Secured Two Phase Geographic Forwarding with GSS Algorithm

B. Prathusha Laxmi[1] and A. Chilambuchelvan[2]

[1] Department of Computer Science and Engineering,
R.M.K. Engineering College, Kavaraipettai,
Chennai, India
prathushakrishnaa@yahoo.com
[2] R.M.K. Group of Institutions, Kavaraipettai,
Chennai, India
chill97@gmail.com

Abstract. This paper focuses on improving the secured geographic routing performance in Wireless Sensor Networks and proposes a Geographic routing oriented Sleep Scheduling algorithm (GSS) on secured TPGF. The length of the explored transmission path of the Secured Two Phase Geographic Forwarding (SecuTPGF) can be reduced compared with the Two Phase Geographic Forwarding connected-K neighborhood algorithm.

Keywords: TPGF, SecuTPGF, GSS, MAC, WSN, WMSN.

1 Introduction

Multimedia sensor nodes are developed to provide more comprehensive information to enhance the capability of traditional WSNs for event description. Efficiently transmitting multimedia streams in Wireless Multimedia Sensor Networks (WMSN) is a significant challenging issue, due to the limited bandwidth and power resource of sensor nodes. SecuTPGF is one of the first designed secured routing protocol in WMSNs. SecuTPGF is designed for non adversarial networks and is not susceptible to outsider attacks. For example, an enemy who is able to compromise an authentic network node; may easily launch more serious insider attacks, by extracting key and security information from the compromised node, and then act as an authentic network participant. This can overcome in the SecuTPGF, which will not disrupt the network operations.

The SecuTPGF protocol provides the following functions:

(1) Preventing outside adversaries from joining the network;
(2) Limiting the impact of insider attacks in a localized area;
(3) Partially detecting insider attacks and avoided them in the network;
(4) Authenticating control messages exchanged between nodes.

V.V. Das and J. Stephen (Eds.): CNC 2012, LNICST 108, pp. 25–30, 2012.

2 SecuTPGF Routing Protocol

2.1 SecuTPGF Overview

The first problem that is addressed in SecuTPGF is achieving source The authentication and protection of mutable information in routing messages. In solution, use message authentication code (MAC) to tackle these problems. The second problem addressed is authentication of node identity and calculation of symmetric key between nodes. In SecuTPGF, using ID-NIKDS Scheme to mitigate these problems, which can avoid the using of certificates for public key authentication. And, no interaction is required to determine the symmetric key between nodes except their unique IDs. Finally, to limit the impact of insider attack, a bootstrapping time information is involved in authentication procedure.

2.2 Operations of SecuTPGF

In general, the operation of SecuTPGF relies on three activities: Initialization and Key setup, secure Neighbor Discovery and Secure Route Discovery.

A. Initialization and Key Setup
This stage is to be executed by the Wireless Sensor Network(WSN) Manager acting as a trusted authority (TA), using its own facilities for processing in order to minimize the nodes power consumption. To startup an ID_NIKD scheme, the base station first needs to generate and distribute private keys and public parameters. It preloads each node X with the values (IDx,Sx,Kx,Ti) and compute public key of any node knowing the ID of the node. Once Initialization stage is completed, all nodes are ready to be deployed into field. The Secure Neighbor Discovery Phase starts up after the network deployment.

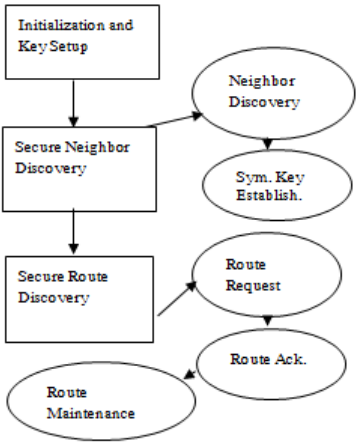

Fig. 1. Flow chart of SecuTPGF

B. Secure Neighbor Discovery

When it is deployed, node *A* bootstraps itself at a preset time Ti_A and tries to discover its neighbors. It broadcasts a HELLO message, which contains its ID (ID_A), its geographic location (L_A), bootstrapping time (Ti_A), and a random nonce (N_A), and then waits for each neighbor *B* to respond.

$$a \text{ ->*\_ : } HELLO(ID_A, L_A, Ti_A, N_A)$$

Node *B* first validates whether the bootstrapping time Ti_A is within a pre specified threshold *L* with its current time *t*. If the check fails, node *B* simply discards the request. Otherwise, *B transmits to A a challenge message* that contains its ID (ID_B), geographic location (L_B), bootstrapping time (Ti_B), a random nonce (N_B), and an authenticator (V_B) calculated as

$$H(k_{BA}, L_B || L_A, Ti_B || Ti_A, N_B || N_A)$$

where H is a hash function.

$$b \text{ -> } a : (ID_B, L_B, Ti_B, N_B, V_B)$$

Upon receiving this *challenge*, node *A* proceeds to compute a verifier as

$$V'_B = H(\hat{}e([s]P_A, P_B), L_B || L_A, Ti_B || Ti_A, N_B || N_A)$$

By the bi linearity of the pairing , the verification is successful if and only if both *A* and *B* have the authentic private keys corresponding to their claimed bootstrapping time. After verifying the equality of V'_B and V_B, node *A* computes a verifier as $V_A = H(k_{AB}, L_B, Ti_B, N_B)$ and sends valid response to node *B*. Node *A* also calculates symmetric key and add node *B* into its neighbor list.

$$a \text{-> } b : (ID_A, V_A)$$

Using a similar approach as node *A*, node *B* verifies that whether node *A* is an authentic neighbor and then establishes a secure link and adds it into its neighbor list.

C. Secure Route Discovery

The source node initiates and forwards a *request message* to intermediate node that is the one hop neighbor nearest to the base station among all its neighbor nodes. The *request message* contains *message identifier (rreq), the ID of the source node (S), the geographic location of the base station (Dloc), a request path number (Pno), and a MAC field*. The MAC field is computed over all elements with a key shared by the Source (S) and the base station (D) (*MACkSD*(*rreq, S,Dloc, Pno*)). The request path number is incremented each time when source node initiates a new route request. The size of the generated MAC is 4 byte.

When the intermediate node receives a *request message* for which it has no next hop node to send, it sends *Block Node message* to its previous-hop node. The *Block Node message* is authenticated using a shared key between the intermediate node and the previous-hop node. Otherwise the intermediate node modifies the request by appending its ID in the path list of the *request message* and replacing the MAC field with a MAC computed on the entire *request message* using a key shared between the base station and the intermediate node. The intermediate node also checks if the path can be optimized. The path will be optimized, if the source or the farthest node listed in the path list (ID sequence) of the *request message* is a neighbor of the intermediate

node. And, if the path is optimized, the intermediate node appends optimized neighbor ID in the path list before its ID when the request is modified.

For example, an intermediate node 'e' receives a *request message* for which the path list contains "a->b->c->d" and nodes 'b' and 'c' are neighbors of node 'e'. The intermediate node 'e' checks whether the source node is a neighbor, if it is not, then searches the path list from the beginning node (node 'a') till it finds a neighbor node in the path list. The searching returns the farthest (the farthest in ID sequence, but not on geographic distance) neighbor node 'b', and then the path list in the *request message* for node 'e' will be modified as "a->b->c->d->b->e". Finally, the intermediate node records the address of the neighbor from which it received the request, and then the modified route request is forwarded. This process is repeated until the *request message* reaches the base station.

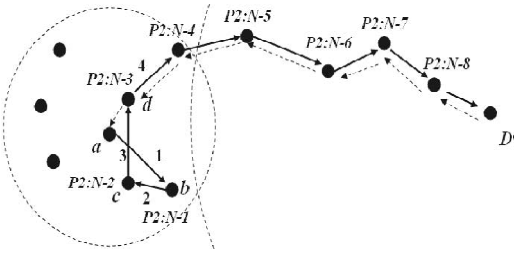

Fig. 2. The dash line shows the reverse travelling in the found path

When the base station receives the *request message*, it verifies the MAC. If this verification is successful, the base station continues to search a duplicated node ID in the path list of the *request message* to get optimized path. If the base station finds a duplicate node ID, it assumes that the next node after the duplicated ID and the duplicated ID nodes are neighbors, so it removes the nodes' IDs in between the two neighbor nodes to get the optimized path.

Route maintenance mechanism detects malfunctioning, dead or subverted nodes along the routing path. In SecuTPGF, each node along the path forwards the data to the next hop node and then attempts to confirm that the data was received by the next hop node. If, after a limited number of local retransmissions of the data, a node in the route is unable to make this confirmation, it propagates a *route error message* (RERR) to the source node to inform that the link is broken. The initiator of *route error message* computed a MAC using a non interactive key. Upon receiving a *route error message*, the source authenticates the RERR and then may re-initiate the route discovery process for the destination.

3 Secured Two Phase Geographic Forwarding with GSS Algorithm

Secured Two Phase Geographic Forwarding with GSS Algorithm and the pseudo code is shown below. During the first part of GSS, the geographic location(gu) of each authentic node u is obtained and the potential nearest authenticated Neighbors to

the sink for each node is identified. In the second part of GSS, a random rank ranku of each node u is picked and the subset Cu of u's currently awake authenticated Neighbors having rank < $rank_u$ is computed. Before 'u 'can go to sleep, it needs to ensure that all nodes in Cu are connected by nodes with rank<ranku each of its authenticated Neighbors has at least k neighbors from Cu and it is not the potential nearest authenticated neighbor node for other nodes.

3.1 Pseudo Code of GSS Algorithm

First: Run the following at each node u.

1. Get its geographic location gu of the current authentic node.
2. Broadcast gu and receive the geographic locations of its all authenticated neighbors Au. Let Gu be the set of these geographic locations.
3. Unicast a flag to w, w Au and gw is the closest to sink in Gu.

Second: Run the following at each node u.

1. Pick a random rank $rank_u$.
2. Broadcast $rank_u$ to the authenticated node and receive the ranks of its currently awake neighbors Nu & authenticate. Let Ru be the set of these ranks.
3. Broadcast Ru and receive Rv from each authenticated v 2 Nu.
4. If |Nu| < k or |Nv| < k for any v ϵ Nu, remain awake. Return.
5. Compute Cu = {v|v ϵ Nu and $rank_v$ < $rank_u$}.
6. Go to sleep if both the following conditions hold. Remain awake. Otherwise.
   - Any two nodes in Cu are connected either directly themselves or indirectly through nodes within u's 2-hop neighborhood that have rank less than $rank_u$.
   - Any node in Nu has at least k neighbors from Cu.
   - It does not receive a flag.

4 Conclusion

Aiming at improving Secured Two Phase Geographic Forwarding, the GSS algorithm is combined with the secured TPGF. By making less potential nodes closest to sink go to sleep. The proposed GSS algorithm can own a better first transmission path when transmitting data in WMSNs with SecuTPGF.

References

1. Shu, L., Zhang, Y., Yang, L.T., Wang, Y., Hauswirth, M., Xiong, N.X.: TPGF: Geographic Routing in Wireless Multimedia Sensor Networks. Telecommunication Systems 44(1-2), 79–95 (2010)
2. Shu, L., Zhang, Y., Zhou, Z., Hauswirth, M., Yu, Z., Hynes, G.: Transmitting and Gathering Streaming Data in Wireless Multimedia Sensor Networks within Expected Network Lifetime. In: ACM/Springer Mobile Networks and Applications(MONET), vol. 13(3-4), pp. 306–322 (2008)

3. Mulugeta, T., Shu, L., Hauswirth, M., Chen, M., Hara, T., Nishio, S.: Secure Two Phase eographic Forwarding Routing Protocol in Wireless Multimedia Sensor Networks. In: Proc. Intl. Conf. Global Communication (Globecom 2010), Miami, Florida, USA (2010); Die, W., Hellman, M.: New directions in cryptography. IEEE Transactions on Information Theory 22(6), 644–654
4. Guerrero-Zapata, M., Zilan, R., Barcel-Ordinas, J., Bicakci, K., Tavli, B.: The Future of Security in Wireless Multimedia Sensor Networks. Telecommunication Systems 45(1), 77–91 (2010)
5. Karlof, C., Wagner, D.: OhHelp: Secure routing in wireless sensor networks: Attacks and countermeasures. In: Proc. Intl. Workshop on First Sensor Network Protocols and Applications in Conjunction with ICC 2003, AK, USA, pp. 113–127 (2009)
6. Sakai, R., Ohgishi, K., Kasahara, M.: Cryptosystems based on pairing. In: Proc. Intl. Symposium on 2000 Cryptography and Information Security, Okinawa, Japan, pp. 26–28 (2000)
7. Zhu, C., Yang, L.T., Shu, L., Joel Rodrigues, J.P.C., Hara, T.: Proc. Intl. Conf. Global Communication(Globecom 2010), Miami, Florida, USA, pp. 140–127 (2010)