Defense against Lap-top Class Attacker in Wireless Sensor Network

Md. Abdul Hamid[1], Md. Mamun-Or-Rashid[2] and Choong Seon Hong[3]

[1,2,3] Department of Computer Engineering, Kyung Hee University
1 Seocheon, Giheung, Youngin, Gyeonggi 449-701 Korea
{hamid,mamun}@networking.khu.ac.kr and cshong@khu.ac.kr

***Abstract* — We consider Wireless Sensor Network (WSN) security and focus our attention to tolerate damage caused by an adversary who has compromised deployed sensor node to modify, block, or inject packets. We adopt a probabilistic secret sharing protocol where secrets shared between two sensor nodes are not exposed to any other nodes. Adapting to WSN characteristics, we incorporate these secrets to establish new pairwise key for node to node authentication and design multi-path routing to multiple base stations to defend against HELLO flood attacks. We then analytically show that our defense mechanisms against HELLO flood attack can tolerate damage caused by an intruder.**

***Keywords* — Routing Security, Multi-path Routing, Authentication, Tree Protocol, Key Sharing.**

1. Introduction

In a large-scale sensor network individual sensors are subject to security compromise. Where the nature of communication is broadcast and, hence, an attacker can overhear messages posted by any sensor node; security is an important issue here. Wireless Sensor Networks (WSNs) are comprised of many small and resource constrained sensor nodes that are deployed in an environment to gather sensed data and forward that data to interested legal users.

To support the reliability of coordinated control, management, and reporting functions, the sensor networks are self-organizing with both decentralized control and autonomous sensor behavior, resulting in a sophisticated processing capability [5]. There are several network layer attacks against sensor networks and are well described in [3]. Among them, spoofed, altered, or replayed routing information, selective forwarding, sinkhole attacks, Sybil attacks, wormholes, HELLO flood attacks, acknowledgement spoofing are well known attacks that try to manipulate sensed data. In this paper we consider routing security against HELLO flood attack.

As shown in figure 1, many protocols require nodes to broadcast HELLO packets to announce themselves to their neighbors, and a node receiving such a packet may assume that it is within (normal) radio range of the sender [3]. This assumption may be false: a laptop-class attacker broadcasting routing or other information with large enough transmission power could convince every node in the network that the adversary is its neighbor. The majority of the sensor nodes are stranded, sending packets into oblivion.

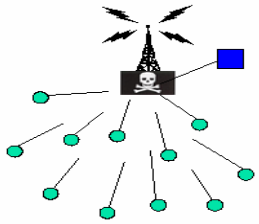

Figure 1. HELLO flood attack. A laptop- class adversary that can retransmit a routing update with enough power to be received by the entire network leaves many nodes stranded. They are out of normal radio range from the adversary but have chosen her as their parent.

There is high possibility for a mote class [3] attacker to create routing loops by spoofing routing updates. For example, if an attacker knows that node *x* and *y* are within radio range of each other, then the attacker can send a forged routing update to node *y* with a spoofed source address indicating it came from node *x*. Node *y* will then mark node *x* as its parent and rebroadcast the routing update. Node *x* will then hear the routing update from node *y* and mark *y* as its parent. Messages sent to either *x* or *y* will be forever forwarded in a loop between the two of them. We assume that a lap-top class attacker can cause a HELLO flood attack by advertising a very high quality route to the base station. So, every node in the network could cause a large number of nodes to attempt to use this route thereby sending the legitimate packets beyond the actual destination.

The main contributions of our work are as follows:

- We present probabilistic secret sharing protocol adopted from [1] where, a small increase in the number of secrets maintained by a user substantially reduces the probability of privacy compromise. And it is beneficial for the case where the sensor nodes do not have the capability to hold sufficient secret to ensure privacy. We show how these secrets can be further used to establish pairwise key and using this resulting key to implement an authenticated, encrypted link between them. To defend against HELLO flood attack, we show that it is possible for every node to authenticate each of its sender and receiver.
- We provide further defense mechanism by incorporating the concept of multi-path multi-base station routing to

This work was supported by MIC and ITRC Project.

improve the tolerance caused by an adversary who has highly sensitive receiver as well as powerful transmitter.

2. Key Sharing Protocol

In this section, we present the probabilistic protocol, the complementary tree protocol, for assigning the initial secrets. We will describe the single complementary tree protocol and then compute the multiple trees based key assignment. We organize the (Fig. 2) secrets in the tree of degree *d*. In this protocol, we require that *d>=3*. All nodes in the tree except the root are associated with a secret. Each leaf of the tree is associated with a sensor node. (Note that a leaf is associated with a sensor as well as a secret.) The secret distribution is as follows. For each level (except level 1), the node gets secrets associated with the siblings of its ancestors (including itself). Thus, node s_1 gets secrets k_2, k_3 (level 2), k_5, k_6 (level 3), k_{14} and k_{15} (level 4). A node does not get the secrets associated with its ancestors.

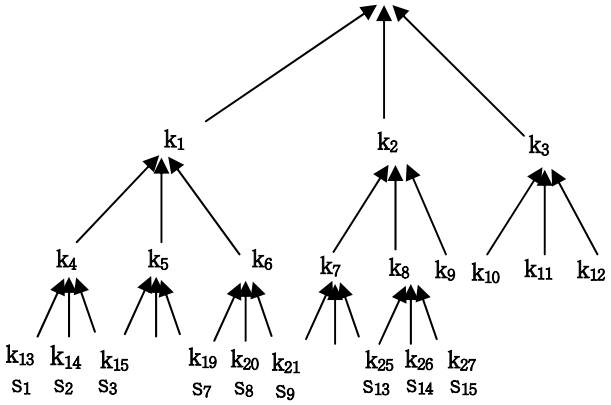

Figure 2. Single Complementary Tree Key Assignment

When two nodes, say *j* and *k*, want to communicate, they first identify their least common ancestor. Let *z* be the least common ancestor of *j* and *k*. Let *x* denote the child of *z* that is an ancestor of *j*. Likewise, let *y* denote the child of *z* that is an ancestor of *k*. Now, to communicate, *j* and *k* use the secrets associated with all children of *z* except *x* and *y*. For example, if s_1 and s_2 want to communicate, they use the secret k_{15}. If nodes s_1 and s_9 want to communicate then they will use the secret k_5. And, if s_1 and s_{15} want to communicate then they will use the secret k_3.

Next we compute the vulnerability of security. Let *x* be an intruder that can observe the communication between any two arbitrary nodes *y* and *z*. We calculate what is the probability that *x* knows the shared secret that *y* and *z* use. During this analysis, let the degree of the secret-tree be *d*.

Since no secrets are associated with the root, first consider the case where *y* and *z* use the secret(s) at level 2. Such a situation occurs if *z* is not a descendant of the level-2-ancestor of *y*. Thus, the probability of this case is *(d-1)/d*. And, in this case, the probability that *x* is aware of all the secrets is *2/d*; *x* knows all the secrets used by *y* and *z* iff *x* is a descendant of the level-2-ancestor of *y* or *x* is a descendant of the level-2-ancestor of *z*. Next, we consider the probability that *y* and *z* use the secret at level 3 in the tree. Such a situation arises if *z* is a descendent of the level-2-ancestor of *y* and *z* is not a descendent of the level-3-ancestor of *y*. Thus, the probability of this case is *1/d x(d-1)/d*. Moreover, *x* is aware of the shared secret(s) between *y* and *z* iff *x* is a descendant of the level-3-ancestor of *y* or *x* is a descendant of the level-3-ancestor of *z*. Thus, the probability of this case is *2/d x 1/d*.

Continuing thus, the probability of compromise, p_c, that *x* is aware of the shared secret used by *y* and *z* is:

$$p_c = \frac{d-1}{d}\frac{2}{d}\left(\sum_{i=0}^{l}(1/d)^{2i}\right)$$
$$= \frac{d-1}{d}\frac{2}{d}\left(\sum_{i=0}^{\infty}(1/d)^{2i}\right)$$
$$= \frac{d-1}{d}\frac{2}{d}\frac{1}{1-1/d^2}$$
$$= \frac{2}{d+1}$$

It is possible to reduce the probability of compromise in the complementary tree protocol even further if we maintain multiple trees. More specifically, if we maintain *K* trees where there is no correlation between node locations in different trees, the probability of security compromise will be $((2/(d+1))^K)$.

3. New Key Setup during Communication

According to the secret distribution protocol described earlier, we know that every pair of nodes shares some number of initial secrets *n*. Let's assume node *x* and *y* share initial secrets k_i ($i=1,2,....,n$). Prior to communicate with each other, these two nodes can generate a new key using *k* initial secrets from *n* secrets on demand. This can be done using some mathematical function e.g. RC5 [10] to generate MAC. So, we say that new key, $E_{new\text{-}key} = MAC(k_1, k_2, ..., k_k)$. This MAC is calculated and used for node to node authentication prior to their communication.

When the initial secrets are distributed according to the tree protocol, a key ID is associated with each key and distributed to the sensor nodes along with the secret keys. So, when the new key (MAC) is calculated, it sends the key ID along with the message, so that, receiving node can calculate the MAC and verify sender as valid one. The receiving node checks the key ID and matches whether it is the same or not. Then it calculates MAC with its own key that has same key ID. The sender is valid if calculated MAC is the same as the received one.

RC5 is a symmetric block cipher designed to be suitable for both software and hardware implementation. It is flexible parameterized algorithm, with a variable block size, a variable

number of rounds and a variable length-key. This provides the opportunity for great flexibility in both performance characteristics and the level of security. For example, a particular RC5 algorithm is designed as RC5-w/r/b. The number of bits in a word, w, is a parameter of RC5. Different choices of this parameter will result in different RC5 algorithms. RC5 is iterative in structure, with a variable number of rounds, r, being the second parameter. It also uses a variable length of secret-key, b (in bytes) that is a third parameter of RC5.

4. Defense against HELLO Flood Attack

If each sensor node constructs a set of reachable neighbor nodes, and is only willing to receive REQ messages from this set of neighbor nodes, then REQ messages from an adversary transmitted with larger power will be ignored. Thus, the damage from a HELLO flood attack can be restricted within a small range. To defend against attack, each request (REQ) message forwarded by a node is encrypted with a key. As we have shown from the tree protocol that any two sensor nodes share some common secrets, the new encryption key is generated on-the-fly (i.e. during communication). In this way, any node's reachable neighbors can decrypt and verify the REQ message while the attacker will not know the key and will be prevented from launching the attack. We show that the new key combined with the echo-back mechanism can well protect this attack.

Each node locally broadcasts an echo message to its neighbor with format:

$$s_1 \rightarrow : ECHO||E_{new\text{-}key}\ (IDs_1||nonce)$$

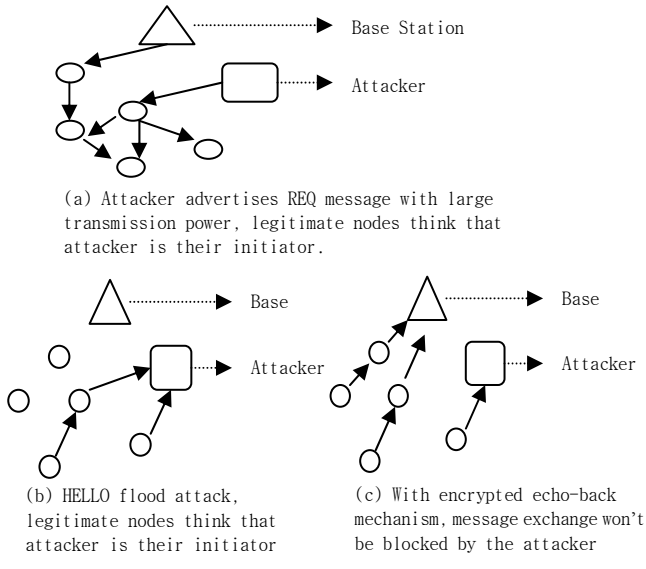

Figure 3. HELLO flood attack and defense

Where, ECHO is the message type, ID is the ID of the sensor node s_1, $E_{new\text{-}key}$ is the new key as we have mentioned earlier and nonce is the random number. If a node, say, s_2 receives this message; it sends echo reply with format:

$$s_2 \rightarrow s_1 : ECHOBACK||E_{new\text{-}key}\ (IDs_2||nonce)$$

When node s_1 receives this message, it records node s_2 as its verified neighbor. If an attacker obtains the shared secrets after a node has received its new encrypted key, it can not know the new pairwise key. Computing the pairwise key is more robust and secure in multiple tree protocol as we have described earlier, where we have shown that the probability of compromise of a secret is very low [1]. However, if an attacker obtains the new key, it can initiate echo-back many times by sending several echo messages. The attacker can generate false identities and can initiate Sybil attack, adding new nodes with false identities. To prevent such attacks, node should destroy its new key from memory after a certain time that is long enough to set up pairwise keys with all its neighbors. Again, during communication, it can calculate new key from the secrets they share.

Although this defense against "HELLO flood" attack is to verify the bidirectionality of a link with encrypted echo-back mechanism before taking meaningful action based on a message received over that link, this defense gets less effective when an attacker has a highly sensitive receiver as well as a powerful transmitter. If an attacker compromises a node before the feedback message, it can block all its downstream nodes by simply dropping feed back messages. And thus, such an attacker can easily create a wormhole to every node within range of its transmitter/receiver. Since the links between these nodes and attacker are bidirectional, the above approach will unlikely be able to locally detect or prevent a "HELLO flood". We propose a different way of reliable exchange of messages among nodes and base stations. We show that when any particular node has different route to send data, this problem is solved.

5. Multi-path Multi-base Station Data Forwarding

We describe how a sensor node can forward its sensed data to multiple routes i.e. multiple base stations in case where an attacker manages to compromise a sensor node. Given the shared secrets and the generated new key between two sensor nodes, the operation of setting up different routing paths is as follows:

Step 1: As each sensor node shares some common keys according to the secret distribution protocol (i.e. Multiple Tree Protocol), every node uses the echo-back scheme to identify its neighbor nodes and sets up pairwise new key with its verified neighbor nodes. Then it uses its new key to exchange messages among them.

Step 2: Each base station broadcasts its request (REQ) message to its neighbor nodes with the following format:

$$REQ||ID_x||E_{key}(ID_B||HCN)$$

Here, REQ is the message type, ID_x is the ID of the sending node x, ID_B is the base station ID who generated this request

message, E_{key} is the key (i.e. initial secret) that is common between any node to which base station floods the message and HCN is the base station's one-way hash chain number. Receiving node verifies that the REQ comes from the base station, then it forwards the REQ to its neighbor node, say, *y*, with the format:

$$REQ||ID_y||E_{new\text{-}key}(ID_B||HCN)$$

Step 3: When any ordinary node say, *y*, receives this REQ message, it checks the sender ID. If *s* is *y*'s verified neighbor, *y* decrypts and authenticates the sender with computed new key $E_{new\text{-}key}$. If the message sender is valid, it replaces the HCN with the new value and encrypts the REQ message with its $E_{new\text{-}key}$ and broadcasts the newly encrypted message.

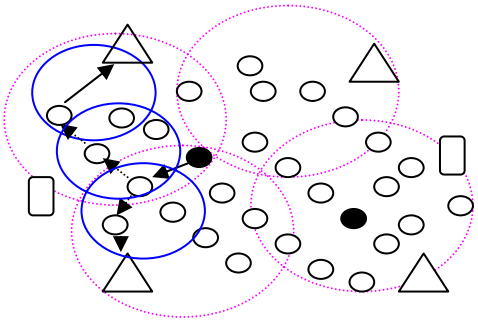

Figure 4. Ordinary node gets REQ message from compromised node but does not forward message to it, rather it sends message to its verified neighbor by alternative routes.

In this way, flooding of REQ messages securely establishes direction of routing without requiring feedback to each base station. Any base station, when receives the sensed data, it can cooperate with other base stations to interpret the sensed data as base station is powerful enough to communicate among themselves.

6. Robustness of the Network

Our defense mechanism considers multiple routing paths to avoid compromised nodes such that with different routes available to every node and the base station, the target is to transmit messages correctly in the presence of attackers and failed nodes.

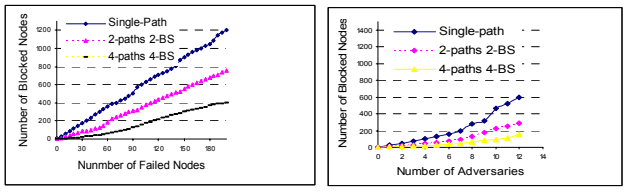

Figure 5. Robustness of Multi-path Routing. (a) Effects of Node Failure (b) Effects of Jamming Signals During Data Routing

We consider two cases to measure the effectiveness of multi-path routing. First, we consider the impact of node failure where, nodes that have failed can no longer forward packets (Fig. 5). Second, we experiment the effect of jamming attack that a compromised node may launch. In both cases, we show the average number of nodes that can be blocked. We measured the number of blocked nodes considering three scenarios; single-path, two-path with two base stations and four-path with four base stations. The experiments had been performed for a network of 2000 nodes randomly deployed in which each individual node has 12 neighbor nodes in average. 40 different combinations of nodes were randomly selected to be failed nodes. For jamming attack, we consider the case where the number of attackers repeatedly sending strong signals to reachable sensor nodes so that these victim nodes can not transmit their data packets. Figure 5 shows the results that the alternate path scheme increases the robustness of the network compared to single-path scheme. The improvement is even more when multi-path routing to multiple base stations is considered.

7. Discussion

In simple defense, we have shown every node to authenticate identity with shared secret by the means of encrypted REQ message with echo-back scheme. We have shown that if the protocol sends the encrypted messages in both directions over the link between the nodes, HELLO floods are prevented.

We have shown a different approach when node to node authentication does not prevent a compromised node. We present multi-path multi-base station routing. The flooding of REQ messages can securely establish direction without feedback to each base station. By setting up a new pairwise key from secret shared by nodes, multi-path routing improves intrusion tolerance. Specific one-way hash chain number (HCN) is addressed to defend against replay attack.

Placement of base stations depends on different applications with different constraints, e.g. number of total sensor nodes and number of base stations. Normally, base stations should be placed far away from each other to make the system resilient to node compromise.

Each sensor node needs to save shared keys, one-way hash chain number, and several random numbers. If each initial secret key is 32 bits long and a node communicates with n neighbors, keeps r random numbers, and there are b base stations, then the node needs 4X4(2n + b +r + 2) bytes to store all keys with 4 initial secret keys in each node. 496 bytes are needed if there are 4 base stations, 10 neighbor nodes, 5 random numbers. Current sensor nodes provide 4 KB SDRAM, 128 KB flash memory, 4KB embedded EEPROM, and 128K extended EEPROM. If the keys are not changed often, they can be stored in the 4KB embedded EEPROM.

There are some important issues that need further improvement of how to realize the new key establishment and its computation complexities. Issues like message loss, nodes joining and leaving the network also need to be addressed and are left as future works.

8. Conclusion

This work described the defense against Lap-top class attacker (who can launch HELLO flood attack) by introducing node to node authentication and multi-path routing using shared secret between sensor nodes. We have adopted a probabilistic key assignment protocol among sensor nodes and during communication, each node can calculate a pairwise key using these common secrets and hence improving the network resilience against security threats. We have shown that our defense mechanism can well tolerate the damage launched by the intruder.

REFERENCES

[1] S. S. Kulkarni, M. G. Gouda, and A. Arora, " Secret instantiation in ad-hoc networks", Special Issue of Elsevier Journal of Computer Communications on Dependable Wireless Sensor Networks, 2005, pp. 1–15

[2] F. Ye, H. Luo, S. Lu, and L. Zhang:, "Statistical En-Route Filtering of Injected False Data in Sensor Networks", IEEE Journal on Selected Areas in Communications, Vol. 23, no. 4, 2005 pp 839 - 850.

[3] C. Karlof and D.Wagner, "Secure routing in wireless sensor networks: Attacks and countermeasures", Elsevier's Ad Hoc Networks Journal, Special Issue on Sensor Network Applications and Protocols, 1(2–3) 2003, pp. 293–315.

[4] R. Di Pietro, L. V. Mancini, and S. Jajodia, "Providing secrecy in key management protocols for large wireless sensors networks", Journal of AdHoc Networks, 1(4), 2003, pp. 455-468.

[5] V. Wen, A. Perrig, and R. Szewczyk, "SPINS: Security suite for sensor networks", Proc. ACM MobiCom, 2001, pp. 189–199.

[6] H. Luo, J. Kong, P. Zerfos, S. Lu, and L. Zhang' "URSA: Ubiquitous and robust access control for mobile ad hoc networks", Proc. IEEE/ACM Trans. Netw., Vol. 12, no. 6, 2004, pp. 1049–1063.

[7] A. Arora, P. Dutta, S. Bapat, V. Kulathumani, H. Zhang, V. Naik, V. Mittal, H. Cao, M. Demirbas, M. Gouda, Y. Choi, and et al., "A Line in the Sand: A Wireless Sensor Network for Target Detection, Classification, and Tracking", Computer Networks (Elsevier), Special Issue on Military Communications Systems and Technologies, 46(5) 2004, pp. 605–634.

[8] J.R. Douceur, "The Sybil attack", 1st International Workshop on Peer-to-Peer Systems (IPTPS _02) 2002.

[9] Y. Hu, A. Perrig, and D. Johnson, "Rushing attacks and defense in wireless ad hoc network routing protocols", Second ACM Workshop on Wireless Security (WiSe'03), San Diego, CA, USA, 2003.

[10] A. Menezes, P. Oorschot, S. Vanstone, Handbook of Applied Cryptography, CRC Press, Boca Raton, 1996.