attack (MMA), key field modification attack (KMA), and PH-field modification attack (PMA).

By utilizing the simple spoofing attack along with other techniques such as physical relocation etc., attacker may launch more complicated attacks such as:

Neighbor Duplication Attack: an attacker compromises one node and finds out all the keying information used by LNAP protocol. And, possibly by placing duplicated nodes into random locations in the network, an attacker will be able to create compromised neighbor nodes with the same identity in different neighborhoods in the network. This attack may be combined with the Sybil attack [10] to increase the number of false broadcast packets injected to the network. Our protocol has the capability to deal with this type of attacks.

Wormhole Tunneling Attack (WTA): an attacker needs to have compromised nodes in at least two separate regions in the network. Node(s) closer to the BS captures the broadcast packet contents and tunnels them to the node(s) located farther from BS via high-speed link for wormhole attack. The farther nodes will re-broadcast the packets as if they have followed valid paths from the BS. This attack may be used to hinder the network management operations such as routing tree discovery protocols. Against our protocol, the predictive hash, PH_{i+d}, contained in P_{i+d} may be altered by an adversary while all the other fields, i.e., M_i and K_i, are not modified. If a node receives P_{i+d}, and if the packet is from this type of attack, then

it would not be able to detect the attack until the corresponding data (M_{i+d}) and key (K_{i+d}) are received in the future packet P_{i+2d}. Our protocol is able to detect this type of attack and isolate the compromised nodes with a delay.

Protocol Description

We may classify the possible scenarios in our protocol into the following cases depending upon whether an arriving packet is an attack packet or not, and upon whether the node has a valid PH value for its message portion.

Case 1: the arriving packet is of regular packet type

(i) node has a valid PH (maybe along with invalid PHs) and the arriving packet is authentic: this is the most ideal case and the key and message portions in the arriving packet will be authenticated and a new PH value will be stored into the PH buffer. However, if there are additional invalid PHs in the PH buffer, it implies that there were PH-modification attacks already launched in one of the previous block periods. In this case, the node will identify the links from which these invalid PHs were received and mark them as P_CMPR (potentially-compromised). Since the node also recorded a hop_count field from the received packet when it recorded the invalid PH, the node knows how many hops away the invalid packet was injected, and it will initiate hop-count based blocking process, which will be explained later in this section.

(ii) key field in the packet is changed: the invalidity of the key will be immediately detected, a packet will be discarded, and a link from which this new packet has arrived will be marked as "CMPR" so that any future packets from this link will be discarded. This case is detected by the OKC key verification function.

(iii) message authentication fails: In this case, the key field authentication succeeds while the message authentication fails. This case may occur either when an authentic PH was lost in the previous block period and a corresponding message portion arrives later, or when the correct PH is in the PH buffer, but the message portion of

(iv) the corresponding packet was altered. In either case, when the node finds out that the PH value from the arrived packet doesn't exist in PH buffer, then it will first record it along with hop_count into PH buffer. The approach adopted here is to formulate a shortened packet and locally broadcast it.

(iv) PH-modification attack packet is received: if the received packet's PH value has been compromised, then the receiver will record the PH value into its PH buffer along with hop count value anyway since it doesn't have any knowledge on the validity of the PH until the actual message authentication succeeds. But, in a case where more than one packet are received with a valid key & message portion (thus first packet message is already authenticated), then one or more of these packets must be from the PH-modification attack. To reduce bandwidth usage, we adopted to formulate shortened packets and forward them. However, to find out whether the second or later packets have valid key and message portions, it may be needed to store the authentic message in each block period. Checking the validity of the key is simple since the last-validated key is kept in a node anyway. But, message portion validation may not be really needed if the key is validated. This is because, if an attacker was able to eavesdrop the valid key from an authentic packet, then it might be as easy to eavesdrop the message portion, too.

(v) packet's arrival time is out-of-bound: if the packet may arrive too early (in terms of the security condition), then it will be simply discarded.

Case 2: the arriving packet is of shortened packet type

(i) if an arriving packet is a shortened packet: the receiver will store its PH value and forward it as long as the number of forwarded packets in the block period (to which the received packet belongs) hasn't exceeded *f_level*.

Remediation: Isolation of the compromised nodes

For MMA and KMA attacks, the links from which the packets are received containing invalid keys or messages may be identified immediately after they are received. Once these links are identified, they will be marked as "CMPR" (Compromised), and future packets received along them will be discarded without any further processing.

However, for PMA attacks, the difficulty of isolation arises from the fact that there exist delays from the time PH values are distributed until the time they are validated/invalidated. Due to this delay, it is inevitable that *f_level*d* number of distinct PHs may be disseminated through the network before the detection/isolation of the compromised nodes (links) can be done, where *f_level* is a maximum number of different PHs that may be disseminated in one block period – thus defines maximum level of DoS attacks, and *d* represents the fixed key-disclosure delay used in PHBBP. To identify and isolate immediate links from the compromised node, we devised a solution approach named as "hop count based blocking". In this approach, when the node detects receipt of (first) invalid PH along a link, it will mark it as "P_CMPR" (Potentially Compromised) instead of immediately blocking them. Then, hop count information in the broadcast packet (either from regular packet or shortened packet) will be used to determine how long each node needs to wait until they block the links.

Hop count based blocking: *A node will block the link set as P_CMPR with hop_count when it detects an invalid PH even after (hop_count+1)×d block periods since the time it was set as P_CMPR.*

In this way, direct neighbors (which must have hop_count=0) of the compromised node will immediately block the links upon detecting additional invalid PH after *2d* block period after the first one was detected while more-than-one-hop away neighbors will never block any links because they will not

receive any invalid PHs from the compromised node after *(hop_count+1)×d* periods due to blocking of the links near the compromised node.

However, if an attacker changes the hop_count value with malicious intent, it may increase the delays until the uncompromised nodes block the compromised links. We will call this type of attacks as "hop count attack" (HCA). To limit the effect of this attack, we will assume that there is a maximum limit on the value hop_count field may have. Let's denote this as "HC_{max}". When a node receives a packet with hop_count= HC_{max}, then it will wrap it around to 0 (before forwarding it) and the next-hop nodes will immediately block the links upon the detection of first invalid PH. As long as at least one invalid PH in *d* block period is delivered to any direct neighbor of the compromised node, this scheme is guaranteed to isolate and cut off all the links from the compromised node. However, in cases where this guarantee on packet losses can't be made, some of the links that are not directly from the compromised node may also be cut off.

IV. Simulation

Our protocol was implemented on TinyOS platform, and extensive simulations were carried using TOSSIM simulator. The following are input parameters for each simulation session:

a) packet size: 30 bytes for regular packet and 13 bytes for short packet (PH size is 8bytes)
b) block period: 2 seconds
c) *d*: key disclosure delay is set to 3 block periods (6 seconds)
d) simulation time: 500 seconds
e) number of nodes in the network: 36 and 64
f) link model: TOSSIM empirical lossy link model (link loss probabilities were obtained with a distance factor of 5 feet by using the LossyBuilder program included in TOSSIM package)
g) network topology: grid+random
h) HCA parameter: in one of the simulation settings, a HCA was launched with an invalid initial hop_count value of 8.

We used the default MAC protocol provided by TOSSIM, which is CSMA protocol. In CSMA, when a node has a pending packet to be sent out, delay arises when it repeatedly enters CSMA wait because they continue to hear a signal on the channel. And, when this delay is large, it is likely that additional broadcast packets may be received by nodes even before the current packet is transmitted. However, for the purpose of simplicity and reducing buffer requirements, no queueing mechanism is implemented in our implementation, and the additional packets will be simply discarded if there is already another packet waiting to be broadcast. Hence, as the network traffic increases, more packets will be lost during broadcasting process and receivers will suffer from lowered delivery rates.

In our simulation scenarios, due to the fact that the attack streams are injected along with the authentic stream, it is very likely that the link will be busy most of the times and the node will enter CSMA wait state with a high probability. Hence, one obvious metric we may use for protocol performance is the packet delivery rate at the receivers. For example, out of N broadcast packets, we may measure how many of them are correctly received by the receivers.

We adopted to implement 2 different attack types, one for MMA and another for PMA. MMA with 5 compromised nodes in 64-node network was launched to compare the performances between PHBBP and µTESLA protocol. The 5 left-upper corner region nodes are compromised and they launched MMA attacks against the rest of the network nodes, while the authentic packets were generated and broadcast by the last node (node 63 in the right-lower corner). Figure 4 shows authentic packet loss rates experienced by the nodes. The nodes in the network with µTESLA protocol experiences more than 6 times of loss rates on average compared to those running PHBBP protocol. However, the power consumptions by CPU and radio units in both cases are almost at the same level. This is because the network (or link) utilizations in both networks are high, and the actual numbers of transmissions carried out by the nodes are almost the same.

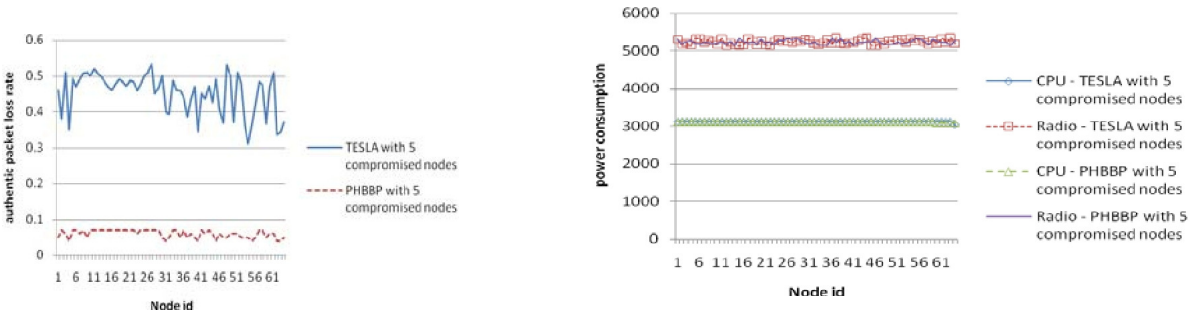

Figure 4: Comparison between µTESLA and PHBBP with 5 compromised nodes. (a) authentic packet loss rates experienced by each node (b) power consumption from CPU and radio communication

.

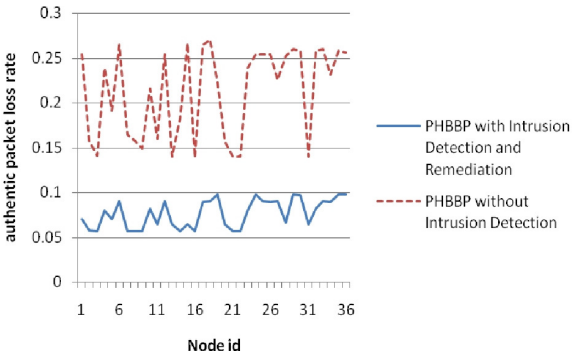

Figure 5: Authentic packet loss rates for PHBBP with intrusion detection / remediation capabilities and PHBBP without such capabilities.

And, PMA attack was launched for comparison purposes between PHBBP with intrusion detection / remediation capabilities and PHBBP without such capabilities. In this case, 36-node network was used and node 0 was an authentic source node while node 14 was a compromised node which launched wormhole-attack style PMA attack. Also, the attacker node maliciously set the hop_count value to be 8 instead of 0 to launch HCA at the same time. Figure 5 shows the reduced authentic packet loss rates for PHBBP protocol with intrusion detection / remediation capabilities. Network without such capability suffered about 3 times of the loss rates on average. Similar to the previous case, power consumptions in both networks were almost at the same level.

V. Conclusion

We proposed a new broadcast protocol for sensor networks which are more resilient to various security attacks against broadcast services. Our protocol also has the capability to detect / isolate compromised nodes in the network. We developed a simulator and presented experimental results which show that our protocol outperforms traditional approach.

References

[1] Sencun Zhu, Shouhuai Xu, Sanjeev Setia1, and Sushil Jajodia. LHAP: A Lightweight Hop-by-Hop Authentication Protocol For Ad-Hoc Networks, 23rd International Conference on Distributed Computing Systems Workshops (ICDCSW'03) , p. 749.

[2] Seonho Choi. Denial-of-Service Resistant Multicast Authentication Protocol with Prediction Hashing and One-way Key Chain, 2005 IEEE International Workshop on Security and Pervasive Multimedia Environments (MultiSec 2005). pp. 701-706

[3] Chris Karlof and David Wagner, Secure Routing in Wireless Sensor Networks: Attacks and Countermeasures, In Proceedings of the First IEEE International Workshop on Sensor Network Protocols and Applications, May 2003.

[4] R. di Pietro, L. V. Mancini, Y. W. Law, S. Etalle and P. Havinga A directed diffusion-based secure multicast scheme for wirless sensor networks.First International Workshop on Wireless Security and Privacy (WiSPr'03)

[5] S. Park, A. Bhatia, and J.-H. Youn (USA) . Hop-Count based Location Discovery in Ad Hoc Sensor Networks. *In Proceeding* (424) Wireless Networks and Emerging Technologies - 2004

[6] Perrig, A., Canetti, R., Song, D., and Tygar, D. 2001. Efficient and secure source authentication for multicast. In Proceedings of Network and Distributed System Security Symposium.

[7] Donggang Liu, Peng Ning, Multi-Level u-TESLA: A Broadcast Authentication System for Distributed Sensor Networks, Submitted for journal publication. Also available as Technical Report, TR-2003-08, North Carolina State University, Department of Computer Science, March 2003.

[8] Suyoung Yoon, Chanchai Veerarittiphan, and Mihail L. Sichitiu, Tiny-Sync: Tight Time Synchronization for Wireless Sensor Networks, to appear in *ACM Transactions on Wireless Sensor Networks*, 2007.

[9] J. Greunen and J. Rabaey, Lightweight time synchronization for sensor networks, in Proceedings of the 2nd ACM international conference on Wireless sensor networks and applications, Pages: 11 - 19 , San Diego, CA, USA, 2003.

[10] J. Newsome, E. Shi, D. Song and A. Perrig, The Sybil Attack in Sensor Networks: Analysis and Defenses, In Proc, of IPSN 2004, Berkeley, Sybil CA, April 2004.