Preventing the cluster formation attack against the hierarchical OLSR protocol

Gimer Cervera, Michel Barbeau, Joaquin Garcia Alfaro, Evangelos Kranakis

Preventing the Cluster Formation Attack Against the Hierarchical OLSR Protocol

Gimer Cervera[1], Michel Barbeau[1], Joaquin Garcia-Alfaro[2], and Evangelos Kranakis[1]

[1] School of Computer Science,
Carleton University, K1S 5B6, Ottawa, Ontario, Canada.
{gcevia,barbeau,kranakis}@scs.carleton.ca

[2] Institut Telecom, Telecom Bretagne,
LUSSI Dept., Cesson-Sevigne, 35576, France.
joaquin.garcia-alfaro@acm.org

Abstract. The Hierarchical Optimized Link State Routing (HOLSR) protocol enhances the scalability and heterogeneity of traditional OLSR-based Mobile Ad-Hoc Networks (MANETs). It organizes the network in logical levels and nodes in clusters. In every cluster, it implements the mechanisms and algorithms of the original OLSR to generate and to distribute control traffic information. However, the HOLSR protocol was designed with no security in mind. Indeed, it both inherits, from OLSR, and adds new security threats. For instance, the existence of misbehaving nodes can highly affect important HOLSR operations, such as the cluster formation. Cluster IDentification (CID) messages are implemented to organize a HOLSR network in clusters. In every message, the *hop count* field indicates to the receiver the distance in hops to the originator. An attacker may maliciously alter the *hop count* field. As a consequence, a receiver node may join a cluster head farther away than it appears. Then, the scalability properties in a HOLSR network is affected by an unbalanced distribution of nodes per cluster. We present a solution based on the use of hash chains to protect mutable fields in CID messages. As a consequence, when a misbehaving node alters the hop count field in a CID message, the receiver nodes are able of detecting and discarding the invalid message.

Keywords: MANETs, Routing, HOLSR, Security, Hash Chains.

1 Introduction

The Hierarchical Optimized Link State Routing (HOLSR) [14] is a proactive routing protocol designed to improve scalability of heterogeneous Mobile Ad-Hoc Networks (MANETs). HOLSR has two phases: i) *cluster formation* and ii) *topology map acquisition*. In the first phase, HOLSR organizes the network in logical levels and nodes in clusters. In the second phase, HOLSR implements the mechanisms and algorithms of the original OLSR [4] to generate and to distribute control traffic messages. Information contained in Hello and Topology Control (TC) messages are used to calculate optimal routes from any given node to any destination within each cluster. Additionally, Hierarchical Topology Control (HTC) messages are implemented to advertise membership

information from a cluster to other nodes in higher levels. Nevertheless, HOLSR was designed without security measures. Therefore, both phases are vulnerable to malicious attacks. In HOLSR networks, a malicious attack can be perpetrated by a node that interrupts the flooding of control traffic information or does not obey the rules of the protocol to maintain the hierarchical architecture. In this paper, we describe a cluster formation attack against the HOLSR protocol during the cluster formation phase.

During the first stage, every cluster head advertises itself through the periodical generation of CID messages that invite other nodes to join. Every CID message has a hop count field that indicates the distance to the originator. The *cluster head* field of a CID messages points to the originator. When the receiver node joins a cluster head, it generates a new message increasing by one the *hop count* field. When a node receives messages from different cluster heads, it joins the closest cluster head, in terms of hops. When a node receives CID messages from multiple cluster heads, but with the same hop count, it attaches itself to the cluster head from which it received the first message and remains with that cluster head until the topology changes. As a consequence, a node at the border of different clusters only accepts control traffic information from its cluster. An attacker might unsettle this process by generating CID messages with and invalid hop count field. This attack, has a higher impact when the hop count field value is drastically reduced. The receiver nodes may join a cluster head which is farther away than it appears. As a result, the affected cluster head may be overloaded due to an unbalanced node distribution. Additionally, the nodes in some clusters have to include more elements in their routing tables adding unnecessary overhead to the cluster. We handle this risk by implementing a mechanism that implements hash chains to protect the *hop count* field in every CID message. Our solution is based on the work of Hong et al. in [6]. They present a wormhole detective mechanism and an authentication protocol to strengthen the neighbor relationship establishment in standard OLSR. We address a different kind of attack in HOLSR networks. Our mechanism protects the integrity of CID messages and enforces the uniform distribution of nodes in every cluster.

Organization of the paper — Section 2 reviews the OLSR protocol. Section 3 presents the HOLSR protocol. Section 4 describes the cluster formation attack. Section 5 presents a new security extension to the protocol leveraging hash chains that mitigates the cluster formation attack. Section 6, shows our results and simulations setup. Section 7 presents the related work. Finally, Section 8 closes the paper with our conclusions.

2 Optimized Link State Routing Protocol

This section presents a brief overview of the OLSR protocol. OLSR is a proactive routing protocol designed for MANETs. The core of the protocol is the selection, by every node, of Multipoint Relays (MPRs) among their one-hop symmetric neighbors. OLSR nodes flood the network with link-state information messages. The link-state information is constructed by every node and involves periodically sending Hello and TC messages. This information is used to determine the best path to every destination in the network. Due to the proactive nature, the routes are immediately available when needed. The OLSR protocol is based on hop by hop routing, i.e., each routing table lists, for each reachable destination, the address of the next node along the path to that

destination. To construct a topology map, every node implements a topology discovery mechanism leveraging the periodic exchange of control traffic messages. Topology discovery includes: link sensing, neighbor detection and topology sensing.

During this first stage (link sensing), every node populates its local link information base (link set). This phase is exclusively concerned with the OLSR interface addresses and ability to exchange packets between such OLSR interfaces. Then, during the neighbor detection stage, every node populates its neighborhood information base (i.e., one-hop and two-hop neighbor set). The link sensing and neighbor detection phases are based on the periodic exchange of Hello messages. Hello messages are solely transmitted to one-hop neighbors. Information contained in Hello messages allows every node to construct and maintain neighbor tables, as well as to select its MPR set. The MPR set is selected such that all two-hop neighbors are reachable through, at least, one MPR. In the neighbor table, each node records the information about the one-hop neighbor link status (i.e., unidirectional, bidirectional or MPR), with this information every node builds its MPR selector set, i.e., the number of neighbors who selected that node as their MPR.

Topology sensing is achieved through the exchange of TC messages. TC messages are generated and retransmitted exclusively by the MPRs. These messages allow each node to construct its topology table and to declare its MPR Selector set. The MPR Selector Set is the collection of nodes that have selected a given node as an MPR. A TC contains the MPR Selector Set of its originator. A node that has an empty MPR Selector Set does not send or retransmit any TC message. An MPR forwards a message if it comes from a node in its MPR Selector Set. This forwarding algorithm is defined in [4]. Using the information from TC messages, each node maintains a topology table where each entry consists of: (i) an identifier of a possible destination, i.e., an MPR selector in a TC message, (ii) an identifier of a last-hop node to that destination, i.e., the originator of the TC message, and (iii) an MPR Selector Set sequence number [8]. It implies that a possible destination (i.e. an MPR selector) can be reached through the originator of the TC message. If there is an entry in the topology table whose last-hop address corresponds to the originator of a new TC message and the MPR Selector Set sequence number is greater than the sequence number in the received message, then the new message is discarded. Routing tables are constructed using the information from the neighbor and topology table.

OLSR implements two optional messages: Multiple Interface Declaration (MID) and Host and Network Association (HNA) messages. MID messages are used to declare the presence of multiple interfaces on a node. HNA messages are employed to inject external routing information into an OLSR network and provide connectivity to nodes with non-OLSR interfaces. HNA and MID are exclusively retransmitted by the MPRs and following the default forwarding algorithm defined in [4]. MID messages are implemented in a network with multiple interface nodes. Additional information is necessary in order to map interface addresses to main addresses. In OLSR, the main address is defined as the OLSR interface address. A node with multiple interfaces must generate periodically MID messages announcing all its interfaces to other nodes in the network. Thus, every node in an OLSR network will associate multiple interfaces to a node's main address. Nodes with just one interface do not generate MID messages

and the main address is the OLSR interface address. A node with several interfaces, where only one of them is participating in an OLSR network must not generate MID messages. MID messages are retransmitted exclusively by the MPRs following the default forwarding algorithm . Upon receiving a MID message, the information is stored in an Interface Association table. This information is used to construct the routing tables. Then, if a node misbehaves and does not retransmit MID messages, the proper construction of the routing tables is compromised.

In an OLSR network, a node with multiple interfaces might be connected to an external network (e.g., an Ethernet) not running OLSR. In this case, the node acts as a gateway and may inject external routing information in the OLSR network. Thus, a node connected to an external network should periodically generate HNA messages announcing its external network address and netmask. HNA messages flood the network following the default MPR forwarding mechanism. The flooded information is used by the OLSR nodes to construct their routing tables. HNA messages can be considered as a generalized version of the TC messages. Like TC messages, the originator of the HNA messages announces reachability to the others.

3 The Hierarchical OLSR Protocol

OLSR is a *flat* routing protocol designed exclusively for MANETs. However, the performance of the protocol tends to degrade when the number of nodes increases due to a higher number of topology control messages propagated through the network. Scalability can be defined as the capacity of the network to adjust and to maintain its performance even when the number of nodes in the network increases [14]. The MPR mechanism is local and therefore very scalable. However, the diffusion by all the nodes in the network of all the link-state information is less scalable. For instance, in [11] Palma et. al. show that OLSR has good results in terms of scalability in networks with up to 70 nodes, preferably with a moderate node speed (e.g., pedestrian speed) and where the number of traffic flows is also moderate. However, OLSR's performance decreases in large heterogeneous ad hoc networks.

Additionally, OLSR does not differentiate the capabilities of its member nodes and, in consequence, does not exploit nodes with higher capabilities. Thus, HOLSR is an approach designed to improve the scalability of OLSR protocol in large-scale heterogeneous networks. The main improvements are a reduction in the amount of topology control traffic and efficient use of high capacity nodes. HOLSR organizes the network in hierarchical clusters. This architecture allows to reduce the routing computational cost, i.e., in case a link is broken only nodes inside the same cluster have to recalculate their routing table while nodes in different clusters are not affected.

In HOLSR, nodes are organized in clusters according to their capacities. The network hierarchical architecture is illustrated in Fig. 1. At level 1, we have low-capability nodes with one interface represented by circles. Nodes at the topology level 2 are equipped with up to two wireless interfaces, designated by squares. Nodes at level 2 employ one interface to communicate with nodes at level 2 and one interface to communicate with nodes at level 1 or 3. Nodes at level 3, designated by triangles, represent

high-capacity nodes with up to three wireless interfaces to communicate with nodes at lower levels. Nodes with more than one interface are selected as cluster heads. In Fig. 1, the notation used to name the clusters reflects the level of the cluster and the cluster head, e.g., C1.A means that the cluster is at level 1 and the cluster head is node A. A node with multiple interfaces is identified at every level with a different interface. For instance, in Fig. 1 node F has two interfaces and can communicate with nodes at levels 2 and 3. Then, F2 and F3 represent node F's interfaces at level 2 and 3 respectively. Node B has three interfaces and establishes communication with nodes at levels 1, 2 and 3 through interfaces B1, B2 and B3 respectively. HOLSR allows formation of multiple clusters and, unlike OLSR, HOLSR nodes can exchange Hello and TC messages exclusively within each cluster. This constraint reduces the amount of traffic information broadcast to the entire ad hoc network.

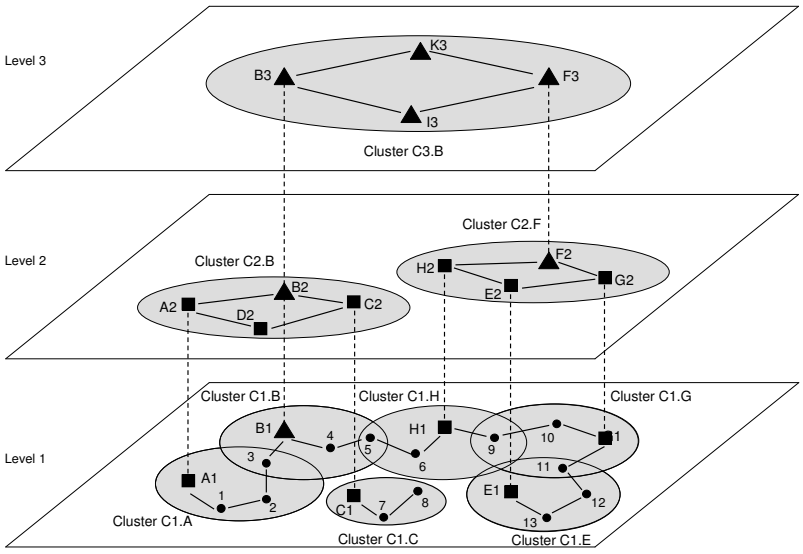


Fig. 1. Example of a hierarchical architecture with heterogeneous nodes.

3.1 Cluster Formation

The topology control information is exchanged between clusters via specialized HOLSR nodes designed as cluster heads. The selection of cluster heads and classification of nodes according to their capabilities are defined at the startup of the HOLSR process. A cluster is formed by a group of mobile nodes -at the same hierarchical level- that

have selected a common cluster head. Nodes can move from one cluster to another and associate to a new cluster head. Any node participating in multiple topology levels automatically becomes the cluster head of the lower-level cluster. In HOLSR, a cluster head declares its status and invites other nodes to join it by periodically sending out CID announcement messages. These messages are transmitted in the same packets together with Hello messages using a message grouping technique. This technique is implemented to reduce the number of packet transmissions. A CID message contains two fields:

- *cluster head*: interface address of the originator of the message.
- *hop count*: distance in hops to the cluster head generating the message.

Once a node has joined a cluster head, it generates a new CID message inviting other nodes farther away to join the cluster. Any given node may receive two or more CID messages, indicating that it is located in the overlapping regions of multiple clusters. In such a case, the node joins whichever cluster is closer in terms of hop count. When a node receives messages from different cluster heads with the same hop count value, it joins the cluster head from which it received the first CID message. Fig. 2 shows the cluster formation process. Nodes A and B are cluster heads and generate CID messages (CID_A and CID_B respectively). The one-hop neighbor nodes join the originator of the message and generate a new message increasing by one the hop count field. Notice that node 2 receives CID messages from CH_A and CH_B with the same hop count value. In this case, node 2 chooses the cluster from which it has received the first message. In the same figure, node 9 joins cluster head A and generates a new CID message with hop count equal to four. Node 11 rejects that message because cluster head B is only three hops away. We refer to neighbor nodes in different clusters, such as nodes 9 and 11, as *border nodes*. Robustness is ensured thanks to a built-in diagnostic feature. Every node registers a timeout value for each CID message received. When a cluster head becomes inactive or moves away, then each neighbor node stops receiving CID messages. Eventually the CID message timeout expires and the CID information becomes invalid.

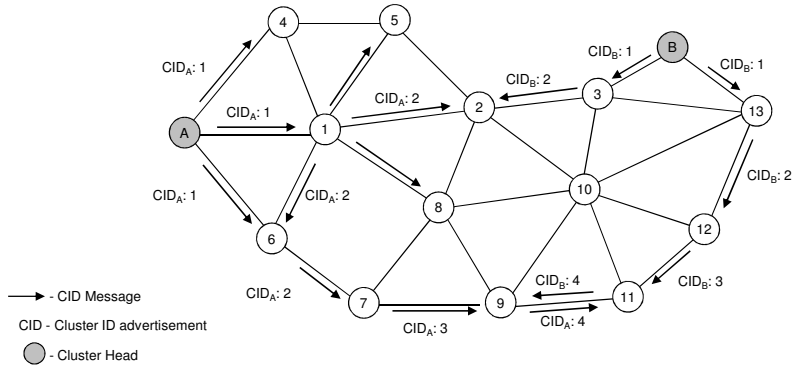


Fig. 2. CID messages.

Thus, each node starts to process new CID messages from other clusters and selects a new cluster head. For instance, in Fig. 2, assuming cluster head CH_A went down, then all nodes attached to it will join cluster head CH_B after receiving new CID messages from that cluster head. If no CID messages are received, then it means that the network is not partitioned in clusters anymore and behaves as the original OLSR protocol.

3.2 Cluster Head Message Exchange

The hierarchical architecture must support the exchange of topology control information between clusters without introducing additional overhead. Thus, **Hierarchical TC (HTC)** messages are generated by the cluster head and used to transmit the membership information of a cluster to higher level nodes. There are three basic types of HTC messages:

- *full membership*: these messages are periodically transmitted by a cluster head to provide information about its cluster members, including any node in lower levels beneath it.
- *update*: to provide information about cluster membership changes. The update HTC is used when a node leaves or joins a cluster.
- *request*: request HTC messages are used when a packet loss has occurred. HTC message carries a sequence number field, which allows a node to request the retransmission of a full membership HTC message.

HTC forwarding is enabled by the MPRs and restricted within a cluster. Nodes at the highest topology level have full knowledge of all nodes in the network and their routing tables are as large as they would be in an OLSR network. However, in lower levels, the size of the routing table of every node is restricted to the size of the cluster and it is smaller than in OLSR. For instance, in Fig. 1 the cluster head A generates an HTC message for the interface A2 (level 2) announcing that nodes 1, 2 and A1 are members of its cluster at level 1. The message is relayed to all nodes at the same level. Then, node B generates an HTC message for the interface B3 (level 3) advertising that nodes 1, 2, 3, 4, 5, 7, 8, A1, B1, C1 (at level 1) and A2, B2, C2, D2 (at level 2) are members of its cluster. Fig. 3 presents a summary of the messages implemented in HOLSR networks.

Messages	Generated by	Retransmitted by	Information reported
Hello	Every node	N/A	One-hop neighbors
TC	MPRs	MPRs	MPR selectors
CID	Cluster heads	N/A	A Cluster head
HTC	Cluster heads	MPRs	Nodes within a cluster

Fig. 3. Summary of control traffic messages in HOLSR networks.

3.3 Topology Control Propagation

Nodes in each cluster select their MPRs to flood control traffic information. Control messages are generated and propagated exclusively within each cluster, unless a node is located in the overlapping zone of several clusters. For example, in Fig. 1 node 2 is within the border of cluster C1.A and may accept a TC or HTC message from node 3 located in cluster C1.B. However, node 2 retains the information without retransmitting it to its cluster. Thus, except for the border nodes, knowledge of nodes about the clusters is restricted to their own cluster. Data transfer between nodes in the same cluster is achieved directly via the information in the routing tables. However, when transmitting data to destinations outside the local scope of a cluster, the cluster heads is used as a gateway. When transmitting data between border nodes in different clusters at the same level, a different strategy might be used. In this situation, the cluster head is not used as a gateway to relay the information. Nearby nodes in different clusters at the same level can communicate directly without following the strict clustering hierarchy. This means that, data transfer between nodes is achieved following three different strategies:

- communication between nodes in the same cluster is achieved via the routing information in their routing tables,
- data transfer between nodes in different clusters is achieved through the cluster heads, but
- if the nodes are neighboring nodes in different clusters at the same topology level, the cluster heads are not used and data packets are directly relayed.

Therefore, HOLSR offers two main advantages (a) messages reflecting local movement are restricted to each cluster (thus, reducing the routing table computation overhead) and (b) an efficient use of high-capacity nodes without overloading them.

4 Cluster Formation Attack against the HOLSR protocol

4.1 Adversary Model

The flow of CID messages is an important vulnerability target. The *hop count* has to be updated every time a new message is retransmitted. Thus, a malicious node might alter this field to unsettle the cluster formation process. The attack, has a bigger impact when the malicious node drastically reduces the *hop count* field. This is because the receiver nodes accept the CID message with the lowest *hop count* value. Thus, when an attacker increases drastically the value, the receiver nodes automatically discard the altered message and accept the valid message from other nodes, as this is described in Section 3.1. If a node that generates a CID message reinitializes the value of the field *hop count*, the receiver nodes may join a farther cluster head and discard valid CID messages from closer cluster heads. Then, we only need to address the case when the *hop count* field is maliciously reduced.

In general, if an attacker is at distance d (in hops) from a cluster head CH_i, and generates a new CID message with *hop count* value j, the nodes with *hop count* greater or equal to $j + \frac{d}{2}$ from the CH_i are potentially affected. For instance, Fig. 4 (a) shows

the correct propagation of CID messages. Fig. 4 (b) shows an example of the attack. In Fig. 4 (b), M_1 is a malicious node at distance six hops from cluster head CH_B. M_1 receives CID messages from CH_B and CH_A, and generates a new CID message assigning an incorrect value to the field *hop count*, i.e., *hop count* is set to two. Thus, all nodes at distance greater or equal to four hops (nodes *2* and *3*) process the message and incorrectly join CH_A. Notice that the lowest value that can be used to reinitialize the field *hop count* is two because CID messages with field *hop count* equal to one are generated exclusively by the cluster heads. Additionally, we consider that the attacker only has one interface, it can not impersonate a cluster head and it only modifies the *hop count* value. In the following section, we present our proposed solution to handle this problem.

5 Handling the Attack with the use of Hash Chains

We describe in this section a security improvement over HOLSR based on the use of hash chains [13]. Authentication and integrity is achieved by using hash key chains. For instance, in [10], Lamport proposes a method of user password authentication based on a secure one-way hash function. We do not attempt to address authentication, but the integrity of the messages. A one-way hash chain is based on a one-way hash function h that is applied n times to a unique value x. Hash functions are relatively easy to compute and can be applied to a block of data of any size.

A hash function can be applied to a block of data of any size and produce a fixed-length output. According to [13], a *strong one-way* hash function h must have the following properties:

1. The **one-way** property implies that for any given value $h(x)$, it is infeasible to find the value of x.
2. The **weak collision resistance** property implies that for any given block x, it is computationally infeasible to find $y \neq x$ such that $h(x) = h(y)$.

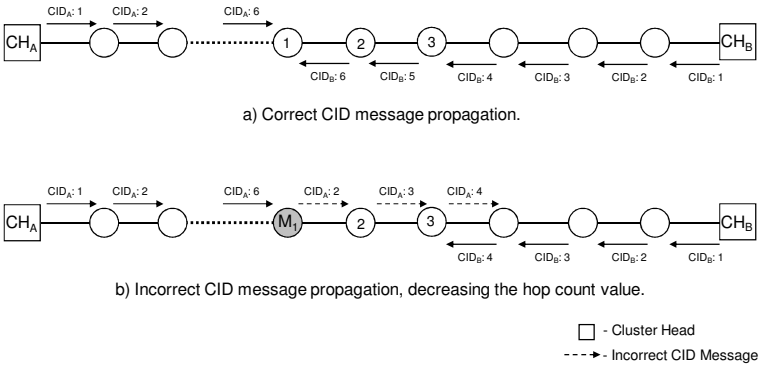


Fig. 4. CID messages.

3. The **strong collision resistance** property implies that it is computationally infeasible to find a pair (x, y) such that $h(x) = h(y)$.

These properties are explained in detail in [13]. Our scheme prevents the attack presented in Section 4 while avoiding the use of computationally expensive cryptographic operations. We use the following notation:

- s_j: is a random value (i.e., a nonce) generated and known exclusively by the cluster head CH_j.
- $h(x)$: is a strong one-way hash function applied to x.
- $h^n(x)$: is a hash chain constructed by applying n times the hash function h to x, $h^n(x) = h(...h(h(x)))$.
- t: is the maximum number of times that a hash function is applied to x.
- Max_j: is the value obtained by applying t times the hash function to a nonce s_j, $Max_j = h^t(s_j)$.
- i: is the distance in hops between the receiver and a cluster head.

Consider that the hash function $h(x)$ and the value of t are known by all nodes in the network. For our purposes, we suppose that the malicious attacker is not able of generating a valid nonce s_j. Algorithm HASH-CHAINED_CID_DISSEMINATION (henceforth HCCD for brevity) formalizes our proposal.

ALGORITHM: HCCD

1. A cluster head (CH_j) generates a random number s_j, i.e., a nonce.
2. CH_j sets the field $i = 1$.
3. CH_j calculates the value $Max_j = h^t(s_j)$.
4. CH_j generates the CID message:$< Max_j, h^i(s_j), i >$.
5. The receiver node verifies that the sender node is i-hops away by applying the following criteria:
   - If $Max_j = h^{t-i}(h^i(s_j))$, then the CID message is valid.
   - Else, the receiver node discards the CID message.
6. If the CID message is valid, then the receiver node generates a new CID message with the hop count increased by one and applying the hash function to $h^i(s_j)$: $< Max_j, h(h^i(s_j)), i + 1 >$.

Algorithm HCCD works as follows: firstly, a valid cluster head (CH_j) generates a random number s_j, i.e., a nonce that is only known by the originator of the message. Then, it initializes the hop count field i equal to one and computes the Max_j value by applying t times the hash function $h(x)$ to the nonce s_j, such as Max_j is equal to $h^t(s_j)$. We assume that Max_j and the value of t are known by all the nodes in the network during the execution of the protocol. Additionally, CH_j applies i times the hash function to s_j, to obtain $h^i(s_j)$. Then, CH_j generates a CID message with the fields: $< Max_j, h^i(s_j), i >$. The receiver node verifies that the CID message is valid

by applying $t-i$ times the hash function to $h^i(s_j)$ and comparing the result with Max_j. Therefore, if Max_j is equal to $h^{t-i}(h^i(s_j))$, then the hop count value i has not been altered and the received CID message is valid. Finally, the receiver node joins CH_j until it receives a CID message from a different cluster head with a lower hop count value. In the mean time, the receiver node generates periodically CID messages announcing its cluster head and the hop count distance to reach it, i.e., $< Max_j, h(h^i(s_j)), i+1 >$.

Theorem 1 *Given a HOLSR network applying the algorithm* HCCD *for the dissemination of CID messages, such that malicious nodes in the network are not able of generating a valid nonce s, h is a strong one-way hash function, i is the distance in hops to reach a cluster head j and Max_j is a value obtained by applying t times h to the nonce s_j. Then algorithm* HCCD *guarantees that a malicious node cannot generate a valid CID message with a hop count value $k \neq i$, such that $Max_j = h^{t-k}(h^i(s_j))$.*

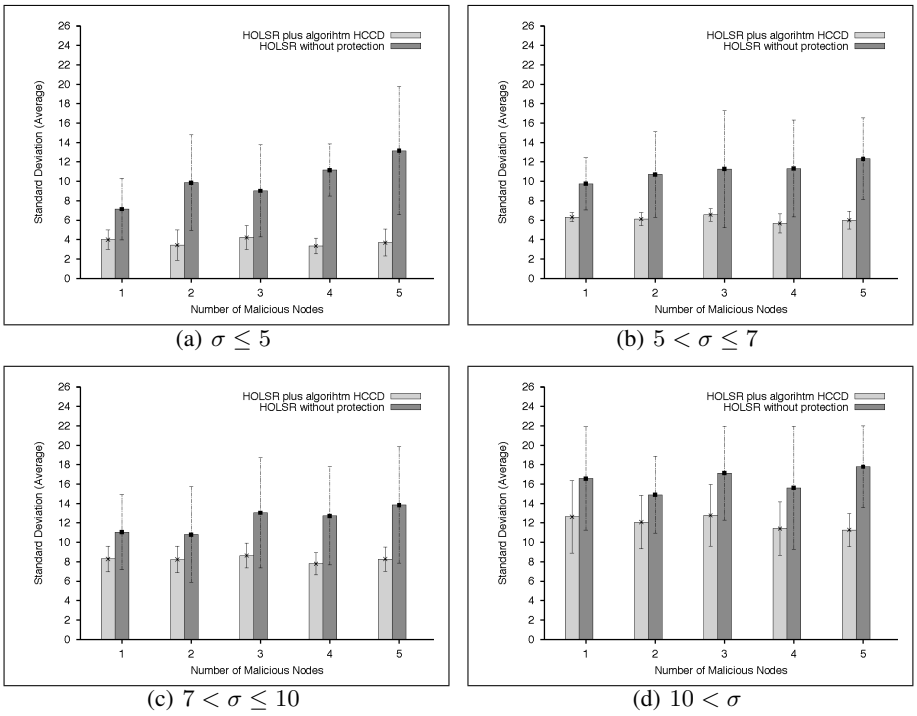
Proof According to algorithm HCCD , $Max_j = h^t(s_j)$ and a CID message is valid if $Max_j = h^{t-i}(h^i(s_j))$. Then, let us assume that there exists a value $k \neq i$ such that $Max_j = h^{t-k}(h^i(s_j))$. Thus, $h^t(s_j) = h^{t-k}(h^i(s_j))$. Then, function h does not meet the **weak collision resistance** property of strong one-way hash functions due to $h^{t-k}(h^i(s_j))$ and $h^{t-i}(h^i(s_j))$ are both equal to Max_j. Therefore, $h^t(s_j)$ is equal to $h^{t-k}(h^i(s_j))$ only if k is equal to i.

□

6 Results and Simulations Setup

In this section, we describe the experiments and results after assessing the effectiveness of our proposed countermeasure to the cluster formation attack presented in Section 4. We conducted our experiments using the NS-3 simulator [5], version 3.9. We modified the original OLSR code developed by Ros and Carneiro to implement the hierarchical approach (i.e., HOLSR), attack, and countermeasure described in Section 5. We tested our algorithm in an HOLSR network with two levels, 200 nodes in the first level with only one interface and four nodes with up to two interfaces (i.e., cluster heads). The transmission range for nodes in the first level and second level is 120 m and 500 m respectively. Nodes at the first level are placed following a uniform distribution in an area of 1000 m by 1000 m. We assume that the administrator of the network can decide the best criteria to distribute the cluster heads. Thus, we divide our scenario in four quadrants and place a cluster head in the center of each of them. We also assume that the malicious node knows the position of the cluster heads and sets itself in the border of two different clusters to maximize the impact of an attack. We also assume that the malicious nodes do not collude to perform an attack, no data traffic is generated and all the scenarios are static.

In an ideal scenario, the number of nodes per cluster must be equally balanced. However, due to the position of the nodes in the network this is not always possible. Additionally, the presence of misbehaving nodes may disproportionately increase the imbalance of the number of nodes per cluster. We compute the average of the standard



(a) $\sigma \leq 5$ (b) $5 < \sigma \leq 7$

(c) $7 < \sigma \leq 10$ (d) $10 < \sigma$

Fig. 5. Standard deviation of the average number of nodes per cluster testing different HOLSR networks with up to five malicious nodes and applying algorithm HCCD .

deviation of the number of nodes per cluster with up to five malicious nodes launching the cluster formation attack. Then, we compare the average of the standard deviation of the number of nodes per cluster on a series of simulated HOLSR networks without protection and the average of the standard deviation of the number of nodes per cluster but applying algorithm HCCD . We use the standard deviation (σ) as a measure of dispersion. The standard deviation is computed with the formula: $\sigma = \sqrt{\frac{\sum(x_j - \bar{X})^2}{N_{CH}}}$ and expressed in the same units as the data, where x_j is equal to the number of nodes in the cluster j, $\bar{X}$ is the average of nodes per cluster, i.e., total number of nodes (n) over the number of clusters (N_{CH}). In our experiments, the standard deviation formula can be simplified as follows: $\sigma = \sqrt{\frac{\sum(x_i - 50)^2}{4}}$. Fig. 5 shows how the CID attack affects the average of the standard deviation of 100 experiments with different topologies and 90% confidence interval.

To present our results, we consider two factors that affect the distribution of nodes per cluster: the network topology and the presence of malicious nodes. Thus, Fig. 5(a) shows the experiments where $\sigma \leq 5$, this means that the distribution of nodes per cluster in the network is relatively balanced when there is no malicious nodes. Fig. 5(b) shows the experiments where the distribution of nodes per cluster is less balanced due to the network topology, i.e., $5 < \sigma \leq 7$, Fig. 5(c) shows the experiments where $7 < \sigma \leq 10$,

and Fig. 5(d) shows the experiments where $10 < \sigma$. In each case, the first column shows the average of the standard deviation with malicious nodes and implementing algorithm HCCD . Therefore, the distribution of the nodes per cluster is affected only by the network topology. The second column represents the average of the standard deviation with malicious nodes but without applying our algorithm. Notice that the average of the standard deviation and size of the confidence interval increase because the number of nodes per cluster is less balanced due to the network topology plus the effect of the attack.

7 Related Work

In this paper, we reviewed the cluster formation phase in HOLSR networks, however other hierarchical approaches based on the OLSR protocol are also vulnerable during the cluster formation stage, for instance: cluster OLSR (C-OLSR) [12] proposed by Ros et al. assumes that a cluster formation mechanism has been executed, nevertheless any security measures during this stage are proposed. The Multi-level OLSR Routing using the Host and Network Association (HNA) messages Extension (MORHE) [15] presented by Voorhean et al. does not specify any secure cluster formation mechanism therefore like C-OLSR, the cluster formation stage is vulnerable to malicious attacks.

A tree-based logical topology [3, 2] to provide hierarchical routing is presented by Baccelli, this approach implements *Branch* messages to form and maintain a tree-based structure. With a *Branch* message a node specifies information such as its identity (the *NodeID* field), the tree where it belongs to (the *TreeID* field) and its parent in the tree (the *ParentID* field). Additionally, the *Depth* field indicates the distance of the node to the root. This approach does not propose any security measure to protect the integrity of *Branch* messages, so an attacker can easily alter the value of the *Depth* field in *Branch* messages. A hierarchical approach similar to HOLSR which uses HNA messages instead of HTC messages for inter-cluster communication is proposed by Arce et al. in [1]. Like HOLSR, cluster heads are predefined then is not necessary a cluster head selection algorithm, however a cluster formation mechanism is needed. Therefore, any strategy that uses the distance in hops as the main parameter to invite other nodes to join a particular cluster head will be affected by the attack presented in this paper.

In [6], Hong et al., present a solution to secure OLSR (SOLSR). Authors present a wormhole detective mechanism and authentication to strengthen the neighbor relationship establishment. Thus, they use digital signature to ensure the non-mutable fields and hash chains to secure Hop Count and TTL fields. Their solution is similar to our proposed algorithm, however it is implemented in *flat* OLSR to protect only standard control traffic messages. Kush and Hwang, present in [9] a mechanism based in hash chains to secure AODV. Then Hashing is done for route request and reply messages to achieve complete security in terms of availability, integrity and authentication, minimal overhead, network performance in terms of throughput and node mobility. Similarly, Hu et al., [7] propose a hashing mechanism to secure distance vector routing protocols.

8 Conclusion

HOLSR has been designed to improve scalability in MANETs. However, the protocol has been designed without security countermeasures. In this paper, we propose a method to protect the cluster formation stage in HOLSR networks. Our mechanism prevents an attacker from maliciously altering the hop count field in CID messages. Thus, we present an algorithm based on hash chains that allows to detect and discard invalid CID messages. Our experiments show that the distribution of nodes is less balanced when the hop count in CID messages is maliciously altered. We also show that we can prevent this kind of attacks by applying our proposed algorithm. Notice that our mechanism, can be also applied in other hierarchical routing protocols for MANETs that utilize mutable information such as the hop count or TTL fields to organize the network in clusters.

Acknowledgments — The authors graciously acknowledge the financial support received from the following organizations: Natural Sciences and Engineering Research Council of Canada (NSERC), Mathematics of Information Technology and Complex Systems (MITACS), Institut Telecom, Spanish Ministry of Science and Innovation (grants TSI2007-65406-C03-03 E-AEGIS and CONSOLIDER-INGENIO CSD2007-00004 ARES), National Council of Science and Technology (CONACYT), and Ministry of Education of Mexico (SEP, Program for Academic Improvement).

References

1. P. Arce, J.C. Guerri, A. Pajares, and O. Lázaro. Performance evaluation of video streaming over ad hoc networks using flat and hierarchical routing protocols. *Mobile Networks and Applications*, 13(3-4):324–336, 2008.
2. E. Baccelli. OLSR scaling with hierarchical routing and dynamic tree clustering. In *IASTED International Conference on Networks and Communication Systems (NCS), Chiang Mai, Thailand*, March 2006.
3. E. Baccelli. OLSR trees: A simple clustering mechanism for OLSR. *Challenges in Ad Hoc Networking, IFIP International Federation for Information Processing*, 197:265–274, 2006.
4. T. Clausen and P. Jacquet. Optimized link state routing protocol (OLSR), RFC3626. IETF Internet Draft, http://www.ietf.org/rfc/rfc3626.txt, October 2003.
5. T. Henderson et. al. The NS-3 network simulator. Software package retrieved from http://www.nsnam.org/, 2011.
6. F. Hong, L. Hong, and C. Fu. Secure OLSR. *Advanced Information Networking and Applications, International Conference on*, 1:713–718, 2005.
7. Y.-C. Hu, D. B. Johnson, and A. Perrig. Sead: secure efficient distance vector routing for mobile wireless ad hoc networks. *Ad Hoc Networks*, 1(1):175 – 192, 2003.
8. P. Jacquet, P. Muhlethaler, T. Clausen, A. Laouiti, A. Qayyum, and L. Viennot. Optimized link state routing protocol for ad hoc networks. In *IEEE International Multi Topic Conference, 2001. IEEE INMIC 2001. Technology for the 21st Century. Proceedings*, pages 62–68. Lahore University of Management Sciences, Pakistan, December 2001.
9. A. Kush and C. Hwang. Proposed protocol for hash-secured routing in ad hoc networks. *Masaum Journal of Computing (MJC)(ISSN 2076-0833) Volume*, 1:221–226, 2009.
10. L. Lamport. Password authentication with insecure communication. *Commun. ACM*, 24:770–772, November 1981.

11. D. Palma and M. Curado. Inside-Out OLSR Scalability Analysis. In *Proceedings of the 8th International Conference on Ad-Hoc, Mobile and Wireless Networks*, ADHOC-NOW '09, pages 354–359, Berlin, Heidelberg, 2009. Springer-Verlag.
12. F.J. Ros and P.M. Ruiz. Cluster-based OLSR extensions to reduce control overhead in mobile ad hoc networks. In *Proceedings of the 2007 international conference on Wireless communications and mobile computing*, pages 202–207. ACM, 2007.
13. W. Stallings. *Cryptography and Network Security, Principles and Practices*. Pearson Prentice Hall, 2006.
14. L. Villasenor-Gonzalez, Y. Ge, and L. Lamont. HOLSR: A hierarchical proactive routing mechanism for mobile ad hoc networks. *IEEE Communications Magazine*, 43(7):118–125, July 2005.
15. M. Voorhaen, E. Van de Velde, and C. Blondia. MORHE: A transparent multi-level routing scheme for ad hoc networks. In K. Al Agha, I. Guérin Lassous, and G. Pujolle, editors, *Challenges in Ad Hoc Networking*, volume 197 of *IFIP International Federation for Information Processing*, pages 139–148. Springer Boston, 2006.