Collaborative Trust-based Secure Routing in Multihop Ad Hoc Networks

Niki Pissinou[1], Tirthankar Ghosh[1], Kia Makki[1]

[1] Florida International University, Miami, Florida
{pissinou, tirthankar.ghosh, makkik}@fiu.edu

Abstract. In this paper we have proposed a secure routing protocol based on AODV for multihop ad hoc networks. Our protocol is unique in the sense that it is capable of finding a secure end-to-end route free of any malicious entity, thus resisting an internal attack within the network either in the form of compromised or disloyal nodes. We propose to find a secure and efficient route to a destination based on collaborative effort of all the nodes.

1 Introduction

Most of the research on ad hoc networks, so far, has been done in the area of routing protocols, though, in recent years, security issues have also been addressed. As the ad hoc nodes are characterized by minimum trust for each other, finding a secure end-to-end route is truly challenging. Most of the work on routing security focus on the efficient use of digital signatures or shared secret keys to authenticate and confide the data and routing headers. However, they always tend to find the shortest path between source and destination irrespective of the presence of malicious nodes in between.

In this paper we propose a secure routing algorithm to find a secure end-to-end route based on collaborative effort of all the nodes in an ad hoc network, which can withstand the attack of any malicious entity, like a compromised or a disloyal node. Our protocol is robust against any internal attack within the network tending to inject malicious routing information and disrupting the network operation.

2 Related Work

Not much work has been done to find an end-to-end secured route in ad hoc networks. Some work has been done to design some secured routing protocols in a public key infrastructure. [8,7,4]. The protocol proposed in [8] works under the assumption of a trusted certificate server, which itself violates the basic paradigm of MANET. The security of the protocol proposed in [7] is based on the assumption of the existence of an efficient key management system that enables the nodes to obtain public key information of other nodes. The authors, however, did not consider the threat from compromised node.

Some symmetric key solutions are also proposed to secure ad hoc networks [5,6,9]. In [5] the authors have proposed a protocol whose security depends upon the assumption of an efficient distribution of shared secrets between the nodes. This itself is a burning research problem. A similar approach is proposed in [6]. The security here is based on the efficient use of one-way hash function and works under the assumption that some secure means of distributing the elements of the hash chain is already there.

A token based approach is proposed in [9] where the use of threshold cryptography is suggested to distribute the tokens securely among the nodes. Another protocol, proposed in [12], aims at isolating the misbehaving nodes, thus making noncooperation unattractive. The monitoring mechanism is implemented by a neighborhood watch concept where the no-forwarding behavior of the nodes are monitored and reported.

In [4], the authors have proposed a protocol for securely discovering the network topology in a public key infrastructure. The protocol is responsible for securing the discovery and distribution of link state information. Another protocol to achieve a similar goal is proposed in [3], which works under the assumption of an already established shared secret between the source and destination.

All the above solutions tend to find the shortest path from source to destination irrespective of some malicious nodes in between. In [1], the authors have proposed a secured routing protocol based upon the trust level of the nodes. Although their approach is unique, the protocol fails under the attack of a compromised or a disloyal node.

3 Our Protocol

3.1 Goals and Assumptions

Our design is based on the following assumptions which we think are justified. First, there is a prior distribution of trust level[1] of all the nodes. Second, all the nodes communicate via a shared wireless channel and all communication channels are bi-directional. Third, all the nodes operate in a promiscuous mode. Fourth, we do not consider here physical layer or MAC layer security. Instead, we concentrate on the network layer. Our proposed routing protocol is actually a secure extension of AODV. Last but not least, we assume that all the nodes are identical in their physical characteristics, i.e., if node A is within the transmission range of B, then B is also within the transmission range of A.

3.2 Protocol Description

Essentially all routing protocols in the ad hoc community tend to find the shortest path to the destination irrespective of the presence of any malicious node in that path. We can argue that, as internal threat[2] in the network in the form of a compromised[3] or disloyal[4] node is of significant concern, a path free of malicious node is more important than the shortest path. The protocol that we propose here is an extension of the Ad hoc On Demand Distance Vector (AODV) routing protocol. The protocol works as follows:

[1] defined in line with the organizational hierarchy of the specific application of the network deployment which is not discussed in this paper. This metric can be dynamically changed depending upon the history of the past behavior of the nodes. We are currently working on that.

[2] defined as an active attack by a compromised or a disloyal node which actively takes part in the ongoing communication.

[3] defined as a node which has been physically taken over by an intruder thus giving access to all its stored secrets and system codes.

[4] defined as a node which has ended its loyalty to the network and has decided to disrupt the network operation by non-cooperation of some means.

When a node wants to find a route to another node, it initiates a route discovery. The RREQ packet header contains a *trust_level* field, in addition to the other fields in AODV RREQ. When an intermediate node receives the RREQ packet, it rebroadcasts it after modifying the *trust_level* field to include the trust level of the node that sends it the RREQ. Every node checks back the rebroadcasted RREQ packet from its next node to see whether it has provided the proper information. If not, it immediately broadcasts a warning message questioning the sanctity of that node. Our protocol does not encourage any intermediate node to send a route reply. The final route selection is based upon the *trust_level* metric. H*op_count* plays a role in deciding the final route only when more than one packet has same *trust_level*. The RREP packet has the next hop information. This is in line with the solution given in [11] to counter the black hole problem. When the source node gets back the first RREP, it waits for a specified amount of time before using that route. If within that time another RREP comes, the source node queries the next hops of the two RREPs. The next hop of the malicious RREP will obviously not have the same route to the destination. Thus, malicious route injection into the network can be prevented.

The pseudocode below shows the action of a node after it receives a route request packet.

```
// when a node receives a Route Request packet
   Receice_RREQ( ) {
 // check whether it is the destination of the route
 // request
     if destination {
      compute_highest_trust_level( )
      // in case more than one RREQ has same trust_level
      // decides on the basis of lowest hop_count
       sends_RREP_to_source( ) }
     else (not destination) {
        if duplicate packet {
          cross_checks_trust_level( )
          if found ok
            drops the packet
          else
            broadcasts roure_warning message( ) }
        else (not duplicate) {
               modifies trust_level
               increments hop_count
               rebroadcasts RREQ }}}
```

The pseudocode below shows the detailed action of the source node after it receives the first route reply.

```
  // when the source node gets back the first Route Reply
     Receive_RREP( ) {
       waits for a specified period
       if receives another RREP {
          queries next_hop( ) }
```

```
        else {
             sends data( ) }}
```

The function *cross_checks_trust_level* can be implemented in two ways. When an intermediate node receives a duplicate route request packet, it checks back the *hop_count* field to find out from which node it is receiving the packet. The following algorithm implements the function.

```
    if (current ->hop_count = = hop_count - 1) {
    cross_checks_trust_level( ) }
    else {
      // the node is trying to put malicious information
      // finds out which node is malicious
      broadcasts roure_warning message( ) }
```

The above algorithm works under our assumption that all the nodes are identical in their radio range. If they are not, a node can receive a duplicate route request from any other node which it cannot reach directly and wrongly assume that the later is trying to act malicious. This will generate false warning messages in the network.

The second possible implementation takes care of this. An intermediate node, on receiving a duplicate route request packet, extracts the address stored in the *lastaddr* field (the *lastaddr* field contains the address of the node from which the next node receives a route request packet) and checks from the neighbor table whether it is from any of its neighbor. The algorithm works as follows:

```
     if (lastaddr = = neighbortable->addr) {
      cross_checks_trust_level( ) }
     else {
      drops the packet( ) }
```

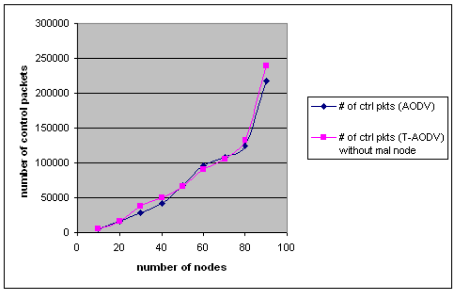
The above implementation can actually increase the computational overhead in each node. The complexity can however be reduced by efficient searching of the neighbor table.

4 Simulation and Results

We have used Glomosim for our simulation. Glomosim is a scalable simulation software used for mobile ad hoc networks. We defined a region of 2 Km by 2 Km with random node placement. The nodes move with uniform speed chosen between 0 to 10 meters/sec. The pause between each successive movement is 30 seconds.

The results that we got confirm the efficiency of our protocol. We have benchmarked our protocol, which we call *Trust-embedded AODV (T-AODV)* with the original AODV protocol. The small percentage increase in overhead (Fig.1) can be traded off with the incorporation of security into the protocol. This increase is due to retransmission of some route request packets because of delayed receipt of route reply by the source nodes (as we do not encourage intermediate nodes to send route reply in our protocol). Actually, this overhead can be brought down by increasing the

NET_TRAVERSAL time. We can also see from Fig.1 that the percentage variation in overhead decreases with increasing number of nodes. This is because, in AODV, as more and more nodes join the network, the probability of sending route replies by intermediate nodes increases, which is not the case in T-AODV as no intermediate node can send route replies.



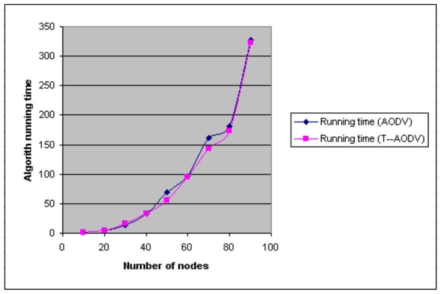
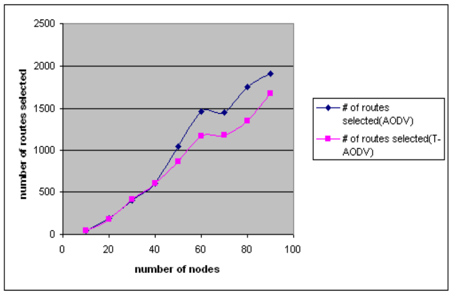


Fig. 1. Routing overhead

Fig. 2. Algorithm Running times

The comparison of the algorithm running times in Fig.2 also shows that our protocol is efficient which runs more efficiently with increased number of nodes. A probable explanation of this is, in AODV, as we increase the number of nodes in the network, more nodes tend to find a route from its cache. In our protocol, as we do not encourage intermediate nodes to send route replies, the running times becomes lower than AODV as more nodes join the network. The time can go up by a small percentage if we increase the NET_TRAVERSAL time to lower the routing overhead.

As no intermediate node is encouraged to come up with route replies, we obviously have lesser number of routes selected in our protocol that that in AODV (Fig.3). However, this should not give any misconception that some of the routes are not properly selected. In fact, our protocol has lesser number of route errors reported than that in AODV (Fig.4). Lesser number of routes selected, in effect, renders lower processing overhead for the source nodes, as they do not have to process all the route replies from the intermediate nodes.



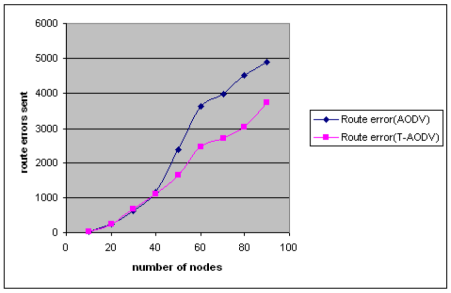


Fig. 3. Numbers of Routes selected

Fig. 4. Route Errors sent

5 Conclusion

Currently we are working on two extensions of the protocol. We are developing a dynamic trust model instead of a predistributed static one. We are also working to make the protocol robust enough to withstand the attack from multiple malicious nodes colluding to disrupt the network, which is not currently incorporated.

References

1. Yi, S., Naldurg, P., Kravets, R.: Security-Aware Ad hoc Routing for Wireless Networks. Report No. UIUCDCS-R- 2001-2241, UILU-ENG-2001-1748, August 2001.
2. Perkins, C., Royer, E.: Ad hoc On-Demand Distance Vector Routing. In Proc.IEEE Workshop on Mobile Computing Systems and Applications, 1999.
3. Papadimitratos, P., Haas, Z. J.: Secure Routing for Mobile Ad hoc Networks. In Proc. SCS Communication Networks and Distributed Systems Modeling and Simulation Conference (CNDS 2002), San Antonio, TX, January 27-31, 2002.
4. Papadimitratos, P., Haas, Z. J.: Secure Link State Routing for Mobile Ad hoc Networks. In Proc. IEEE Workshop on Security and Assurance in Adhoc Networks, in conjunction with the 2003 International Symposium on Applications and the Internet, Orlando, FL, January 28, 2003.
5. Hu, Yih-Chun, Perrig. A., Johnson, D. B.: Ariadne: A Secure On-Demand Routing Protocol for Ad hoc Networks. MobiCom ’02, September 23-26, 2002, Atlanta, Georgia, USA.
6. Hu, Yih-Chun, Perrig. A., Johnson, D. B.: SEAD: Secure Efficient Distance Vector Routing for Mobile Wireless Ad hoc Networks. In Fourth IEEE Workshop on Mobile Computing Systems and Applications (WMCSA ’02), June 2002, pages 3-13, June 2002.
7. Zapata, M. G., Asokan, N.: Securing Ad hoc Routing Protocols. WiSe ‘02, September 28, 2002, Atlanta, Georgia, USA.
8. Sanzgiri, K., et al,: A Secure Routing Protocol for Ad hoc Networks. In Proc of the 10th IEEE International Conference on Network Protocols (ICNP’02).
9. Yang, H., Meng, X., Lu, S.: Self-Organized Network Layer Security in Mobile Ad hoc Networks. WiSe ’02, September 28, 2002, Atlanta, Georgia, USA.
10. Zhou, L., Haas, Z. J.: Securing Ad hoc Networks. IEEE Network, November/December 1999.
11. Deng, H., Li, W., Agrawal, D. P.: Routing Security in Wireless Ad Hoc Networks. IEEE Communications Magazine, October 2002.
12. Buchegger, S., Le Boudec, Jean-Yves.: Performance Analysis of the CONFIDANT Protocol (Cooperation Of Nodes: Fairness In Dynamic Ad-hoc Networks), MOBIHOC ’02, June 9-11, 2002, Switzerland.