Securing the Ad Hoc Dynamic Source Routing Protocol

Rajan Shankaran, Vijay Varadharajan, Michael Hitchens

Department of Computing, Macquarie University, Sydney Australia

{rshankar, vijay, michaelh}@ics.mq.edu.au

Abstract-Ad-hoc routing protocols are challenging to design and the need for security services further complicates the situation. So far the proposals for routing in ad-hoc networks offer no security mechanisms at all, or have only partial solutions for protecting the routing. In this paper, we present a scheme for providing security services for routing of control messages in an ad-hoc network. Our focus is on on-demand routing protocols for ad-hoc networks, specifically the Dynamic Source Routing Protocol.

Keywords: Mobile, security, ad hoc

I. Introduction

A mobile ad hoc network (MANET) may be defined as a collection of mobile hosts that maintain interconnection without the intervention of a centralised access point. In other words, such networks are self-organising and adaptive. The self-organising property implies that a network can be formed on the fly and then change its topology without the presence of system administration entities. The term 'adaptive' simply implies that an ad-hoc network can take different forms and has highly variable mobile characteristics such as power and transmission conditions.

There are two classes of ad-hoc routing protocols; on-demand and table driven. It has been demonstrated that the former perform better with significantly less overheads than the latter in many situations [1, 4, 6, 8]. Our focus is on on-demand routing protocols for ad-hoc networks, specifically the Dynamic Source Routing (DSR) protocol [5]. DSR is a simple and efficient routing protocol that facilitates load balancing by allowing multiple routes to any destination.

In this paper[1] we present an approach to securing DSR. It allows both sender and receiver to securely identify intermediate nodes, allows for identification of the source of a message and handles both uni- and bi-directional communication, as allowed for by DSR

II. Routing in DSR

DSR employs explicit source routing. All the packets carry a list of intermediary nodes that they should pass through from the sender to the destination. The source is able to select an ideal path amongst a set of available routes but needs to first discover the possible routes. When a source *S* needs to send data packets to a destination *D*, *S* first checks its local cache to determine if it has an available path to *D*. If the cache does not have an appropriate entry, *S* has to initiate the *Route Discovery* phase. *S* generates a route request (*RREQ*) packet which is broadcast it to all of its neighbours. A *RREQ* message has unique request identifier and a record field. The purpose of the record field (initially empty) is to record the address of each intermediate node. After receiving a *RREQ* packet from *S*, an intermediate node first determines if it is the intended destination. If so, it returns a route reply (*RREP*) packet to *S*. Otherwise the node will append its ID to the *RREQ* packet and broadcast it to its neighbours without changing the request ID. Every intermediate hop will perform this broadcast based search process till the packet reaches the intended destination *D*. When *D* receives a *RREQ* it will generate a *RREP* to be sent to the source *S***.** To find a path back to *S*, D has a number of options. It may have a suitable path in its cache. It may simply reverse the path in the *RREQ*, if all links in the path are bi-directional. If neither of these is possible then *D* must initiate its own *Route Discovery* for *S*. Piggybacking is used to prevent infinite recursion of route discovery in this case.

Whenever a node cannot deliver a packet to the next node in the chosen route, it unicasts a *RERR* message back to the source.

III. Security Issues in DSR

Obviously the security of the information in the routing messages is paramount to the security of DSR routing. Possible attacks in the absence of security include:

- Flooding – by initiating multiple false *RREQ*/*RREP* messages.
- False reporting – by propagating false control messages either with or without a false identity
- Replay – by recording and retransmitting old messages.
- Disruption – by malicious alteration of the contents of control messages.
- Rushing – by rapid dissemination of *RREQ* messages.

Rushing attacks may be specifically targeted at on-demand routing protocols that use duplicate suppression at each node [3]. They may cause the loss of any later legitimate *RREQs* as nodes drop them due to the duplicate suppression.

Considering the above we have determined a number of basic requirements for security in DSR routing:

[1] This work was supported by ARC Research Grant funding from Macquarie University

1-4244-0517-3/06/$20.00 ©2006 IEEE

- It should be possible to identify the source of a message. This will make it easier to identify the source of false messages and help mitigate the effects of denial of service attacks.
- It should be possible for the sender and ultimate destination of a message to have some guarantee of the identity of all the nodes that make up the path between them. This will assist in the identification of the source of false reporting, for example inaccurate routes.
- It should be possible to detect if a message has been altered in transit. This will prevent malicious nodes altering the contents of messages.
- It should be possible to check the freshness of a message. This will help prevent replay.

IV. Security Framework for DSR

In this section we propose a security framework for DSR to meet the requirements given in the previous section. We make the following assumptions:

- The scheme is based on public key cryptography.
- Node have relatively synchronized clocks
- Each node has a public and private key pair.
- Each node is capable of storing its own certificate and, as required, those of other nodes.
- Network links may be uni- or bi-directional.

Security services are implemented by extending existing control messages of the DSR protocol. There are no changes to the protocol operation itself but each node now performs additional, security related functions, when DSR messages are exchanged.

We opt for a public key based system over a symmetric key based approach for the following reason. Setting up shared secret keys requires pre-existing confidentiality, whereas a public key system does not. Furthermore, fewer public keys than secret keys are generally needed, because in a network with n nodes only n public keys are needed, and can potentially be broadcast, whereas n (n+1)/2 secret keys need to be set up in the case of pair-wise shared secret keys [2]. We recognise the extra computational requirements of asymmetric cryptography but contend that such cryptography makes it far easier to meet the requirements laid out above.

Assumption (1) requires a scheme for learning, storing, and distributing public keys of all routing nodes. It also requires a guarantee that public keys indeed came from a trusted party and therefore requires the use of a certificate and a certification authority. All nodes must know the public key of this authority. This approach may not appear to lend itself well to mobile ad-hoc networks since this requires the presence of some infrastructure. Note, however, that we have not assumed the existence of a directory infrastructure for the storage and distribution of node certificates. Essentially we envisage nodes storing their own certificates and supplying them to other nodes as required. This could, for example, be implemented using a Pretty Good Privacy (PGP) [11] based public key system.

V. Secure DSR Protocol Messages

In outline our protocol extends DSR as follows:
RREQ messages securely identify the originator of the message and all the intermediate broadcasting nodes (if different *RREP* messages do the reverse path validation by requiring each hop in the routing list confirm its commitment to the routing process by inserting its identity and correlating its response to its previous upstream hop's request to prevent any masquerading or a replay attack.

RRER messages securely identify their originating node as well as all the intermediate hops specified by the original message sender up to the originating node of the error message.

We use the following notation to describe our security protocols and cryptographic operations:
RREQ-U, RREP-U and RRER-U are the existing (unsecured) versions of the DSR protocol messages. RREQ-U_x, RREP-U_x and RRER-U_x are messages in the original format sent by node x.
CERT-X: Certificate of X generated by some CA.
T_x, N_x, L_x: Timestamp, Nonce and Lifetime associated with a message issued by some node X.
$P_{K\text{-}X}$: Public Key of a principal X.
$S_{K\text{-}X}$: Private Key of a principal X.
[….] $P_{K\text{-}X}$: The contents within […] are encrypted using the public key of principal X, $P_{K\text{-}X}$.
{….} $S_{K\text{-}X}$: The contents within {…} are hashed and signed using the private key of principal X, $S_{K\text{-}X}$.

The route discovery process in DSR begins when the source node floods the network with *RREQ* message. In response to this request, the target node or some intermediate node must respond with a *RREP* message.

The most important change to the *RREQ* message is to add timestamps, lifetimes and signatures. This allows nodes to identify replayed messages (in unsecured DSR a malicious node could easily alter the *RREQ_ID*). The timestamp also gives nodes to some chance of identifying messages passed on by intermediate nodes who have not added them self to the route list in an attempt to invisibly insinuate themselves into routes.

Assume that a source S is trying to discover a route to a destination node D and that such a route exists, with intermediate nodes B and C. The protocol steps are as follows:

(1) S → Broadcast: RREQ-U_S, L_S, T_S, Cert_S, Sig_S
where **Sig_S: {RREQ-U_S, L_S, T_S} $S_{K\text{-}S}$**

The sending node must authenticate itself to other nodes when passing its request to locate a target destination. The source S achieves this by broadcasting the *RREQ* with the following security extensions. It contains a lifetime L_S that indicates how long this request is valid. If a target node receives this request message and finds that the period L_S has expired it must discard this message. A timestamp T_S together with the *RREQ ID* in the original message format will indicate the freshness of the message and help prevent replay attacks. This message content is signed under the private key of the sender so that receiving nodes can verify the message

1-4244-0517-3/06/$20.00 ©2006 IEEE

contents. The sender's certificate is included for the benefit of those nodes that do not already have that certificate. The sender's identity is not separately included in the signed part as it is part of *RREQ-U_S*. The extended format of the *RREQ* message follows:

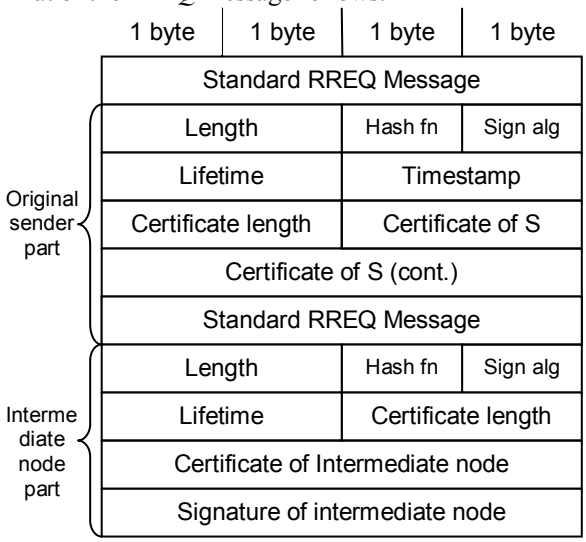

Figure 1: Secure Extensions to RREQ Message

Upon receiving such a message, the next hop node *B* validates the signature of the signed element using the public key of *S*. *B* then checks for replays, using the *RREQ ID* and timestamp and validates the lifetime of the message to determine whether it has expired or not. If any of these tests fail then B must discard this message. Otherwise, B rebroadcasts the *RREQ* as follows:

(2) B → Broadcast: RREQ-U_B, L_S, T_S, T_B, Cert_S, Sig_S Cert_B, Sig_B

where **Sig_B: {RREQ-U_B, L_S, T_S, T_B } S_{K-B}**

This message allows any receiving nodes to securely identify replayed messages and to be assured that *B* was the sender of the message. It securely places *B* in a possible route between *S* and *D*. Note that the only different between *RREQ-U_S* and *RREQ-U_B* is the latter has had *B's* address added to the route list. This *RREQ-U_S* can be reconstructed by any subsequent recipient and the signature Sig_S checked. Note that if B had possessed an active route to *D* when it received the *RREQ* it could have created this message at the time, as nodes store the route lists and signatures associated with active routes.

If a node receives the same *RREQ* messages from two different intermediate nodes but with the same *ID* (which is possible as many different routes may exist in the network) it discards the later arrivals, according to the existing DSR definition.

This process (of addition of node addresses to the route list and signatures to the message) continues until a route to destination *D* is found, if such a route exists. Assuming the route from *S* to *D* the following message (3) would be sent to the penultimate node in the route (here *C*) and eventually back through the route to *S*. If the route is uni-directional the following message is piggybacked onto the route discovery from *D* to *S*.

(3) D → C: RREP-U_D, RREQ_ID, L_D, T_D, Cert_S Sig_S, Cert_B, Sig_B, Cert_C, Sig_C, D, Cert_D, Sig_D

where **Sig_D: {RREP-U_D, RREQ_ID, D, S, C, L_D, T_D} S_{K-D}**

This message allows *D* to authenticate itself to the nodes in the route as well as, eventually, to the source *S*. *D* adds its certificate and signature to those of *S*, and all intermediate nodes (such as *B*). This will, in part, enable *S* to learn the identity of all nodes along the route. The signature of *D* and the inclusion of *RREQ_ID* (which is not included in *RREP-U_D*) binds the reply to the initial request. The lifetime (L_D) gives a validity period for the reply. If a receiving node finds that the lifetime period L_D has expired then it must discard this message. As receiving nodes may use this reply to confirm the path back to the sender, the lifetime L_D could be relatively long. The timestamp (T_D) and *RREQ_ID* enable receiving nodes to detect replays. Note that there is sufficient information above for the route request messages to be reconstructed and the signatures of the intermediate nodes verified.

Upon receiving such a message an intermediate node checks the signature and for replays. The public key of *D* is obtained from the certificate of *D*. It can use the contents of the message to determine which request this is a reply to and to which node the reply should now be sent. For bi-directional links that are traversed in the return of the *RREP* message we can add extra security by allowing the nodes at the receiving end of such links to check additional timestamps and make a judgement on their reliability. Any timestamps judged too old are considered to reflect nodes attempting to hide their presence and the reply is discarded. The gap between timestamps (both forward and reverse) is also checked by *S* as part of the assessment of the route. This is especially important if the route contains any uni-directional links. If the sender does not want the intermediate node to make such checks it can set a flag in the *RREQ*, which should be copied y *D* into the *RREP*. A sample of the return message follows:

(4) C → B: RREP-U_D, RREQ_ID, L_D, T_D, T_{C1}, Cert_S, Sig_S, Cert_B, Sig_B, Cert_C, Sig_C, D, Cert_D, Sig_D, Sig_{C1}

where **Sig_{C1}: {RREP-U_D, RREQ_ID, C, T_{C1}} S_{K-C}**

The purpose of the message (5) is for the node C to authenticate itself to its previous hop B and to correlate the request message forwarded by B with its response thereby preventing replay attacks from happening. It also verifies if the reverse path taken by the message indeed matches with the original forward path and that no other node is either masquerading as any legitimate node in the route list or replaying a previously captured message. The signed element includes the original request and the response id. This message is signed using the private key of C.

If, in our example, during packet transmission the node *C* is unable to reach *D*, *C* must send a *RERR* message to the source *S*. This message is unicast to *S*, either via *B* or through some other route in the link *B-C* is not bi-directional.

(5) RRER-U_C, T_C, Cert_C, Sig_C

where **Sig_C: { RRER-U_C, T_C } S_{K-C}**

VI. Comparison with other Approaches

In [2] the design and evaluation of Ariadne, a new ad-hoc routing protocol is described that provides security against one compromised node and arbitrary active attackers, and relies only on efficient symmetric cryptography. The design is based on the basic operation of the DSR protocol. This scheme, like our prosed model, enables the target to verify the authenticity of a route request and also provides a mechanism for the initiator to verify each node along the path to the destination. However, unlike our approach, this scheme does not seamlessly integrate with DSR as it requires a careful redesign of each protocol message and its processing. It also requires all end-to-end communicating source and destination pair nodes to share a set of MAC keys prior to the route discovery process. Our scheme does not need this unrealistic key set-up process. The scheme in [10] proposes a Secure Routing Protocol (SRP) that counters malicious behaviour that targets the discovery of topological information. In essence this work secures against non-colluding adversaries and unlike our approach, does not aim to authenticate intermediate nodes that forward "route Requests', and thus do not handle authorisation.

In [7], a security extension to the on demand ad hoc routing protocol is proposed with a focus on Ad Hoc On Demand Distance Vector Routing (AODV) protocol. Unlike our scheme, the lifetime field is not very strongly authenticated at intermediate nodes. As intermediate nodes are allowed to reply to *RREQs*. they can lie about the lifetime. In our scheme the intermediate node must use a lifetime supplied, and signed, by the ultimate destination. This approach also only identifies, and has signatures from the original source of a message. Our scheme also identifies the immediate re-sender of *RREQ* and *RREP* messages, thus allowing better protection against replay and denial of service attacks. The scheme in [7] is also weaker against replay attacks due to the lack of timestamps in the message extensions. This will make it harder for nodes to detect replays in that scheme. Finally, [7] does not provide a route list in the *RREP*. Without this route list it is impossible for a sender to make any judgements about a returned route.

The approach proposed in [9] to secure ad hoc on demand routing protocol used a challenge-response mechanism. A three-way communication occurs between every pair of intermediate nodes increasing the overheads considerably and assuming bi-directionality, which we do not. This also implies that the basic functionality of the protocol has been modified to introduce security, which we have avoided. We have also added security to the *RERR* messages, an aspect ignored in [9]**.**

VII. Conclusion

We have proposed a security framework by applying security to the existing control messages in DSR. These protocols meet our previously identified design goals. Finally, we compared our solution with other prominent approaches that aim to secure DSR. Our approach seamlessly integrates into the existing DSR framework without impinging upon the protocol functionality, including, particularly, allowing uni-directional links.

References

[1] J. Broch, D. A. Maltz, D. B. Johnson, Y-C. Hu & J. G. Jetcheva. A Performance Comparison of Multi-Hop Wireless Ad Hoc Network Routing Protocols. In Proceedings of 4th ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom' 98), pp 85-97, October 1998.

[2] Y-C. Hu, A. Perrig, & D. B. Johnson. Ariadne: A secure on-demand routing protocol for ad hoc networks. In The 8th ACM International Conference on Mobile Computing and Networking, September 2002.

[3] Y-C. Hu, A. Perrig, & D. B. Johnson. Rushing Attacks and Defense in Wireless Ad Hoc Networks. Technical report TR01-384, Department of Computer Science, Rice University, June 2002.

[4] P. Johnson, T. Larsson, N. Hedman, B. Mielczarek, & M. Degermark. Scenario based performance Analysis of Routing Protocols for Mobile Ad Hoc networks. In the Proceedings of the Fifth Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom' 99), pp 195-206, August 1999.

[5] D. B Johnson, D. A. Maltz, & Y-C. Hu The Dynamic Source Routing Protocol for Mobile Ad Hoc Networks. Mobile Ad Hoc Networking Group Internet Draft: draft-ietf-manet-dsr-09.txt

[6] D.A. Maltz, J. Broch, J. Jetcheva, & D.B. Johnson. The Effects of On-Demand Routing Behaviour in Routing Protocols for Multi-Hop Wireless Ad Hoc networks. IEEE Journal in Selected Areas in Communications, 17(8): 1439-1453, August 1999.

[7] P. Papadimitratos & Z.J. Haas. Secure Routing for Mobile Ad-hoc Networks. In Proceedings of the SCS Communication Networks and Distributed Systems Modelling and Simulations Conference (CNDS 2002), San Antonio, TX, January 27-31, 2002.

[8] C.E. Perkins & E. M Royer. Ad Hoc on-Demand Distance Vector Routing. In second IEEE Workshop on Mobile Computing systems and Applications (WMCSA' 99), pages 90-100, February 1999.

[9] L. Venkatraman & D. P. Agrawal. An Optimised Inter-Router Authentication Scheme for Ad Hoc networks. Proceedings of the 13th International Conference on Wireless Communications, 9-11 July 2001, Calgary, Canada, pp. 129-1

[10] M. G. Zapata. Secure Ad Hoc On-Demand Distance Vector (SAODV) Routing. Mobile Ad Hoc Networking Group Internet Draft: draft-guerrero-manet-saodv-00.txt. October 2001.

[11] P. Zimmermann. The Official PGP User's Guide. MIT Press, 1995.