A Specification-based Intrusion Detection Model for OLSR

Insha Altaf
Dept. of IT, National Institute of Technology, Srinagar, J&K, India
insha.altaf39@gmail.com

Insha Majeed
Dept. of IT, National Institute of Technology, Srinagar, J&K, India
insha333@gmail.com

***Abstract*—In this paper, we in introduce a specification based intrusion detection model for detecting attacks on routing protocols in MANETs. Intrusion detection is a viable approach to enhancing the security of existing computers and networks. Briefly, an intrusion detection system monitors activity in a system or network in order to identify ongoing attacks. Intrusion detection techniques can be classified into anomaly detection, signature-based detection, and specification-based detection. In anomaly detection, activities that deviate from the normal behavior profiles, usually statistical, are flagged as attacks. Signature-based detection matches current activity of a system against a set of attack signatures. Specification-based detection identifies system operations that are different from the correct behavior model. Our specification-based approach analyzes the protocol specification of an ad hoc routing protocol to establish a finite-state-automata (FSA) model that captures the correct behavior of nodes supporting the protocol. Then, we extract constraints on the behavior of nodes from the FSA model. Thus, our approach reduces the intrusion detection problem to monitoring the individual nodes for violation of the constraints. Such monitoring can be performed in a decentralized fashion by cooperative distributed detectors, which allows for scalability. In addition, since the constraints are developed based on the correct behavior, our approach can detect both known and unknown attacks.We choose OLSR (Optimized Link State Routing) [10] as the routing protocol for the current investigation.**

***Keywords*—** Access control, AODV, storage node, Optimized Link State Routing,Topology Control, hop,finite-state-automata,MANET,OLSR.

I. INTRODUCTION TO OLSR

OLSR is a proactive table-driven link-state routing protocol developed by INRIA [10]. The protocol is a refinement of traditional link state protocols employed in wired networks; in the latter, the local link state information is disseminated within the network using broadcast techniques. This flooding effect will consume considerable bandwidth if directly employed in the MANET domain, and therefore, OLSR is designed to optimally disseminate the local link state informationaround the network using a dynamically established sub-network of multipoint relay (MPR) nodes; these are selected from the existing network of nodes in the MANET by the protocol.

OLSR employs two main control messages: Hello messages and Topology Control (TC) messages to disseminate link state information. These messages are periodically broadcast in the MANET in order to establish the routing tables at each node independently. In OLSR, only nodes that have bidirectional (symmetric) links between them can be neighbors. Hello messages contain neighbor lists to allow nodes to exchange neighborinformation, and set up their 1-hop and 2-hop neighbor lists; these are used to calculate multi-point relay (MPR) sets[1].

An MPR set is a 1-hop neighbor subset of a node to be used to reach all 2-hop neighbors of the node. OLSR uses MPR sets to minimize flooding of the periodic control messages. Nodes use Hello messages to announce their MPR sets together with 1-hop neighbor sets. When a node hears its neighbors choosing it as an MPR node, those neighbors are MPR selectors of the node, and the node will announce its MPR selector set to the network by broadcasting TC messages[2].

TC messages are forwarded by MPR nodes to all nodes of the network. When a node receives a TC message, it will note that the originator of TC message is the “last-hop” toward all MPR selectors listed in the TC message. The links are then added into the topology table. Using its topology table, the node can set up its routing table by recursively traversing the (last-hop to node, node) pairs in its topology table (see Figure 1) and picking up the shortest path with the minimal hop count. Therefore, each node of the network can reach all other nodes[3].

Figure 1: Generation of a route from Topology Table

Several studies have been done on the vulnerabilities of OLSR [15][4]. In general, an attacker can fabricate packets, intercept and modify packets going through it, or refuse to forward packets, causing compromises of confidentiality, integrity, and availability. In this work, we only focus on those vulnerabilities that could compromise the integrity of the network, i.e., the routing tables in the nodes. In OLSR, each node injects topological information into the network through HELLO messages and TC messages. Therefore, a malicious node can inject invalid HELLO and TC messages to disrupt the network integrity, causing packets to route incorrectly or to the advantage of the attacks[4].

Table 1 displays the critical fields in the Hello message and the TC message on which the computation of the routing table depends. The 1-hop neighbor list in a Hello message is used by its neighbor to create the 2-hop neighbor list and MPR set. The MPR set in a Hello Message denotes the MPR set of the sender. The MPR selectors in TC messages are used in calculating routing tables at nodes receiving the messages[5].

Table 1: Critical fields in Hello and TC Messages

Message Type	Critical fields
Hello Message	1-hop neighbor list MPR sets
TC Message	MPR selectors Advertised neighbor sequence number (ANSN)

II. ATTACK IMPACT

Since every node concludes the same topology for the network from the TC messages broadcasted around the MANET, an attacker can influence this topology using the four attack methods described above. He can add or delete links in the routing tables of other nodes with these invalid messages. In addition, invalid messages from an attacker may trigger other incorrect messages that invalidate routing tables in the entire MANET [6].

For example, using the first method, an attacker can add a non-neighbor node in the 1-hop neighbor list of its Hello message. Other neighbor nodes of the attacker node may add the attacker as MPR in their Hello messages due to this non-existent neighbor. The attacker can now advertise this in its TC messages. As the TC message propagates through the whole network, every other node's routing table is corrupted [6].

With regards to the TC message vulnerabilities, examples of attack include the following: If, in an initiated TC message, an attacker node fails to include a legitimate MPR selector, this may potentially deny service to this existent MPR selector; this denial of service may be partial or total depending on the topology around the victim node. Similarly, if, in a forwarded TC message, an attacker modifies the ANSN field, or the MPR selector list, then it effectively alters how the routing table is established at other nodes around the network. This may affect not only the network service at the neighborhood of the victim node that originated the TC, but may result in cascading network effects that arise from how routing decisions are made by nodes around the network[7].

These modifications of OLSR control message fields used by a single attacker as described above follow the basic format specifications of OLSR messages. This makes them hard to detect. However, they conflict with other OLSR control messages from other nodes. We call these conflicts "inconsistencies"[7].

III. INTRUSION DETECTION MODEL

This section describes our specification-based approach to detecting attacks in OLSR. In general, specification-based detection recognizes attacks by comparing the activity of an object with a model of correct behavior of the object. It has been applied to detect attacks on computer programs and network protocols. Specification-based detection is particularly suitable for detecting attacks on network protocols because the correct behavior of a protocol is usually well defined and documented in the protocol specification. The challenge is to extract a suitable model of behavior from the protocol specification that can be checked at runtime using network monitoring. We first list assumptions employed, and then present the correct behavior model of OLSR under these assumptions [8].

A. ASSUMPTIONS

We assume a distributed intrusion detection architecture that allows cooperative detectors to promiscuously monitor all Hello and TC messages, and exchange their local data if necessary. IDS detectors in this architecture can monitor all Hello and TC messages sent by each node of the network, always exchange IDS data successfully, and will not be compromised.In addition, we assume that cryptographic protection, such as DRETA is employed to guard against spoofing attacks. Furthermore, we assume OLSR is the only routing

protocol in the network and each node has only one network interface. In other words, Multiple Interface Declaration (MID) and Host and Network Association (HNA) messages are not used here. Lastly, we assume nodes forward TC messages following OLSR Default Forwarding Algorithm and nodes forward normal packets to the correct next hop [9].

B. CORRECT BEHAVIOR MODEL OF OLSR

Figure 2 shows the FSA model of the OLSR protocol that defines the correct operation of an OLSR node in handling control traffic. When a node receives a Hello control message, it will update its neighbor list and MPR set. Upon receiving a TC control message, a node updates the topology and routing table. In addition, the node will forward the TC if it is a MPR node. In addition, a node will periodically broadcast Hello and TC messages [10].

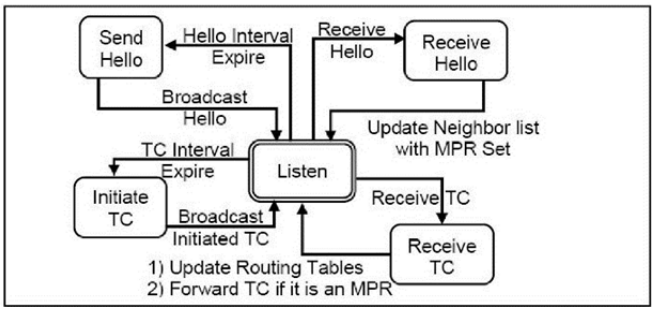


Figure 2: OLSR Routing Finite State Automata (FSA)

We describe the constraints on the control traffic between neighbor nodes for detecting inconsistencies within the control messages.

C1: Neighbor lists in Hello messages must be reciprocal. E.g., if node 2 is the neighbor of node 1, then node 1 must be node 2's neighbor.

C2: The MPR nodes of a node must reach all 2-hop neighbors of the node and the MPR nodes must transmit TC messages periodically.

C3: MPR selectors of a TC message must match corresponding MPR sets of Hello messages. E.g., if node 2 is node 1's MPR selector, node 1 must be node 2's MPR.

C4: Integrity of forwarded TC messages must be maintained.

C1 ensures that 1-hop neighbor lists of Hello messages from all nodes are consistent. According to the OLSR routing specification, since 1-hop neighbor lists are consistent, nodes can produce correct 1-hop and 2-hop neighbor lists. C2 ensures that MPR nodes of each node connect all 2-hop neighbors of the node. By definition of MPR, MPR sets are correct. C3 ensures that MPR selector sets are consistent with MPR sets and therefore are correct. C4 ensures that the forwarded MPR selector sets are correct [11].

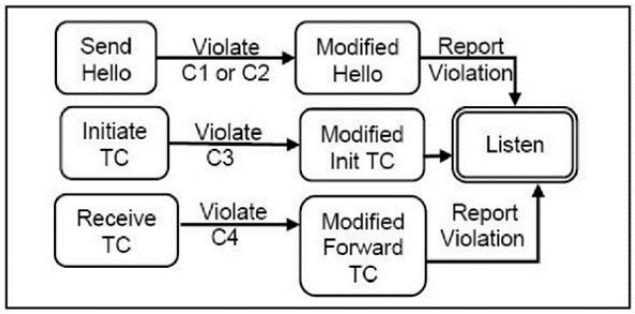


Figure 3: Security Specification Finite State Automata

Figure 3 (an extension of the FSA in Figure 2) depicts the FSA used by the specification-based intrusion detection system. When a OLSR control message violates one of the constraints, the FSA moves from a normal state into one of the alarm states (Modified Hello State, Modified Init TC State, Modified Forward TC State) To recover from the errors, a detector may broadcast the corrected TC message, or force the node causing the violation to resend the corrected Hello message, and thereby recover corrupted routing tables of infected nodes. Thus, the report violation actions in the FSA can be enhanced to perform the corrective action. Since our proposed model is only dealing with intrusion detection, we do not explore such recovery actions further in this work. However, this preliminary recovery model is incorporated into our simulation experiments using GloMoSim [12].

C. TEMPORARY INCONSISTENCY

Temporary violation of constraints C1, C2 and C3 may occur in a short period of time as links are created or removed when the topology changes. To avoid false alarms, a detector must wait for the two nodes on both sides of a link to learn the new link status before asserting the inconsistency as an attack. For example, if a new link between node A and node B is created, node A may update the status of link A-B and send a Hello message that is not consistent with the previous Hello message of node B, which does not claim that link A-B exists. The detector should wait for node B to receive the new Hello message from A and send a new Hello message that reflects the addition of link A-B. In case of broken links, (leading to lost messages), the detector should wait for the expiration of the old records at the nodes. In other words, if a detector detects violation of constraint 1, 2 or 3 with regard to nodes A and B, and the violation continues to occur after a certain threshold, then the detector will raise an alarm. In addition, because temporary inconsistency propagates due to an unstable asymmetric link, constraints 1 and 2 require 12 seconds and constraint 3 requires 15 seconds because of the 5 second TC interval time. For constraints C4, since the validation of new messages depends on the messages from the originators, temporary inconsistency does not occur [13].

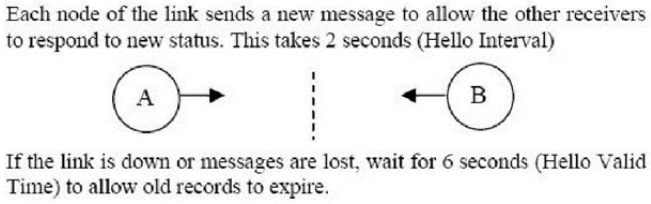


Figure 4: Resolving temporary inconsistency between nodes of a link

Table 2: Important Parameters for Temporary Inconsistency

Constraint Alert thresholds		OLSR Default Parameters	
C1 (1-hop neighbors)	12 sec	Hello message sending interval	2 sec
C2 (2-hop neighbor vs MPR)	12 sec	Hello message valid time	6 sec
C3 (MPR vs MPR selector)	15 sec	TC message sending interval	5 sec
C4 (Forwarded TC)	0 sec	TC message valid time	15 sec

D. LIMITATIONS

For a single attack or non-correlated attacks, the model can detect all attacks since we capture all possible ways to modify a single message at a time. But if two or more attackers launch a correlated attack in which incorrect information is supplied to multiple nodes consistently, the constraints may not be able to detect it. For example, if two attackers are not neighbors but both claim they are neighbors, there may be no detectable violation. Because Hello messages are 1-hop broadcast messages and detectors do not know who actually receive them, detectors are not able to employ constraint C1 to detect violations. This attack is a tunneling attack— attackers build up a virtual link between them.

1. Exchange 1-hop neighbor lists by Hello messages
2. Establish 2-hop neighbor lists by 1-hop lists
3. Generate MPR sets by 2-hop neighbor lists and announce them with Hello messages
4. MPR nodes generate TC messages advertising the nodes (MPR selectors) that can be reached by the MPR nodes.
5. MPR nodes forward TC messages so that they will reach all nodes in the network.
6. Generate topology and routing tables from MPR selector sets

Table 3: OLSR Routing Table Establishment

IV. ANALYSIS OF THE OLSR DETECTION MODEL

In this section, we analyze the OLSR protocol and the proposed detection model to show that the set of constraints C1 — C4 can identify attacks in MANETs. As discussed, a malicious node can disrupt the integrity of the network (causing good nodes to change their routing table to its advantage) by intentionally generating and forwarding incorrect control messages. In particular, we show that in an OLSR network with only one malicious node, these constraints ensure that the malicious node cannot compromise the integrity of the routing tables of any good nodes [14].

Table 3 describes the process for establishing the routing table from the perspective of a node. Initially, a node exchanges its 1-hop neighbor list with its neighbors using Hello messages. Then the node establishes its 2-hop neighbor list based on the Hello messages fromits neighbors.

Based on the 2-hop neighbor list, the node generates the MPR set and announces them in Hello messages. Nodes that are chosen to be MPR will generate TC messages and forward TC messages originating from other nodes so that every node will receive all the TC messages. Finally, a node computes the routing table from the information in the Hello messages and TC messages.

According to the OLSR protocol RFC [10], each node maintains a link set and a topology set that are used for calculation of the routing table. The link set contains the link information of its 1-hop neighbor, and is constructed from the Hello messages it receives. The topology set contains topology tuples in the form of T[DestAddr], T[LastHopAddr], T[Seq], T[HoldingTime], which indicate that one can reach T[DestAddr] through T[LastHopAddr]. The topology set is constructed from the TC messages a node receives. A node computes the routing table from its link set and topology set. Therefore, the routing table of a node is correct if its link set and topology set are correct [14].

Lemma 1: All good nodes will have a correct link set if constraint C1 holds.

First, according to the OLSR routing specification, a node builds and maintains its link set from the 1-hop neighbor field of the Hello messages it receives. Therefore, if the 1-hop neighbor fields of all Hello messages and the source address are correct, then all nodes will have a correct link set.

Now, we show that a Hello message with an incorrect 1-hop neighbor field will be detected as a violation of C1. Consider a bad node which produces a Hello message with an incorrect 1-hop neighbor field. There are two possibilities:

1) It claims another node A as its 1-hop neighbor, but A is not. In this case, the IDS will detect this when it compares the Hello message from the bad node with the Hello message from A.

2) It omits, in its set of 1-hop neighbors, a real neighbor B. In this case, the IDS will detect a violation of C1 when it compares the Hello message from the bad node to the Hello message from B.

In both cases, the incorrect Hello message will be detected as a violation of constraint C1. Given that the source address of a Hello message is correct (by the assumption of no spoofing), all nodes will have a correct link set if constraint C1 holds.

Lemma 2: The MPR selector field of a TC message generated by an MPR node must be correct if constraint C3 holds.

According to the OLSR specifications, a (complete) TC message contains the set of MPR selectors of the originating node. There are two cases in which the MPR selector field in the TC message could be wrong.

1) The MPR selector field contains a node X that is not a MPR selector of M.

2) The MPR selector field misses a node Y that is a MPR selector of M

In case 1, the Hello message generated by node X will be inconsistent with the TC message. Therefore, the IDS will detect the violation of constraint C3. In case 2, the Hello message generated by node Y will be inconsistent with the TC message, and thus will be detected.

Lemma 3: The MPR selector fields of all TC messages must be correct if con¬straints C3 and C! hold

Table 4: Radio Propagation Parameters in GloMoSim

PROPAGATION-LIMIT (dBm)	-111
RADIO-TX-POWER (dBm)	15
RADIO-ANTENNA-GAIN (dBm)	0
RADIO-RX-SENSITIVITY (dBm)	-91
RADIO-RX-THRESHOLD (dBm)	-81
Antenna Height (m)	1.5

Any TC message in the network is either an original message sent by the originating node or a forwarded message. In the former case, Lemma 1 guarantees the correctness of the selector fields. In the latter case, constraint C4 assure that the forwarded TC message must be the same as the original TC message; thus, the MPR selector field must be correct.

Lemma 4: For a node x, which is a n-hop neighbor of a different node y, x will receive TC messages of y with n-1 forwarding if C2 holds.

We use induction to prove this lemma.

1) For n = 1, all y's one-hop neighbors will receive TC messages without forwarding. For n = 2, all y's two hop neighbors will receive TC messages of y with one forwarding if C2 hold.

2) (Induction step) We assume that any node A will receive a TC message of an n-hop neighbor B with n−1 forwarding if C2 holds for all $2 < n < k$. For a node x that is a k-hop neighbor of a node y, without loss of generality, let x, N1, N2, ...,$_{Nk-1,}$ y be a path from x to y such that N1 is a MPR of N2. We argue that such a path exist if C2 holds — since N2 is a 2-hop neighbor of x, there must be a MPR of N2 through which N2 can reach x. As node N1 is a $k - 2$ hop neighbor of y, by the inductive assumption N1 will receive TC messages from y with $k - 2$ forwarding. Therefore, x will receive TC messages from y through N1 $k - 1$ forwarding.

By induction, Lemma 4 is true for all integer $n > 0$.

Theorem 1: All nodes will have a correct routing table if constraints C1, C2, C3, and C! hold.

Since each node in the MANET computes the routing table based on the link set and the topology set, the routing table will be correct if the two sets are correct. Given that C1 holds, Lemma 1 ensures that the link set in each node is correct. Given that C3 and C4 hold, Lemma 3 ensures that the MPR selector field of all the TC messages that a node receives is correct. Given C2, Lemma 4 ensures that a node will receive TC messages from all nodes. According to the OLSR specification, the topology set is computed from the TC messages. Therefore, the topology set will be correct if, in addition, every MPR sends out the TC messages. Since constraint C2 guarantees that all nodes in the true MPR set send out TC messages, the topology set in each node must be correct. Therefore, the routing table in each node must be correct [11].

V. SIMULATION

To measure and validate the effectiveness of our approach, we have implemented the detection mechanism for checking the constraints and experimented with it in a simulated OLSR network under a variety of mobility scenarios. We have implemented several of the example attacks to test the detection capability. In addition, we tested the prototype under a normal situation to measure the performance, especially false positive characteristics [11].

A. SIMULATION ENVIRONMENT

We used the GloMoSim simulation platform to experimentally validate our approach. The simulation is based on IEEE 802.11 and Ground Reflection (Two-Ray) Model, having both the direct path and a ground reflected propagation path between transmitter and receiver. The radio range is around 376.7 meters, calculated by the parameters shown in Table 4 [11].

The network field is 1000 m x 1000 m region divided into cells. Nodes are placed into each cell randomly. Each attack scenarios has a stable topology with 10 nodes. Total simulation time is 600 seconds.

In the experiments all mobile nodes follow the Random Waypoint Mobility Model with speed of 5,10, and 20 m/s, and pause times of 0, 30, and 60, ..., 300 seconds. For background traffic, the numbers of mobile nodes uses were 50, 100, 200, and 400. In the experiments, 10% of mobile nodes continuously generated 1024 byte packets at a constant rate of 1 packet per second, 8K bps, across the network topology. The simulation metrics mainly focus on false positives, false negatives, the distribution of temporary inconsistency lasting time and maximum value for each constraint.

B. IMPLEMENTATION OF DETECTION MECHANISM

Our proof-of-concept prototype is implemented as a global detector that can monitor all Hello and TC messages in the simulated OLSR network. It is important to note that although the current prototype is a centralized detector, the proposed intrusion detection model can be implemented in a decentralized. As the goal of the proof-of-concept prototype is to validate the detection model, a centralized implementation suffices for validating the false positive and false negative characteristics under our assumptions [9].

Four data tables are maintained by the global detector to record 1-hop neighbors, 2-hop neighbors, MPR and MPR selector sets of all nodes. Four constraints are evaluated according to data tables and incoming messages. An alert will be raised if a constraint is violated. However, topology changes will cause temporal inconsistency and lead to false alert. To minimize the false positive rate, we develop a mechanism to detect temporal inconsistency between new message and old history data. First, we set threshold time for each constraint according to intervals of Hello messages and TC message. Then we generate alerts only when an inconsistency last beyond the threshold time of a constraint. As an example, we list the pseudo code of Constraint C1 in Figure 5 [6]:

```
Constraint 1 (1-hop Table, node i)
For each 1-hop neighbor j in 1-hop Table i
 If i is not in 1-hop Table j  // if there is inconsistency between link states of node i and j
  { If 1-hop Table(i,j).alert = = FALSE  //if no inconsistency before
    {Set 1-hop Table(i,j).alertTime = Current Time //set time stamp
    Set 1-hop Table(i,j).alert = TRUE //mark the inconsistency}
    Else {If (Current Time - 1-hop Table(i,j). alertTime) > Threshold of C1 // if inconsistecy
       Raise Alarm of C1}}          //has lasted more than threshold, raise an alarm
    Else{1-hop Table(i,j).alert = FALSE}
```

Figure 5: Pseudo code of constraint C1

C. EXAMPLE ATTACK SCENARIO AND RESULTS

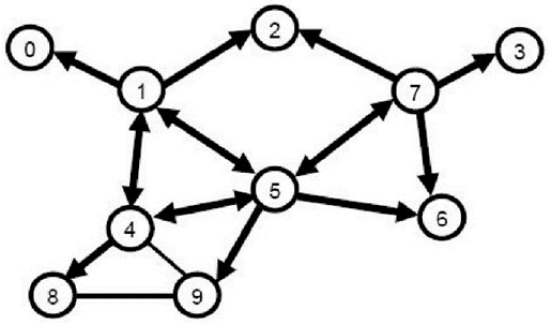


Figure 6: Example Topology in OLSR

We implemented one example of a man-in-the-middle attack and two examples of denial of service attacks using the four attack methods. We present an example topology, shown in Figure 6 and Table 5, in order to illustrate the details of the example attacks and their impact.

Table 5: Relevant OLSR data for example topology

	0	1	2	3	4	5	6	7	8	9
1-hop	1	0,2,4,5	1,7	7	1,5,8,9	1,4,6,7,9	5,7	2,3,5,6	4,9	4,5,8
2-hop	2,4	6,7,8,9	0,3,4,5,6	2,5,6	0,2,6,7	0,2,3,8	1,2,3,4,9	1,4,9	1,5	1,6,7
MPR	1	4,5	1,7	7	1,5	1,4,7	5,7	5	4	5
MPR Selector	-	0,2,4,5	-	-	1,5,8	1,4,6,7,9	-	2,3,5,6	-	-

In each example attack, the attacker uses attack mechanisms slightly modifying the control messages to trigger changes in the routing tables of other nodes as desired by the attacker. These example attacks demonstrate that, by employing carefully designed modifications, an attacker can successfully manipulate routing tables at other nodes. Note that we simulate the attacks with no mobility to ensure the attacks are effective [7].

For each example attack, the detector detects the attacks as violations of the constraints. In this implementation, we combine a recovery model with the intrusion detection model. To recover the corrupted routing tables of infected nodes from the attack, the detector may send the correct TC message with a higher ANSN and the correct MPR selector set to override the corrupted TC message. If the compromised node is the originator of the message, the detector commands the node to resend correct messages to override the corrupted messages. The simulation shows the correct messages successfully override the corrupted messages and correct the infected routing tables [8].

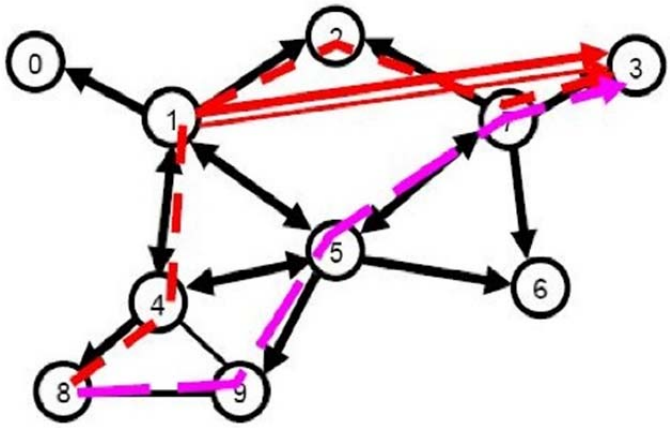


Figure 7: Man in the Middle Attack by A1&A3 Man in the Middle Attack by A1&A3

An attacker at node 1 intends to change a route, 8 —* 9 —* 5 —* 7 —* 3, to go through itself. It uses attack methods 1 and 3 to convince nodes 8 and 4 to forward packets toward 3 through itself, and then it can use 2 to forward the packets from 8 to 3. First, by attack method 3, node 1 adds 3 into its MPR selectors in its new TC message to make 8 choose 4 as the next hop toward 3. 8 receives the new forged message and choose 1 as the last hop to 3 in its topology table; this is used to reach all node in 3 or more hops away. Since 1 is a 2-hop neighbor of 8, from 8's point of view, route 8 —* 3 becomes 8 —* 4 —* 1 —* 3, so 8 chooses 4 as the next hop toward 3 in its routing table [6].

Second, by attack method 1, node 1 adds 3 to its 1-hop neighbors in its new Hello message in order to make 4 choose 1 as the next hop toward 3. After receiving the forged message, 4 adds 3 into its 2-hop neighbor list, and chooses 1 as the next hop toward 3 in its routing table. Thus, when 8 forwards packets toward 3 to 4, 4 forwards them to 1, and attacker 1 can forward the packets from 4 to node 2 in order to successfully change the route from 8 to 3. Since 2 received 7's Hello first and added 3 as a 2-hop neighbor, 2 will not choose node 1 as its next hop toward 3. 2 forwards the packets to 7 and 7 forwards them to 3. The attack is a success. Note that attacker 1 has to continuously broadcast forged messages to make the attack remain effective.

If an attacker arbitrarily adds other non-neighbors into its 1-hop or MPR selectors, it will easily make itself a black hole. This will attract much useless traffic, marking itself as an attacker. However, in this case, the attacker (node 1) successfully launches a man in the middle attack by slightly changing its two messages without forging its own address, and therefore it is difficult to detect this attack using other existing approaches [8].

Using constraint 3, the detector detects that 1's MPR selectors = [2,3,4,5] in 1's TC message do not match 3's MPR= [7] in 3's Hello message. Additionally using constraint 1, 1's 1-hop neighbors = [0,2,3,4,5] in 1's Hello message do not match 3's 1 hop neighbors = [7] in 3's Hello message. Since the attacker keeps sending the forged messages, the detected inconsistencies easily last over the temporary inconsistency threshold for C1 and C3, which is 12 seconds. Therefore, the detector detects the attacks correctly. The maximum temporary inconsistency here is less than 12 seconds. Finally, the detector commands node 1 to send correct TC(1) = [2,4,5] and Hello(1) = [0,2,4,5], and then 8 and 4 receive 1's correct TC and Hello and use 9 and 5 to reach 3. The route is 8 —* 9 —* 5 —* 7 —* 3 is recovered.

Denial of Service by A2

An attacker at node 7 intends to annul a route 8 —* 4 —* 5 —* 7 —* 3 by attack method 2, i.e., declaring an incorrect MPR list in its Hello message. First, 7 removes 5 from its MPR set(becoming empty) in its Hello message. Second, 5 receives 7's modified Hello and believes 5 is not in 7's MPR set, so 5 removes 7 in its MPR selectors = [1,4,6,9] in 5's new TC. When 8 receives 5's new TC, 8 believes 8 cannot use 5 as the last hop to reach 7. Since 7 is 3 hops away from 8 and 8 cannot use anyother node as the last hop to reach 7, 8 cannot reach 7 and therefore cannot reach 3.

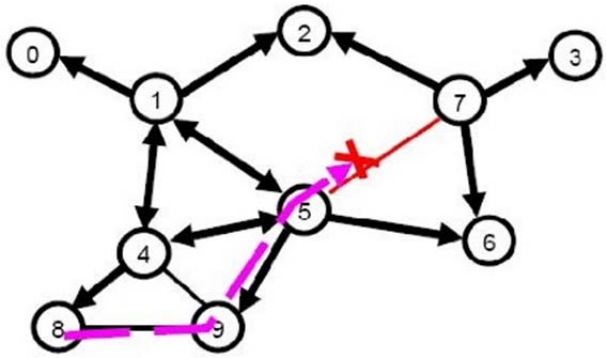


Figure 8: Denial of Service by A2

The route 8 to 3 is down. This attack is harder to detect than the first one because it requires 2-hop neighbor information which is not explicitly sent out in Hello or TC messages.

Note that 5 will not forward 7's TC messages, so no node except 7's 1-hop neighbor will have 7's TC messages. This makes 0, the other 3 hop neighbor of 7, unable to connect to 7. If there were a route from 0 to 3, it is also down.

Using constraint 2, the detector detects that 7's MPR set is empty and 7's MPR set has not reached all of 7's 2-hop neighbors [1,4,9]. Once the inconsistency lasts over 12 seconds, the alert is raised. So the detector commands 7 to send correct a MPR set [5] in 7's new Hello message. When receiving correct Hello message from 7, 5 adds 7 back to 5's MPR selectors [1,4,6,7,9] in 5's new TC. Then 8 receives 5's new TC and uses 5 to connect to 7. The route becomes available again. Here there is no temporary inconsistency for C2, so no false positive.

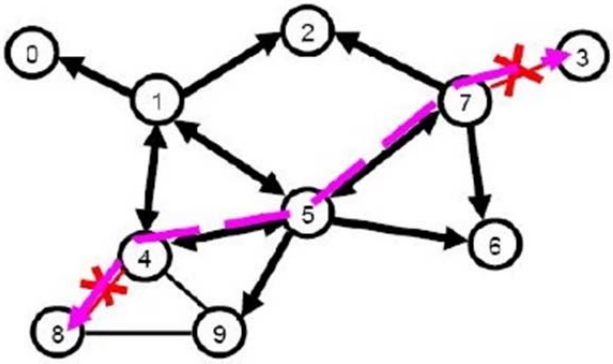


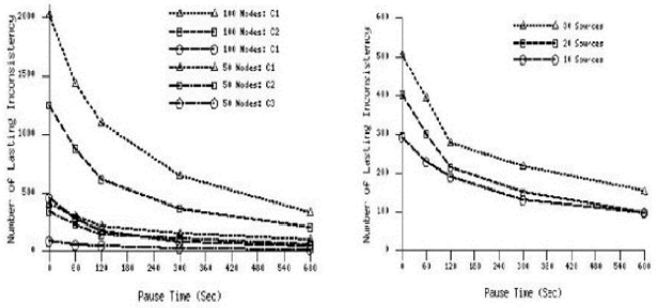
Figure 9: Denial of Service by A4 Denial of Service by A4

An attacker at node 2 intends to annul route 8 +-> 4 +-> 5 +-> 7 +-> 3 by attack method 4, where forwarded TC messages are modified with a high ANSN. It uses two forged forwarded TC messages to remove the global links, 4 —* 8 and 7 —* 3. First, 2 broadcasts TC(7)=[2,5,6] without 3 to make 8 not use 7 to reach 3; thus route 8 —* 3 is down. Again, 2 broadcasts TC(4), which is [1,5], without 8 to make 3 unable to use 4 to reach 8; now route 3 —* 8 is down. Since the forged TC messages have high ANSN, all other nodes hearing them replace the correct information with the forged one, so that 8 and 3 cannot communicate with each other. The bidirectional route is down. Note that other nodes except 3,4,7,8 can do the same thing. If 4 or 7 does this, it is using attack method 3, not 4. This attack can be detected by authenticating forwarding messages.

By constraint 4, the detector detects that TC(4) and TC(7) sent by 2 do not match those from the originators, 4 and 7, respectively. The detector sends correct TC(4) and TC(7) with ANSNs higher than forged messages to override them. Finally, 3 and 8 receive correct TC messages, and are able to communicate with each other. The route is recovered. Here C4 does not require considering any temporary inconsistency thresholds, and there are no false positives and false negatives [9].

D. TEMPORARY INCONSISTENCY AGAINST MOBILITY

With no mobility, temporary inconsistencies only happen when nodes establish 1-hop neighbor relationships in the first 5 seconds. Once they are capable of sending TC messages, no temporary inconsistency occurs. In mobile topologies, temporary inconsistencies keep happening while nodes move. We choose different mobility pause times of 0, 30, 60, 120, 300, and 600 seconds employing the Random Waypoint Mobility Model with a speed range of 1 to 20 meter/sec to demonstrate different levels of mobility. We also simulate 10, 20, and 30 traffic sources with continuously generating 512 byte packets at a constant rate of 1 packet per second, 5K bps, across the network topology.



(a) 100 nodes vs 50 nodes, 20 sources (b) 50 nodes with 10, 20, 30 traffic sources (C1 only)

Figure 10: Number of lasting temporary inconsistencies with different number of nodes and sources

Most temporary inconsistencies will be resolved by the next message of the same kind sent from the same originator and only few of the inconsistencies may last. Figure 10 shows the number of lasting temporary inconsistencies caused by mobility. In Figure 10(a), 100 nodes in a 2000m x 2000m area result in many more inconsistencies than 50 nodes in a 1000m x 1000m area. Although 100 nodes generate 2 times the number of messages than 50 nodes, 100 nodes roughly generate 4 times the number of temporary inconsistencies. The higher the degree of mobility is, the more inconsistencies are generated, especially for inconsistency against C1.

Figure 10(b) shows the number of temporary inconsistencies against C1 in a 50 node topology with 10, 20, and 30 traffic sources. With higher traffic load, the inconsistencies occur more. However, the impact of traffic load for temporary inconsistencies is not as much as that of number of nodes. Therefore, the number of nodes and their degree of mobility are the two main factors of temporary inconsistency.

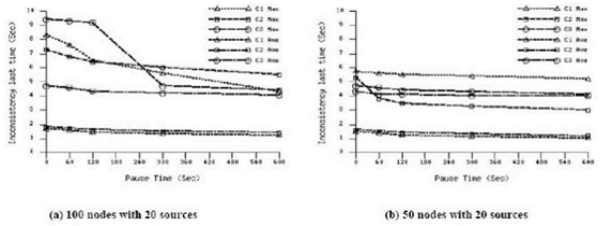


Figure 11: Maximum and Average Temporary inconsistencies lasting time

Maximum temporary inconsistency lasting time indicates what the alert threshold for constraints should be. In Figure 11(a), the maximum lasting times of C1, C2 and C3 are less than the thresholds 12 seconds and 15 seconds, and do not lead to false alarms. If the thresholds are 6 seconds, there will be less than 15 false alarms in a 100 node-topology with low pause times. Although the maximum temporary inconsistency lasting time in a 100-node topology is greater than in a 50 node- topology, their average lasting time is roughly the same, where the times of C1 and C2 are about 1.5 to 1.8 seconds, and those of C3 are 4.7 to 4.3 seconds.

Also, attacks using the four attack methods are tested in 100 node and 50 node mobile topologies. These attacks consist of arbitrary modified values of 1-hop neighbors, MPR, and MPR selectors in the Hello and TC messages and they will continuously send modified messages at least 1 minute. If the attacks contain the addresses of inactive nodes, which do not send Hello messages for 1 minute and include unused nodes, or the attacks violate C4, the detector raises alarms immediately. If the attacks violate C1, C2 or C3, the detector raises an alert while the attacks last larger than the thresholds. The detector detects all attacks in which the modified messages are sent by the attackers. No false positives are found in a mobile topology with background traffic (20 sources).

VI. CONCLUSION

Analyzing the OLSR routing specification, we define the normal OLSR routing behavior and list possible attack mechanisms from a single attacker. Based on the normal routing behavior, nodes retrieve routing information, and establish and maintain their routing tables correctly using the Hello and TC messages. We develop constraints on these Hello and TC messages in order to establish that the integrity of the routing tables at all nodes is not compromised. We develop the proof of satisfaction of the requirement that the integrity of routing tables of all nodes is safe-guarded. Besides, we implement the constraints and example attacks on the Glomosim simulation platform.

At first in this paper, we assumed that it has all Hello and TC messages to have sufficient data for intrusion detection, then we present a distributed intrusion detection model that can supply sufficient routing data for the detection. Secondly, we assumed that a cryptographic protection detects spoofing attacks.

REFERENCES

[1] DEMEM: Distributed Evidence-driven Message Exchange intrusion detection Model for MANET. In Proceeding of the 9th International Symposium Recent Advances in Intrusion Detection (RAID), Hamburg, Germany, 2006.
[2] R. Canetti A. Perrig, D. Tygar, and D. Song. The TESLA broadcast authentication protocol. Cryptobytes, 5(2):2–13, 2002.
[3] O. Kachirskiabd R Guha. Effective Intrusion Detection Using Multiple Sensors in Wireless Ad Hoc Networks. In 36th Annual Hawaii International Conference on System Sciences (HICSS 2003).
[4] C. Adjih, T. Clausen, P. Jacquet, A. Laouiti, P. Muhlethaler, and D. Raffo. Securing the OLSR protocol. In Med-Hoc-Net 2003.
[5] Yi an Huang and Wenke Lee. Attack Analysis and Detection for Ad Hoc Routing Protocols. In Proceedings of International Symposium Recent Advances in Intrusion Detection (RAID) 2004.
[6] Yi an Huang and Wenke Lee. A Cooperative Intrusion Detection System for Ad Hoc Networks. In Proceedings of the ACM Workshop on Security in Ad Hoc and Sensor Networks (SASN) 2003.
[7] FarooqAnjum and Rajesh R. Talpade. LiPad: Lightweight Packet Drop Detection for Ad Hoc Networks. In Proceedings of IEEE 60th Vehicular Technology Conference 2004.
[8] S. Buchegger and J. Boudec. Performance Analysis of the CONFIDANT Protocol: Cooperation Of Nodes - Fairness In Distributed Ad hoc NeTworks. In Proceedings of MobiHoc 2002.
[9] L. Buttyan and J.-P. Hubaux. Stimulating Cooperation in Self-organizing Mobile Ad Hoc Networks. Technical Report DSC/2001/046, Swiss Federal Institute of Technology, Lausanne, 2001.
[10] T. Clausen and P. Jacquet. Optimized Link State Routing Protocol. IETF RFC 3626.
[11] Daniel Sterne et. al. A General Cooperative Intrusion Detection Architecture for MANETs. In Proceedings of the 3rd IEEE International Information Assurance Workshop 2005.
[12] DhanantSubhadhrabandhuet. al. Efficacy of Misuse Detection in Adhoc Networks. In Proceedings of SECON 2004.
[13] K. Bhargavan et al. VERISIM: Formal Analysis of Network Simulations. IEEE Transactions of Software Engineering, 28(2):129, 2002.
[14] SumitGwalani, KavithaSrinivasan, Giovanni Vigna, Elizabeth Belding-Royer, and Richard Kemmerer. An Intrusion Detection Tool for AODV-based Ad hoc Wireless Networks. In Proceedings of Computer Security Applications Conference 2004.
[15] N. Haller. The S/Key one-time password system. Internet Society 1994.
[16] Andreas Hafslund, Andreas Tonnesen, Roar BjorgumRotvik, Jon Andersson, and Oivind Kure. Secure Extension to the OLSR protocol. In In OLSR Interop and Workshop 2004.