Securing MAODV: Attacks and Countermeasures

Sankardas Roy V. Gopala Addada Sanjeev Setia Sushil Jajodia
Center for Secure Information Systems, George Mason University, Fairfax, VA 22030

***Abstract*— Most of the multicast routing protocols proposed for ad hoc networks assume a trusted, non-adversarial environment and do not take security issues into account in their design. In this paper, we investigate the security of MAODV (Multicast Ad hoc On-Demand Distance Vector protocol), a well-known multicast routing protocol, and identify several attacks on it. We show, via simulation, that these attacks can have a significant impact on the performance of MAODV. We present an authentication framework for MAODV and propose countermeasures that can prevent or mitigate the impact of these attacks.**

I. Introduction

Many important applications for ad hoc networks are group-oriented in nature, and can therefore benefit from a multicast communication service. Example applications include mobile conferencing outside the office, battlefield communications, and disaster recovery operations. Many multicast routing protocols [5] that support group-oriented communication have been described in the literature.

For the most part, these protocols assume a trusted, non-adversarial environment and their design does not take security issues into account. Although, several articles have investigated security issues for unicast routing protocols for ad hoc networks, there has been relatively little attention paid so far to securing *multicast* routing protocols for ad hoc networks. In this paper, we take a first step towards securing multicast routing protocols for ad hoc networks. Specifically, we examine the security vulnerabilities of MAODV [26], a well-known protocol that is representative of tree-oriented multicast routing protocols for ad hoc networks [19].

Security attacks on routing protocols can loosely be classified into two categories: insider attacks and outsider attacks. Outsider attacks are launched by unauthorized nodes that do not possess the credentials to join an ad hoc network, whereas insider attacks are launched by compromised nodes that are part of the ad hoc network. Many outsider attacks can be prevented by requiring all communication to be authenticated, thereby preventing nodes that do not possess the requisite cryptographic credentials from being able to inject messages into the network with the goal of disrupting the functioning of the routing protocol. This approach, however, is not sufficient to prevent insider attacks, since the attacker possesses all the cryptographic material of the compromised node(s).

In mobile ad-hoc networks, nodes are more vulnerable to capture than in traditional networks with a fixed infrastructure. Even tamper-proof hardware cannot prevent critical keying material of a compromised node from being revealed to the attacker. Hence, in the design of a secure multicast routing protocol, it is necessary to take into account the possibility of malicious insiders, and to develop security mechanisms that can prevent attacks by such insiders. Previously, Ning *et al.* [20] have presented a systematic study of insider attacks on AODV [22]. To the best of our knowledge, however, the impact of insider attacks on MAODV has not been studied.

In securing tree-oriented multicast protocols such as MAODV, there is an additional complication in that the data delivery tree set up by the protocol includes some nodes that are not members of the multicast group. However, these nodes execute the routing protocol and participate in the forwarding of data and routing control packets. Thus, we need to consider the impact of attacks on MAODV by such nodes, in addition to attacks by compromised group members and by non-members that are not part of the multicast tree.

This paper makes the following contributions:

- We assess the vulnerability of MAODV to attacks launched by both insider and outsider nodes. In particular, we identify attacks on multicast tree formation and maintenance that have no counterpart in unicast routing protocols.
- We describe an authentication framework that can be used for preventing or mitigating the security attacks on MAODV. We show, via a detailed simulation, that without the countermeasures enabled by our authentication framework, security attacks can result in a significant degradation in the performance of MAODV.

The rest of this paper is organized as follows. The next section presents the related work. In Section III, we provide an overview of MAODV protocol. In Section IV, we describe several security attacks on MAODV. Next in Section V, we present an authentication framework for securing MAODV, and describe countermeasures against the attacks described in Section IV. Section VI contains the results of simulation experiments that show the impact of the attacks on the performance of MAODV. Finally, Section VII concludes the paper.

II. Related Work

Several researchers [13], [3], [14], [36], [17] have addressed the problem of secure group communication in ad hoc network, and proposed schemes for establishing and maintaining a group key that is used for encrypting and authenticating data that is multicast to the group. Several protocols have been proposed for efficiently rekeying the group when its membership changes. These papers do not address the challenges in securing multicast routing protocols, which is our focus in this paper.

Cordeiro et al. [5] present a survey of the multicast routing protocols for ad hoc networks that have been proposed in the literature. They classify multicast protocols for ad hoc networks into four broad categories: (i) tree-based protocols such as AMRIS [31] and MAODV [26], (ii) mesh-based protocols such as ODMRP [15] and CAMP [16], (iii) stateless multicast protocols such as DDM [11], and (iv) hybrid protocols such as AMRoute [2]. Most of these protocols assume a trusted, non-adversarial environment and their design does not take security considerations into account. In this paper, we focus on the security issues in MAODV, which falls in the category of tree-based protocols.

In recent years, there has been a great deal of research on securing unicast routing protocols for ad hoc networks. Some of this work e.g., SAR [32], ARIADNE [8], SEAD [7], ARAN [27], SAODV [33] has focussed on adding security mechanisms to previously proposed unicast routing protocols such as AODV and DSR. Other researchers have focussed on designing new secure routing protocols [21], whereas some researchers [35], [10], [1] have proposed general security mechanisms for routing protocols.

An existing body of work that is closely related to this paper addresses the security of AODV, which is the unicast counterpart of MAODV. A number of attacks on AODV have been proposed in the literature [6], [27], [30]. Ning et al. [20] propose a taxonomy of attacks on the AODV protocol. Our authentication framework described in Section V adapts some security mechanisms used in ARAN [27] and SAODV [33] for securing MAODV.

Intrusion detection can also be used to guard against attacks on routing protocols in mobile ad hoc networks. Vigna et al. [30] present a tool aimed at real-time detection, which utilizes misuse detection techniques to reduce the number of false positives. Zhang et al. [34] present an intrusion detection technique that uses cooperative statistical anomaly detection techniques. Marti et al. [18] propose that each node use a component called watchdog to detect malicious nodes. Based on the information collected by the watchdog, each node uses another component called pathrater to choose a reliable route.

III. An Overview of MAODV

MAODV is a multicast routing protocol for ad hoc networks that dynamically constructs a shared multicast tree which connects the group members possibly via some non-member nodes. MAODV also allows a non-member node, which may or may not be on the tree, to multicast data to all the group members.

MAODV is the multicast extension of Ad hoc On-Demand Distance Vector (AODV) routing protocol, and it shares many similarities and packet formats with AODV. The Route Request (RREQ) and Route Reply (RREP) packet formats are similar to those used in AODV. In addition to these packets, MAODV uses two routing control packets – Multicast Activation (MACT) and Group Hello (GRPH) packets.

MAODV maintains a sequence number for the multicast group that is initialized by the group leader and increased periodically. The primary responsibility of the group leader is to periodically broadcast Group Hello (GRPH) messages across the network and to maintain the group sequence number. A GRPH message contains the group sequence number, multicast group address and corresponding group leader IP address. The sequence number is used to ensure that routes discovered to the multicast group are always the most current ones available. Given the choice between two routes to a multicast tree, a requesting node selects the one with greater sequence number.

We now describe the route discovery and multicast tree maintenance operations of MAODV.

A. Route Discovery and Link Activation

When a node either wants to join the multicast group or find a route to the multicast group, the node broadcasts a RREQ message. For join requests, a reply is sent when the RREQ reaches a node that is already a member of the multicast tree, and the node's record of the multicast group sequence number is at least as great as that contained in the RREQ. For non-join requests, any node with a current route to the multicast tree may respond to the RREQ.

The route to the multicast tree is made available by unicasting a RREP back to the source of the RREQ. Since each node receiving the request caches a route back to the source of the request, the RREP can be unicast back to the source from any node able to satisfy the request. In case of join requests, after waiting for a specified period to receive RREPs, the requester node selects the best route to the multicast tree and unicasts a MACT message (multicast activation), with J (join) flag set, to the next hop which is on the selected route. This message officially grafts the selected route onto the existing multicast tree.

B. Multicast tree maintenance

Multicast tree maintenance mainly involves three operations – (i) Tree Pruning, (ii) Link Repair, and (iii) Partition Merging.

a) Tree Pruning: If a tree-node, which is not a member of the multicast group, becomes a leaf node, it prunes itself from the tree. It sends a MACT message, with P (prune) flag set, to its next hop. The next hop, on receiving this MACT message, deletes the entry for the sender node from its multicast route table. If this node is itself not a member of the multicast group, and if the pruning of the other node has made it a leaf node, it can similarly prune itself from the tree.

b) Link Repair: Each node on the multicast tree always monitors the status of the links with its immediate neighbors. When a link breakage is detected, the node downstream of the break (i.e., the node that is further from the multicast group leader) is responsible for repairing the broken link. The downstream node (say A) initiates the repair by broadcasting a RREQ with J (join) flag set. This RREQ packet includes the distance of A from the group leader. The only nodes that reply to this RREQ are nodes that are at least as close to the group leader as the node A. After the link is repaired A may find that its hop count from the group leader, or the group sequence number has been changed. In that case A sends a

MACT message with U (update) flag set to its downstream nodes with the updated information.

If node A does not receive any reply, it takes the following actions. If it is a group member, it becomes the new group leader. If it is a non-member and it has only one downstream neighbor then it prunes itself from the tree. If it has more than one downstream neighbors, it sends a MACT packet, with G (group leader) flag set, to one of these neighbors. This process continues till a group member is reached and it becomes the group leader.

c) Partition Merging: Nodes in one partition of the multicast tree will discover another partition when they receive Group Hello packets from the group leader of another partition. These two partitions should be merged to improve the group connectivity. The multicast group leader with lower IP address (say GL1) initiates the merging process [25]. GL1 sends a RREQ with J (join) and R (repair) flags set to the group leader of the other partition (say GL2). Upon receiving the RREQ, GL2 takes the larger of its own and the received multicast group sequence numbers, increments this value by one, and sends a RREP with R flag set to GL1. As the RREP is propagated back to GL1, new links are grafted to connect these two trees. The next time GL2 broadcasts a GRPH, it sets the U (update) flag to indicate that there is a change in group leader information and the nodes of the other partition update their group leader address and group sequence number.

IV. Attacks

MAODV does not include any provisions for security; thus, it is susceptible to attacks by outsiders as well as malicious insiders. Many attacks on routing protocols for ad hoc networks have been described in the literature. Attackers may drop, modify, replay or fabricate routing messages [6]. Nodes may also impersonate other nodes [27] while sending fabricated messages. Further, multiple attacker nodes may collude to launch attacks, e.g. wormhole attacks [9].

In general, attacks on MAODV can be divided into two categories: (i) attacks on route discovery and establishment, (ii) attacks on multicast tree maintenance. The route discovery and establishment protocols for MAODV are similar to the protocols used in AODV. Thus, the attacks on these protocols in MAODV are similar to the attacks on AODV that have been discussed in the literature. In contrast, the attacks on the multicast tree formation and maintenance in MAODV have no counterpart in unicast routing protocols.

We describe below several attacks on the operation of MAODV. Each attack has a two-part name - the first part states which message (e.g., RREP) or which property (e.g., group leadership) is misused to launch the attack, and the second part indicates the outcome of the attack, e.g., partition in the multicast tree. For brevity, group leadership is abbreviated as GL, partition in the multicast tree is abbreviated as PART, invalid route as INV and multicast tree formation as MTF. To launch an attack, if the message misused (e.g., MACT) includes a special flag (e.g., Join flag J), the message name includes the flag in parentheses (e.g., MACT(J)).

In the following discussion, we frequently use some terms which we define here. The term tree-node refers to a node that is on the shared multicast tree, and non-member refers to a node which is not a member of the multicast group. We assume that there is only one multicast group.

A. Attacks against Route Discovery and Establishment

1) Attacks against Route Discovery:

a) **RREP-INV:** When a node either wants to join a multicast group or find a route to a multicast group, it broadcasts a RREQ message. In this attack, a malicious node replies to the RREQ message with the goal of deceiving the node into believing it has found the best route to the multicast group.

Consider a group member A that wants to join the multicast group. A broadcasts a RREQ packet with the multicast group address as the destination address and with the J (Join) flag set. Only nodes on the multicast tree qualify to send a reply (RREP) to this request. However, a malicious node M can respond to the RREQ packet with a RREP even if it is not on the multicast tree. A RREP packet includes the replying node's view of the group sequence number. Since A is likely to receive RREPs from multiple nodes, in order to increase the chances of the route to M being selected as the best route to the multicast group, M can fabricate the sequence number field in its RREP.

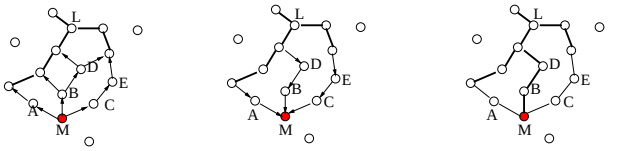
This attack can be launched by a non-tree-node or by a tree-node. If a non-tree-node succeeds in deceiving the joining node into believing it has a route to the multicast group, it can discard any future messages sent by the node to the group, effectively negating the work done by the route discovery protocol. A malicious tree-node's motivation for launching this attack is more subtle. A tree node that succeeds in becoming a root of many branches of the multicast tree can control the delivery of multicast data to all the nodes in these branches. It can therefore have a potentially large impact on the operation of the application that is using the multicast group.

2) Attacks against Link Activation:

a) **MACT(J)-MTF:** A node (say M) may receive replies in response to its route request from more than one node. In normal MAODV operation, M will select the best route and send a MACT packet, with J flag set, to graft the corresponding edge onto the tree. If M is malicious, it may select more than one route, which will result in many extra edges being grafted on to the multicast tree, i.e., creating a mesh instead of a tree.

This attack is illustrated in Figure 1. Fig. 1a shows a node M broadcasting a RREQ request. The current multicast tree is highlighted by bold lines. Fig. 1b, shows the route replies sent to M. Suppose node M is malicious. If M sends MACT packets to all the nodes (A, B and C) from which it received RREPs, then many unnecessary branches will be grafted on the multicast tree, connecting M to the existing tree. In Fig. 1c, the thin lines represent the unnecessary edges. Note that, to launch this attack, M does not need to be the initiator of RREQ; it may be simply an intermediate node which rebroadcasts the RREQ.

This attack is an instance of a resource consumption attack, since it will result in unnecessary packet duplication and energy expenditure. Note that in this attack a single malicious node can increase the energy expenditure of many other nodes.



(a) RREQ Propagation (b) RREP Propagation (c)new multicast tree

Fig. 1. Creating an energy ineffi cient multicast tree – the malicious node M unnecessarily grafts node A, C, and E on the tree.

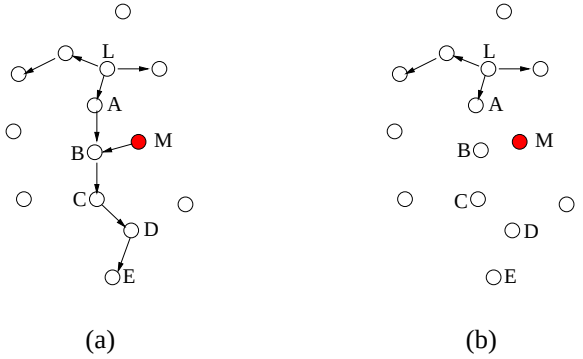


Fig. 2. MACT(P)-PART attack (a) A malicious node M sends a MACT(P) i.e., prune message to B, impersonating A. (b) Non-members B, C, D prune themselves from the tree, and the group member E gets partitioned from other part of the tree.

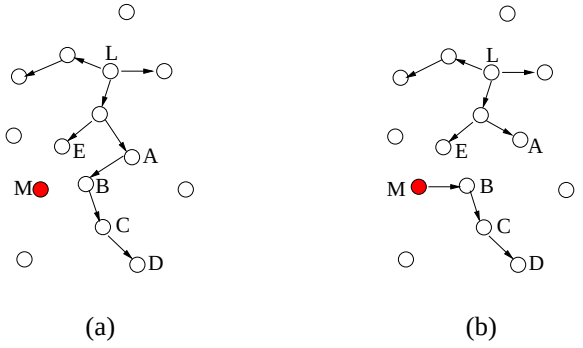


Fig. 3. RREP-PART attack (a) The link between A and B breaks and B starts broadcasting RREQ. (b) A malicious node M, sends a RREP and B, C, D get partitioned from other part of the tree.

B. Attacks on Multicast Tree Maintenance

Each of the tree maintenance operations, i.e., tree pruning, link repair, and merging partitions, is vulnerable to attacks by malicious nodes.

1) Attacks on the tree pruning process:

a) **MACT(P)-PART:** In this attack, a malicious node impersonates a tree node and sends a MACT(P) packet, i.e., a prune message, to the tree node's children in the multicast tree. If a downstream node is a non-member and has only one downstream link, it also prunes itself and sends a similar prune message to its downstream node. This may lead to the multicast tree being partitioned as explained below.

As an example, consider the multicast tree shown in Fig. 2a. Suppose that B, C, D are non-members whereas A, E are group members. The attacker M impersonates A and sends a MACT packet with P flag set to A's immediate downstream node B (Fig. 2a).

Since B is not a group member, it prunes itself from the tree by forwarding the MACT packet to its downstream node. In this example, each of B, C, D will prune itself and E will become another group leader (Fig. 2b). Note that only one forged MACT packet causes many nodes to prune themselves, making a partition in the multicast tree, at least temporarily. Node E will be isolated from the rest of the multicast tree, at least until partition merging can occur.

2) Attacks on the link repair process:

a) **RREP-PART:** When a node's link to its upstream node in the multicast tree breaks, it attempts to repair the link by broadcasting a RREQ packet with the J (join) flag set. Note that a similar procedure is followed when a node wants to join a multicast group as discussed earlier, and thus a similar attack can be launched.

Fig. 3 illustrates this attack. Fig. 3a shows a multicast tree with L acting as the group leader. Suppose M is a malicious node. Suppose that the link between A and B breaks. Following MAODV, B starts route discovery by broadcasting RREQ packets with the multicast group address as the destination address. B's RREQ packets will include the group sequence number and its hop count from the group leader L, which is equal to three. Only tree-nodes with current group sequence number greater or equal to the sequence number indicated in the RREQ packet and whose hop count from L is at most the hop count indicated in the RREQ packet should respond with a RREP to this request.

However, a malicious node M, even if it is not a tree-node, may respond with a RREP message with sequence number higher than the current group sequence number and with a false hop count that is smaller than three. This results in node B accepting M as its upstream node as shown in Fig. 3b. Even if B receives an RREP from E, it will select M because M has larger sequence number and smaller hop count. Thus nodes B, C and D get partitioned from other group members by M.

b) **MACT(J)-MTF:** The link repair operation involves the grafting of new edges on the multicast tree. This operation is similar to link activation during route discovery and establishment. Hence, the same attack can be launched during the link repair process.

c) **MACT(U)-PART:** After repairing a link, it is possible that the node that initiated the repairing process is now at a different distance from the group leader than it was before the link repair. In this case, it informs its downstream neighbors of their new distance from the group leader via a MACT message with U (update) flag set, and the hop count field set to the node's new distance from the group leader. The downstream nodes repeat this process by sending MACT messages with U

flag set to their downstream nodes.

In this attack, the attacker impersonates a tree node and sends false information to its downstream nodes, which makes them decrease their hop count from the group leader. This may lead to the creation of a loop in that branch of the multicast tree and may isolate that branch as shown in the following example.

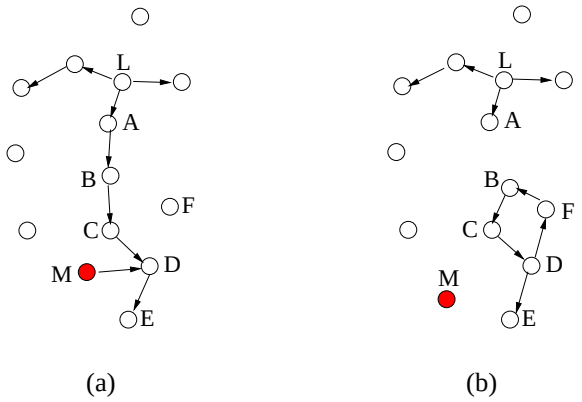


Fig. 4. The MACT(U)-PART attack (a) M sends a false MACT(U) message to D impersonating C, which makes D decrease its hop count from the group leader. The link between A and B breaks and B starts broadcasting RREQ. (b) D sends back RREP via F and a loop DFBC is created.

We present an instance of this attack with the help of Fig. 4. The attacker impersonates node C and sends a false MACT packet, with update flag U set, to node D with hop count from the group leader less than the correct one (say one instead of three). On receiving this message, D updates its hop count field. Now if the link between A and B breaks, node B will try to repair the link by broadcasting a RREQ packet. Node D qualifies to send node B an RREP and the RREP can propagate possibly through an intermediate node F. So there is a possibility that the loop $DFBC$ will be created after the route repair and the nodes B, C, D, E, F will be isolated from other members of the multicast group. Note that no node in this isolated branch will initiate a route repair because every node thinks that it is connected with its upstream node.

3) Attacks on the partition merge process:

a) **GL-PART:** In MAODV, a group leader is responsible for broadcasting Group Hello packets, and taking the necessary steps for reconnecting two partitioned trees when connectivity is restored. If group leader (say M) detects that two partitions are within communication range, it needs to take one of the following actions: (i) if M has a lower IP address than that of the other leader (say N), M unicasts a RREQ packet to N, with R (repair) and J (join) flag set, (ii) otherwise, after receiving the RREQ packet from N, M unicasts a RREP packet (with R flag set) towards N. If M is malicious, it may not perform these actions and the two partitions will remain disconnected.

We note that in the absence of authentication, any malicious node (say M) can become a group leader simply by broadcasting GRPH packets. If node M's GRPH packets include a (potentially fabricated) IP address that is higher than the IP address of the current group leader (say N), M will replace N as the group leader of the nodes in its tree.

V. Securing MAODV

In this section, we describe an authentication framework for securing MAODV from the attacks described in the previous section. We also describe additional countermeasures that are needed to mitigate some insider attacks that cannot be prevented through the use of authentication mechanisms.

A. Authentication Framework

1) Design Goals: We distinguish between three types of nodes that can launch attacks on the operation of MAODV. First, attacks can be launched by outsiders, who do not possess the credentials to join the ad hoc network. At the other extreme, attacks can be launched by compromised group members. Insider attacks are very difficult to prevent since the attacker possesses the credentials to participate in all the operations of the protocol. The third category of attackers includes non-member nodes that possess the credentials to join the ad hoc network but are not members of the multicast group under consideration. In MAODV, such nodes can participate in a subset of the operations of the protocol. For example, a non-member can become a tree node and participate in the routing of packets. However, a non-member tree node cannot become a group leader.

Given these three categories of attackers, our authentication framework has three objectives. First, an unauthorized node should not be able to participate in the MAODV protocol. Second, a non-member node should not be able to impersonate a group member. The third goal is to design an efficient authentication scheme that prevents any node which is not on the multicast tree from impersonating a tree node.

2) Authentication Mechanisms: To achieve the goals outlined above, we propose the use of an authentication framework in which nodes need the appropriate credentials to participate in the MAODV protocol as a group member or tree node. The routing control messages exchanged between nodes are augmented to include additional fields that allow the receiving node to verify the authenticity of the message. The different elements of our authentication framework are described below.

1) Each authorized node (group members as well as non-members) in the network possesses a public/private key pair and a certificate signed by a Certification Authority (CA), which can be verified by all nodes. This certificate binds a node's public key with its IP address. We refer to this certificate as a **node certificate**. Only nodes that possess a node certificate are eligible to participate in routing.
2) A group member has an additional **group membership certificate** that proves that the certificate holder belongs to a particular multicast group. This certificate binds the group member's public key and IP address with the IP address of the multicast group. When a node sends a routing control message that only group members

are entitled to send, it includes its group membership certificate. Thus non-members cannot impersonate a group member.

A node can obtain its certificate(s) off line before it joins the network or by some out-of-band communication with the CA. We assume that there is a single CA and that its public key is known to all the nodes in the network; thus, there is no need to contact a CA for verifying the credentials of a node.

3) In MAODV, a non-member node joins the multicast tree if it is needed for the tree construction at that point of time, and it prunes itself from the tree when its presence on the tree no longer improves group connectivity. To distinguish tree nodes from other nodes, all current tree nodes are given a credential, which we refer to as the **tree key**. In Section V-A.3, we describe how the tree key is securely disseminated to all current tree nodes by the group leader, and how nodes that want to join the group can verify that node replying to their route request possesses the tree key. The tree key is periodically refreshed so that only nodes that are currently on the tree will possess a valid tree key.
4) A node on the multicast tree establishes **pairwise shared keys** with each of its immediate neighbors. This can be done using the public keys of the two nodes, as in the SSL Handshake Protocol [29]. All messages exchanged between neighboring tree nodes include a MAC computed using this pairwise key to provide strong source authentication, and prevent impersonation attacks.
5) The Group Hello packets broadcast by a group leader are **digitally signed** for authentication. Alternatively, a lightweight broadcast authentication scheme such as TESLA [23] can be used. In fact, since the contents of a Group Hello message will be known in advance to the group leader, the variant of TESLA that allows immediate authentication [23] can be used. In the rest of this paper, however, we assume that digital signatures are used for authenticating GRPH packets.

3) Secure Tree Key Dissemination & Verification: An important element of our authentication framework for MAODV is the use of the tree key credential for distinguishing between tree nodes and other nodes. Our approach for efficiently and securely distributing the tree key to current tree nodes takes advantage of the multicast tree already set up by MAODV, and the pairwise keys established between neighboring nodes in the multicast tree.

In our framework, it is the responsibility of the Group Leader to periodically distribute a new tree key to current tree nodes. Starting with the group leader, each tree node sends the tree key separately to each of its downstream neighbors after encrypting it using the pairwise key shared with that neighbor[1]. This procedure is repeated recursively down the multicast tree until all the leaf nodes have received the tree key.

[1]Note that separate pairwise keys are established for encryption and authentication.

In addition, the Group Hello packets broadcast by the group leader include the tree key authenticator $f(tree\ key)$, where f is a one-way function. Nodes that receive the tree key can verify its authenticity by applying the one-way function f to it and comparing the result against the authenticator included in the latest GRPH packet. This approach is used both by tree nodes to verify the authenticity of a new tree key received from its upstream neighbor in the multicast tree, and by nodes that receive a RREP in response to a route request to verify that the replying node is a tree node.

4) Hop Count Authentication using One-way Hash Chains: Another important consideration in the design of our framework is the need for a mechanism that enables a tree node to verify its distance from the group leader (Recall that the RREP-PART attack on MAODV discussed in Section IV, involves a malicious node misrepresenting its distance in hops from the group leader). For enabling each node to verify its distance from the GL, we adapt the technique based upon the use of one-way hash chains that was proposed by Zapata [33].

In this approach, the group leader uses a one-way function H to generate the one-way hash chain $k_{N_{GL}}$, $k_{(N-1)_{GL}}$, $k_{(N-2)_{GL}}$, ..., $k_{0_{GL}}$ for each tree key refreshing interval. Here N is the diameter of the network. The last element of the hash chain $k_{0_{GL}}$ is referred to as the hop count anchor and is included in the Group Hello packets broadcast by the group leader.

The elements of this hash chain are used as hop count authenticators as follows. The group leader disseminates $k_{N_{GL}}$, N, and its distance from itself (0) to each of its children in the multicast tree. Instead of using a separate message for the purpose, these fields are piggybacked on the periodic message sent by the GL when it refreshes the tree key. Assume that one of the GL's children in the multicast tree is A, that B is A's child, and that C is B's child in the tree. On receiving $k_{N_{GL}}$, node A verifies that $H^N(k_{N_{GL}}) = k_{0_{GL}}$, where H is the one-way function used to generate the hash chain. If the verification succeeds, A calculates $k_{(N-1)_{GL}}$, and includes this value and its hop count from GL (which is 1) in its message to its child B that contains the new tree key. In turn, after verifying that $H^{N-1}(k_{N-1_{GL}}) = k_{0_{GL}}$, B sends $k_{(N-2)_{GL}}$, and its hop count from GL (which is 2) in its message to C. This process continues recursively down the multicast tree until each node has received a hop count authenticator from its parent in the multicast tree.

This mechanism prevents a node from claiming to be at a smaller distance from the GL than its actual distance[2]. Figure 5 shows the messages exchanged between the GL and A, A and B, and B and C for disseminating the tree key and hop count authenticator. It also shows the format of the Group Hello message that includes the hop count anchor and tree key authenticator fields described above.

Finally, we note that the same hop count authentication mechanism can be used in the route discovery process to en-

[2]However, a node can claim to be one hop closer to the GL than its actual distance [7] if it simply forwards the hop count authenticator it received from its parent in the multicast tree.

$GL \rightarrow * : (grp\ seq\ num, IP_{GL}, IP_{Grp}, f(tree\ key), k_{0_{GL}}, N)_{K^-_{GL}}$
$GL \rightarrow A : MAC(K^1_{GL-A}, (grp\ seq\ num, E_{K^2_{GL-A}}(tree\ key), k_{N_{GL}}, N, 0))$
$A \rightarrow B : MAC(K^1_{A-B}, (grp\ seq\ num, E_{K^2_{A-B}}(tree\ key), k_{(N-1)_{GL}}, N, 1))$
$B \rightarrow C : MAC(K^1_{B-C}, (grp\ seq\ num, E_{K^2_{B-C}}(tree\ key), k_{(N-2)_{GL}}, N, 2))$

Fig. 5. Hop-by-hop tree key and hop count authenticator dissemination from the group leader to A, B and C. The Group Hello message is also shown. Note that K^1_{X-Y} denotes the authentication key shared by X and Y, whereas K^2_{X-Y} is the encryption key.

sure that the hop count field in a RREP packet sent in response to a RREQ reflects the actual distance of the responding tree node from the requester.

B. Authenticated MAODV operations

We now describe how the authentication framework can be used to secure MAODV against the attacks discussed in Section IV.

1) Route Discovery and Establishment:

a) Route Discovery: When a group member (say S) wants to join the multicast group, it broadcasts a RREQ packet. When a tree node (say C) receives this RREQ, it unicasts a RREP back to S possibly via some intermediate nodes. Figure 6 illustrates how the messages exchanged during route discovery are authenticated.

S broadcasts a signed RREQ packet that includes the IP address of the multicast group, the request ID, and its group membership certificate. The first node that receives the message (say A) signs the message using its private key before forwarding it. Each subsequent node (say B) that forwards the request checks the outer and the inner signatures, and if the verifications are successful, it replaces the outer signature with its own signature. This authentication approach is adapted from the approach used by ARAN [27].

When the request reaches the tree node C, it responds with an RREP to the node from which it received the request (B in this example). As shown in Figure 6, this RREP packet is signed and includes the IP address of the group, current group sequence number, an authenticator to prove that C is a tree node, a hop count authenticator, and C's node certificate.

C generates the tree node authenticator by encrypting the current tree key concatenated with its IP address using S's public key, i.e., $E_{K^+_S}(C, tree\ key)$. The requester S decrypts this field with its private key to extract the tree key. Using a recent Group Hello packet, S can verify whether C is a tree node using the tree key authenticator $f(tree\ key)$ present in the Group Hello packet.

The RREP message sent by C also includes a hop count authenticator (k_{N_C}) and hop count anchor (k_{0_C}). These fields correspond to the first and last elements of a one-way hash chain generated by C. They are used to authenticate the distance (in hops) from C of an intermediate node on the path from C to S. The scheme used for hop count authentication is identical to the approach described in Section V-A.4 for authenticating the distance of node from the group leader. This approach prevents an intermediate node on the path from C to S from inserting a false hop count in place of its actual distance from C.

The RREP message is authenticated in the same manner as the RREQ message, i.e., each intermediate node verifies an outer and inner signature, and replaces the outer signature with its own signature before forwarding the message.

b) Link Activation: To establish a route to the multicast group, the node S sends a signed MACT packet with J (join) flag set to the neighbor (say A) with the shortest distance to the multicast tree (as indicated by the hop count field in the RREP message received from A). To prove that S was the node that initiated the route discovery, the MACT packet includes a field which is computed by applying a one-way hash function H to the tree key and its IP address, i.e., $H(S, tree\ key)$. Note that only the node that initiated the route discovery could have obtained the tree key from the RREP message since the tree key is encrypted with the public key of S. When the MACT(J) message is received by the tree node that sent the RREP message (say C), it checks this field to verify that S has the credentials to activate a route to the multicast group.

Figure 7 shows how the MACT packet is authenticated at each intermediate node on the path from S to the tree node. The approach used here is identical to the approach used for authenticating the RREQ and RREP messages in the route discovery step, i.e., each intermediate node verifies the outer and inner signatures in the message, and replaces the outer signature with its own signature before forwarding the message. As the MACT(J) message is propagated to the tree node C, each link on the path is grafted on to the multicast tree. In addition, neighboring nodes on the path establish pairwise shared keys with each other with the help of their public keys. Recall that pairwise shared keys are used for securely distributing the tree key to the current nodes on the multicast tree. As discussed below, these keys are also used to authenticate routing control messages exchanged between neighboring nodes on the multicast tree.

2) Tree Maintenance: We discuss below how the tree maintenance operations, namely tree pruning, link repair, and partition merging can be authenticated.

a) Tree Pruning: If a tree-node, which is not a member of the multicast group, notices that it has become a leaf node on the multicast tree, it prunes itself from the tree after sending a MACT packet, with P (prune) flag set, to its neighbor. To prevent impersonation attacks, this message needs to be authenticated. To this end, MACT(P) messages are authenticated using a MAC computed using the pairwise keys shared between a node and its neighbor.

b) Link Repair: For the most part, the RREQ, RREP, and MACT(J) messages exchanged in MAODV's link repair

$S \rightarrow * : (JoinReq, IP_{Grp}, ReqID, gmcert_S)_{K_S^-}$
$A \rightarrow * : ((JoinReq, IP_{Grp}, ReqID, gmcert_S)_{K_S^-})_{K_A^-}, cert_A$
$B \rightarrow * : ((JoinReq, IP_{Grp}, ReqID, gmcert_S)_{K_S^-})_{K_B^-}, cert_B$
$C \rightarrow B : (JoinRep, IP_{Grp}, S, ReqID, grp\ seq\ num, cert_C, k_{0_C}, N, 0, k_{N_C})_{K_C^-}, E_{K_S^+}(C, tree\ key)$
$B \rightarrow A : ((JoinRep, IP_{Grp}, S, ReqID, grp\ seq\ num, cert_C, k_{0_C}, N)_{K_C^-}, 1, k_{(N-1)_C})_{K_B^-}, E_{K_S^+}(C, tree\ key), cert_B$
$A \rightarrow S : ((JoinRep, IP_{Grp}, S, ReqID, grp\ seq\ num, cert_C, k_{0_C}, N)_{K_C^-}, 2, k_{(N-2)_C})_{K_A^-}, E_{K_S^+}(C, tree\ key), cert_A$

Fig. 6. Authenticated Route Discovery.

$S \rightarrow A : (JoinMACT, IP_{Grp}, grp\ seq\ num\ at\ S, H(S, tree\ key))_{K_S^-}$
$A \rightarrow B : ((JoinMACT, IP_{Grp}, grp\ seq\ num\ at\ S, H(S, tree\ key))_{K_S^-})_{K_A^-}$
$B \rightarrow C : ((JoinMACT, IP_{Grp}, grp\ seq\ num\ at\ S, H(S, tree\ key))_{K_S^-})_{K_B^-}$

Fig. 7. Authenticated Link Activation

protocol can be authenticated in the same way as these messages are authenticated in the route discovery and link activation protocols. However, one key difference is that an extra hop count authenticator field in a RREP is used to verify the replying node's *distance (in hops) from the group leader*, in addition to the hop count authenticator used for distance between the replying node and the node that sent the RREQ. The approach used for hop count authentication was described in Section V-A.4.

Another difference between the link repair and route discovery protocols is that the initiator (say I) of the link repair is a tree node before the link breakage occurs. Hence, it will already possesses the current tree key. Further, it is not necessary for a tree node to be a group member. Therefore, unlike the RREQ packet sent by a node that wants to join the multicast group, the RREQ request sent by I does not include a group membership certificate. Instead, the RREQ contains a field $H(I, tree\ key)$ where H is a one-way function.

After the link is repaired, I may find that its hop count from the group leader has been changed. In that case, I sends a MACT message with U (update) flag set to its downstream nodes to pass this information. This MACT(U) packet is authenticated by a MAC computed using the pairwise key of I and its downstream node.

c) Partition Merging: The RREP, RREQ, and MACT(J) messages exchanged between two group leaders for merging their partitions can be authenticated in the same fashion as the corresponding messages exchanged during route discovery and link activation. After a route is established between the two group leaders, the new group leader broadcasts a Group Hello packet with U flag set. All the tree nodes which were previously in the partition with a different group leader update their group leader address. The current tree key is also disseminated from group leader to all the tree nodes of the merged tree.

C. Security Analysis

The security of our authentication framework is derived from the following facts – (i) Any node which does not have a valid node certificate cannot participate in routing; thus the protocol is protected from outsider attacks (ii) Only nodes that possess the latest tree key can claim to be tree nodes, (iii) No tree node can claim to have a group sequence number higher than the correct one because the group sequence number is broadcast throughout the network with a signed Group Hello packet from the group leader, (iv) A tree node cannot claim to have a hop count from the group leader less than its actual distance by more than one hop; this property is achieved with the use of the hop count authenticator disseminated from the group leader to all the tree nodes, (v) No node can impersonate as a one-hop neighbor of a tree node because any one hop communication between two neighboring tree nodes is authenticated using their pairwise key.

We now comment on the security of the authentication schemes used for different MAODV operations. In the route discovery phase, the replying node proves that it has the tree key, and the requester can verify it. The tree key sent with the RREP is encrypted with the public key of the requesting node so that only that node can obtain the tree key. In the example illustrated in Figure 6, the tree node C sends the credential $E_{K_S^+}(C, tree\ key)$ to S. Node C concatenates its IP address with the tree key before encrypting it, so that nodes other than C cannot use it in their RREP to S. In the link activation phase, C activates the downstream link only after it verifies that S received the tree key in the previous step.

A similar security analysis applies for the link repair and partition merging protocols. In the link repair protocol, the replying tree node includes in its RREP packet the appropriate hop count authenticator to prove that its hop count from the group leader is less than (not less than or equal to as required in original MAODV protocol [25]) that of the requester. Even if a tree node cheats on its hop count by one hop – a limitation of this hop count authentication scheme which uses elements from a hash chain – loop formation is avoided.

A tree node sends MACT(P), MACT(G) or MACT(U) packet only to its neighboring tree nodes. These packets are authenticated by a MAC calculated using the pairwise key of the two neighbors, preventing any attackers from impersonating a specific tree node.

D. Countermeasures for specific attacks

Many of the attacks discussed in Section IV are prevented by our authentication framework. However, some insider attacks launched by compromised nodes cannot be prevented using authentication mechanisms alone. Specifically, insider attacks such as MACT(J)-MTF and GL-PART in which a malicious node does not follow the MAODV protocol cannot be prevented by our authentication framework. In these cases, additional countermeasures based on monitoring the behavior of the nodes are necessary. We discuss below how each attack discussed in Section IV is prevented or mitigated.

(a) RREP-INV & RREP-PART: The authentication framework prevents any node not on the multicast tree from sending a valid RREP because it cannot produce the required credential, i.e., the tree key. It also stops the attacker from forging the group sequence number because any node can verify the current group sequence number with the help of the signed Group Hello packet. However, if a tree node (say M) is compromised and if it colludes with other attacker nodes, which are not on the multicast tree and which may have a wormhole [9] with M, the tree key can be leaked to the attacker nodes. In this scenario, our authentication framework cannot prevent this attack, and additional countermeasures, e.g. packet leashes [9] are needed.

(b) MACT(U)-PART: In our authentication framework, MACT(U) packets are authenticated using the pairwise key of the two neighboring tree nodes preventing an attacker from launching this impersonation attack. Even a malicious tree node cannot forge the group sequence number and hop count in the MACT(U) packet that it sends to its downstream nodes.

(c) MACT(J)-MTF: The authentication framework prevents non-tree members from launching this attack by sending false RREQ(J) packets. To prevent a malicious tree-node from launching this attack, after receiving a MACT(J) packet every node (e.g., A in Fig. 1) monitors the outgoing traffic from the sender (i.e., M) for a short time period to see if that node transmits a second MACT(J) packet activating a second route to the same multicast group. If it overhears such a packet, A prunes the link with M from its routing table. If there are only a few attackers in the network, this simple countermeasure can mitigate the effect of the attack.

(d) MACT(P)-PART: MACT(P) packets are authenticated using the pairwise key of the two neighboring tree nodes preventing an attacker from launching this impersonation attack.

(e)GL-PART: The authentication framework prevents any node which is not a group member from becoming the group leader; thus, no non-member can launch this attack. Preventing malicious group members from launching this attack is much more difficult; it requires nodes on the multicast tree to monitor their group leaders, and detect anomalous behavior.

Nodes in the two partitions that are merging can conclude that one of the two group leaders involved is misbehaving if an appropriately lengthy time interval has elapsed and the two partitions have not yet merged. In this event, it is necessary to design a recovery protocol that identifies the malicious group leader and takes corrective actions to merge the two partitions. We note MAODV includes provisions to recover from the failure of a group leader; thus, the recovery protocol will need to extend MAODV to treat a misbehaving group leader as a failed node.

VI. Performance Analysis

In this section, we report on a detailed simulation study that examined the impact of the attacks discussed in Section IV and the authentication framework described in Section V-A on the performance of MAODV. Our simulations were written using the ns-2 simulator (version 2.26) with CMU Monarch extensions. We modified the MAODV source code provided by Zhu et al. [37] to develop an attacker agent that runs on misbehaving nodes. We also implemented our proposed countermeasure for the MACT(J)-MTF attack.

A. Simulation Environment

In our simulations, we used the two-ray ground reflection model [24] to model radio propagation, the IEEE 802.11 Distributed Coordination Function (DCF) [4] as the MAC layer protocol, and the random waypoint model [12] as the node mobility model. Table I summarizes the parameters used in our simulation experiments, and Table II lists the default values of the MAODV parameters used. The node and group certificates in our simulations use a 512 bit public key and 16 byte signature as in ARAN [28].

The scenario simulated consists of a network with 50 nodes and a single multicast group with 10 members. All group members join the multicast group at the beginning of simulation leading to the construction of the multicast tree. Each simulation run corresponds to 750 seconds of simulated time. After 30 seconds, one group member starts transmitting data packets at Constant Bit Rate (CBR) flow of 2 packets per second.

For each set of experiments, we performed 30 runs whose average values are used to plot the graphs in Fig. 8. In all cases, our results have 95% confidence intervals that are within 10% of the reported value. We include error bars corresponding to 95% confidence intervals in some cases but for clarity, we do not show the confidence intervals in all the graphs.

Metrics:

Most of the attacks we have identified attempt to create partitions in the multicast tree, whereas one attack (MACT(J)-MTF) is a resource consumption attack that results in unnecessary packet transmissions. The metric used to evaluate the impact of partitioning attacks is the Packet Delivery Ratio (PDR) which is defined as the ratio of the total number of data packets received by multicast group members to the product of the number of data packets sent and the number of group members.

The impact of the resource consumption attack is measured by computing the additional number of data packet transmissions in the network after the attack is launched.

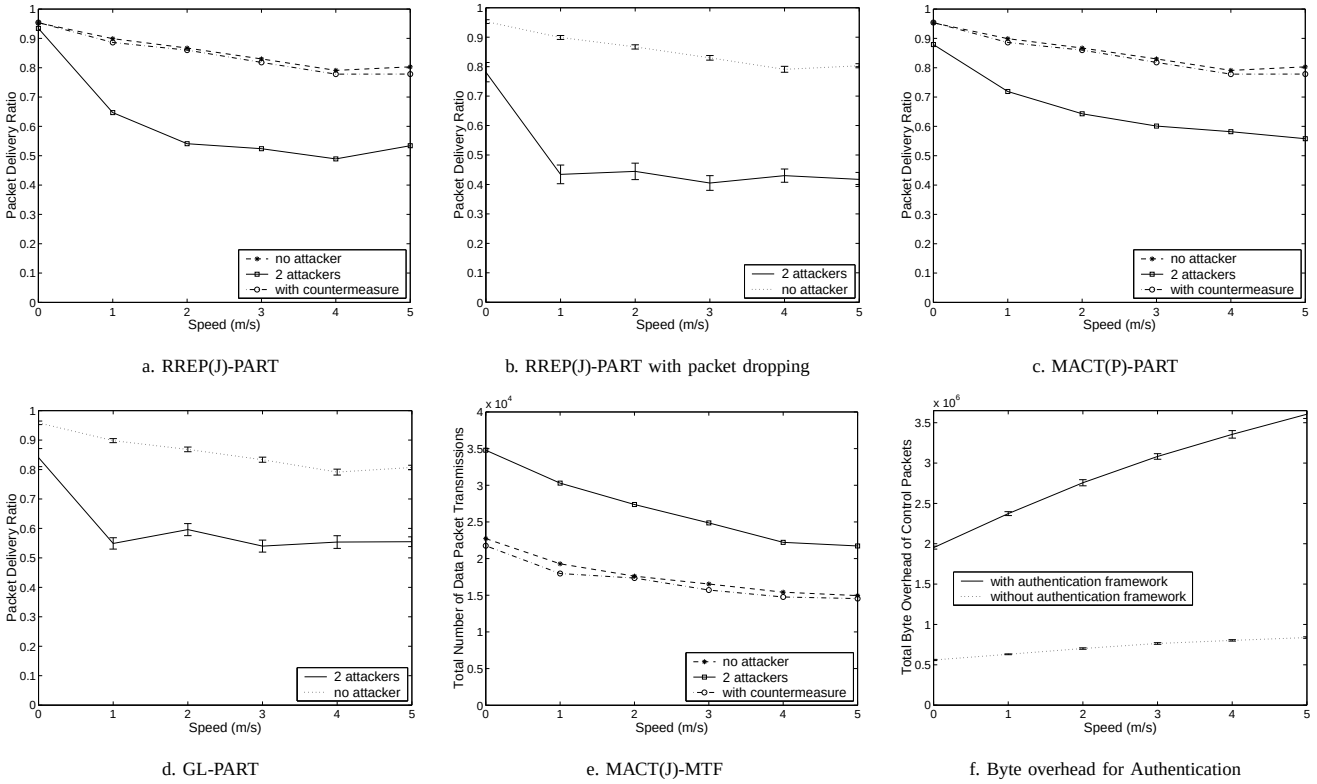


Fig. 8. Experimental Results showing effects of attacks and countermeasures, and overhead of authentication framework

TABLE I

SCENARIO PARAMETERS FOR SIMULATIONS

Number of Nodes	50
Maximum Speed (v_{max})	0 m/s, 1 m/s, 2 m/s, 3 m/s, 4 m/s and 5 m/s
Pause Time	50 sec
Simulation Grid Size	1000m$\times$1000m
Radio Transmission Range	250 m
Traffi c Source	CBR at 2 packets/second
Application Data Payload Size	256 bytes/packet
Total Application Data load	8 Kbps
Physical/Mac Layer	IEEE 802.11
Physical Link Bandwidth	2 Mbps

TABLE II

MAODV PARAMETERS VALUES

Parameter	Value
Number of Allowed Hello Loss	3
Group Hello Interval	5 secs
Hello Interval	1 secs
Time to Wait to Receive a MACT	1 sec
Lifetime of Route Table Entries	3 secs
Max Number of RREQ Retransmissions	3
Max Time to Wait for a RREP	0.5 secs

Our authentication framework also adds control overhead due to the additional control packets introduced (e.g. for tree key distribution) and the increased control packet sizes due to the additional fields for signatures and certificates. To capture this cost, we calculate the byte overhead imposed by the framework, and compare it with that of the original MAODV protocol.

B. Results and Discussion

In simulating each attack, we randomly select two nodes in the network as attackers. We do not distinguish between RREP-PART and RREP-INV attacks as both these attacks involve attacker sending false replies to RREQ messages. For attacks against link activation, tree pruning, and link repair, we use two non-members as attackers, whereas for attacks against

partition merging, we select group members as attackers.

1) RREP-PART: Fig 8a illustrates the impact of RREP(J)-PART attack on the PDR for various speeds of mobile nodes. We see that the PDR deteriorates by as much as 35% in the presence of attackers. However, the use of our authentication framework prevents this from happening with a negligible impact on the PDR. We observe that increasing the routing control packet sizes (for authentication framework) does not have any significant performance impact.

In most of our experiments, attacker nodes do not drop any data packets, i.e., if the attacker receives a multicast data packet, it forwards it according to the original MAODV protocol. However, in order to show the impact of packet dropping on throughput, we also show the PDR in a scenario where attackers launching a RREP(J)-PART attack drop the data packets they receive. Fig. 8b shows that PDR further decreases (by 47%) under RREP(J)-PART attack when attackers drop data packets.

2) MACT(P)-PART: Fig. 8c shows the effect of MACT(P)-PART attack on performance of MAODV. To implement MACT(P)-PART attack, we modelled the attacker such that whenever it overhears a Group Hello Message from a tree node A, it broadcasts a spoofed MACT(P) message with A's address as source address. We see that this attack can lead to a degradation in the PDR of as much as 25%. This attack is also prevented by our authentication framework, without any impact on the PDR as shown in Fig. 8c.

3) MACT(U)-PART: From our simulations, we found that the MACT(U)-PART attack results in very small (less than 2%) degradation in the throughput of the protocol.

4) GL-PART: We consider the impact of the GL-PART attack assuming that the attack is launched by two compromised multicast group members. Fig. 8d shows the effect of the attack on the PDR for various speeds. We observe that this attack can lead to a degradation in the PDR of as much as 35%.

5) MACT(J)-MTF: This is a resource consumption attack which places some extra nodes on the multicast tree resulting in increased multicast data traffic in the network. We modelled an attacker node that periodically broadcasts a RREQ(J) message every 25 seconds after the attack is launched. After receiving replies to the RREQ the attacker sends MACT(J) to all the nodes that sent it an RREP. We calculated the number of additional multicast data packets transmissions in the attack scenario.

Fig. 8e compares the number of data packet transmissions in the presence of 2 attackers with those before the attack. We observe that the number of data packet transmissions is increased by more than 40% when the network is under attack. The data packets transmitted by the attackers are not considered to enable a fair comparison.

We implemented the countermeasure for MACT(J)-MTF by enabling neighbors to overhear the MACT(J) transmissions as described in section V-D. Fig. 8e shows that our countermeasure does not allow the attacker nodes to increase the total number of data packet transmissions. However, the PDR of the protocol drops by about 5% in presence of our countermeasure.

6) Byte Overhead: Our authentication framework increases the byte overhead due to the increase in size of the multicast routing control packets. We computed the byte overhead for MAODV with and without our authentication mechanisms. Fig. 8f shows adding our authentication framework increases the byte overhead of MAODV by 3.5 to 4.2 times, increasing with the speed of the nodes. We believe that this is an acceptable performance cost, given that the attacks prevented have a much larger impact on the performance of the protocol.

VII. Conclusions

In this paper, we investigated the security of MAODV, which is a representative of tree-based multicast routing protocols for ad hoc networks. We identified several insider and outsider attacks on MAODV. The goal of these attacks is either to create a partition in the multicast tree or to build an energy inefficient multicast tree. Our simulation results confirm that these attacks disrupt the normal operation of MAODV to a large extent.

We also proposed an authentication framework to guard against these attacks. Our simulation shows that our countermeasures are effective. Furthermore, the countermeasures have a negligible impact on MAODV performance during the normal operation of the protocol.

As a part of our future work, we plan to further explore the impact of these attacks where multiple attackers collude with each other. We also plan to do security analysis of other types of multicast protocols e.g., mesh-based multicast protocols.

References

[1] Stefano Basagni, Kris Herrin, Danilo Bruschi, and Emilia Rosti. Secure pebblenets. In *MobiHoc '01: Proceedings of the 2nd ACM international symposium on Mobile ad hoc networking & computing*, pages 156–163. ACM Press, 2001.

[2] E. Bommaiah, A. McAuley, R. Talpade, and M. Liu. Amroute: Adhoc multicast routing protocol. IETF, Internet Draft. draft-talpade-manet-amroute-00.txt, August 6, 1998.

[3] Danilo Bruschi and Emilia Rosti. Secure multicast in wireless networks of mobile hosts: protocols and issues. *Mobile Networks and Applications*, 7(6):503–511, 2002.

[4] IEEE Computer Society LAN MAN Standards Committee. *IEEE 802.11: Wireless LAN Medium Access Control and Physical Layer Specifi cations*, 1997.

[5] C. Cordeiro, H. Gossain, and Dharma P. Agrawal. Multicast over wireless mobile ad hoc networks: Present and future direction. In *IEEE Network, special issue on Multicasting: An Enabling Technology*, volume 17, pages 52–59, jan/feb 2003.

[6] S. Gupte and M. Singhal. Secure routing in mobile wireless ad hoc networks. *Ad Hoc Networks*, 1:151–174, 2003.

[7] Yih-Chun Hu, David B. Johnson, and Adrian Perrig. Sead: Secure effi cient distance vector routing for mobile wireless ad hoc networks. *Ad Hoc Networks*, 1(1):175–192, 2003.

[8] Yih-Chun Hu, Adrian Perrig, and David B. Johnson. Ariadne: A secure on-demand routing protocol for ad hoc networks. In *Proceedings of the Eighth Annual International Conference on Mobile Computing and Networking* (MobiCom 2002).

[9] Yih-Chun Hu, Adrian Perrig, and David B. Johnson. Packet leashes: A defense against wormhole attacks in wireless networks. In *Proceedings of IEEE Infocomm 2003*, April 2003.

[10] J. Hubaux, L. Buttyan, and S. Capkun. The quest for security in mobile ad hoc networks. In *Proceeding of the ACM Symposium on Mobile Ad Hoc Networking and Computing (MobiHOC)*, 2001.

[11] L. Ji and M. S. Corson. Differential destination multicast – a manet multicast routing protocol for small groups. In *Proceedings of INFOCOM*, pages 1192–02, 2001.

[12] David B Johnson and David A Maltz. Dynamic source routing in ad hoc wireless networks. In *Mobile Computing*, volume 353. Kluwer Academic Publishers, 1996.

[13] T. Kaya, G. Lin, G. Noubir, and A. Yilmaz. Secure multicast groups on ad hoc networks. In *SASN '03: Proceedings of the 1st ACM workshop on Security of ad hoc and sensor networks*, pages 94–102. ACM Press, 2003.

[14] Thomas Kostas, Diane Kiwior, Gowri Rajappan, and Michel Dalal. Key management for secure multicast group communication in mobile networks. In *Proceedings of DARPA Information Survivability Conference and Exposition*, 2003.

[15] S.-J. Lee, W. Su, and M. Gerla. On-demand multicast routing protocol in multihop wireless mobile networks. In *ACM/Baltzer Mobile Networks and Applications, special issue on Multipoint Communication in Wireless Mobile Networks*, 2000.

[16] E.L. Madruga and J.J. Garcia-Luna-Aceves. Multicasting along meshes in ad-hoc networks. In *Proceedings of IEEE ICC'99*, pages 314–318, 2000.

[17] Silja Mäki, Tuomas Aura, and Maarit Hietalahti. Robust membership management for ad-hoc groups. In *Proc. 5th Nordic Workshop on Secure IT Systems (NORDSEC 2000)*, Reykjavik, Iceland, October 2000.

[18] Sergio Marti, T. J. Giuli, Kevin Lai, and Mary Baker. Mitigating routing misbehavior in mobile ad hoc networks. In *Mobile Computing and Networking*, pages 255–265, 2000.

[19] Prasant Mohapatra, Chao Gui, and Jian Li. Group communications in mobile ad hoc networks. *IEEE Computer*, 37(2):52–59, 2004.

[20] Peng Ning and Kun Sun. How to misuse aodv: A case study of insider attacks against mobile adhoc routing protocols. In *Proceedings of the 4th Annual IEEE Information Assurance Workshop*, pages 60–67, West Point, June 2003.

[21] P. Papadimitratos and Z.J. Haas. Secure routing for mobile ad hoc networks. In *Proceedings of the SCS communication Networks and Distributed Systems Modeling and Simulation Conference*, pages 27–31, 2002.

[22] C. Perkins. Ad-hoc on-demand distance vector routing. MILCOM '97 panel on Ad Hoc Networks, Nov. 1997.

[23] Adrian Perrig, Ran Canetti, Dawn Song, and J. D. Tygar. Efficient and secure source authentication for multicast. In *Network and Distributed System Security Symposium, NDSS '01*, pages 35–46, February 2001.

[24] Theodore Rappaport. *Wireless Communications: Principles and Practice*. Prentice Hall PTR, 1996.

[25] Elizabeth M. Royer and Charles E. Perkins. Multicast ad hoc on demand distance vector (maodv) routing. IETF Internet Draft. draft-ietf-manet-maodv-00.txt, July, 2000.

[26] Elizabeth M. Royer and Charles E. Perkins. Multicast operation of the ad-hoc on-demand distance vector routing protocol. In *Mobile Computing and Networking*, pages 207–218, 1999.

[27] K. Sanzgiri, B. Dahill, B. N. Levine, C. Shields, and E. M. Belding-Royer. A secure routing protocol for ad hoc networks. In *Proceedings of 2002 IEEE International Conference on Network Protocols (ICNP)*, 2002.

[28] K. Sanzgiri, D. LaFlamme, B. Dahill, B. N. Levine, C. Shields, and E. M. Belding-Royer. Authenticated routing for ad hoc networks. *IEEE Journal on Selected Areas in Communications*, 23(3):598–610, 2005.

[29] William Stallings. *Network Security Essentials: Applications and Standards*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 1999.

[30] G. Vigna, S. Gwalani, K. Srinivasan, E. Belding-Royer, and R. Kemmerer. An Intrusion Detection Tool for AODV-based Ad Hoc Wireless Networks. In *Proceedings of the Annual Computer Security Applications Conference (ACSAC)*, pages 16–27, Tucson, AZ, December 2004.

[31] C. W. Wu, Y. C. Tay, and C. K. Toh. Ad hoc multicast routing protocol utilizing increasing id-numbers (amris) functional specification. Internet Draft. Nov, 1998.

[32] Seung Yi, Prasad Naldurg, and Robin Kravets. Security-aware ad hoc routing for wireless networks. In *Proceedings of the ACM Symposium on Mobile Ad Hoc Networking and Computing (MobiHOC 2001)*, Long Beach, CA, October 2001.

[33] Manel Guerrero Zapata. Secure ad hoc on-demand distance vector (saodv) routing. Mobile Ad Hoc Networking Working Group Internet Draft. draft-guerrero-manet-saodv-00.txt, 12 August 2001.

[34] Yongguang Zhang and Wenke Lee. Intrusion detection in wireless ad-hoc networks. In *MobiCom '00: Proceedings of the 6th annual international conference on Mobile computing and networking*, pages 275–283, 2000.

[35] Lidong Zhou and Zygmunt J. Haas. Securing ad hoc networks. *IEEE Network*, 13(6):24–30, 1999.

[36] Sencun Zhu, Sanjeev Setia, Shouhuai Xu, and Sushil Jajodia. Gkmpan: An efficient group rekeying scheme for secure multicast in ad-hoc networks. In *1st ACM International Conference on Mobile and Ubiquitous Systems (Mobiquitous'04)*, pages 42–51, 2004.

[37] Y. Zhu and T. Kunz. Maodv implementation for ns-2.26. Technical Report SCE-04-01, Systems and Computing Engineering, Carleton University, January 2004.