Hindawi Publishing Corporation
International Journal of Distributed Sensor Networks
Volume 2013, Article ID 521497, 20 pages
http://dx.doi.org/10.1155/2013/521497

Research Article

On Lightweight Intrusion Detection: Modeling and Detecting Intrusions Dedicated to OLSR Protocol

Mouhannad Alattar,[1] Françoise Sailhan,[2] and Julien Bourgeois[1]

[1] *UFC/FEMTO-ST Institute, UMR CNRS 6174, 25201 Montbéliard, France*
[2] *Cédric Laboratory, CNAM, 75003 Paris, France*

Correspondence should be addressed to Mouhannad Alattar; mouhannad.alattar@univ-fcomte.fr

Received 1 March 2013; Accepted 23 April 2013

Academic Editor: S. Khan

Mobile ad hoc networks mostly operate over open, adverse, or even hostile environments and are, therefore, vulnerable to a large body of threats. Conventional ways of securing network relying on, for example, firewall and encryption, should henceforth be coupled with advanced intrusion detection. To meet this requirement, we first identify the attacks that threaten ad hoc networks, focusing on the Optimized Link State Routing Protocol. We then introduce IDAR, a signature-based Intrusion Detector dedicated to ad hoc routing protocols. Contrary to existing systems that monitor the packets going through the host, our system analyses the logs so as to identify patterns of misuse. This detector scopes with the resource-constraints of ad hoc devices by providing distributed detection; in particular, depending on the level of suspicion and gravity, in-depth cooperative diagnostic may be launched. Simulation-based evaluation shows limited resource consumption (e.g., memory and bandwidth) and high detection rate along with reduced false positives.

1. Introduction

Securing mobile [Ad hoc] networks (MANETs for short) is particularly challenging because these networks often operate in adverse or even hostile environments [1, 2]. In addition, they are characterized by the open radio-based medium of communication [3], the dynamic topology [4, 5], the lack of centralized administration/security enforcement points (e.g., switches and routers) [6], the low degree of physical security of the mobile nodes, and the limited resources (e.g., energy, bandwidth) [7]. Hence, MANET is much more vulnerable to attacks than the traditional infrastructure-based network [8]. Conventional prevention techniques, for example, firewall, encryption [9], and authentication [10], cannot totally eliminate the attacks. Indeed, they address the outsiders but they are less helpful for protecting against the insiders; if a legitimate node becomes compromised, then all the secrets associated with this latter would be open to attackers. Moreover, new attacks emerge and find usually a way to penetrate the aforementioned techniques. Therefore, there is a need for a reactive mechanism, such as Intrusion Detection Systems (IDSs for short) [11], which constitutes a second line of defense.

As a first step upon this goal, we survey the attacks that have been reported in the literature. More specifically, we focus on the attacks targeting the Optimized Link State Routing (OLSR for short) [12]; its central role, namely, determining multi hop paths among devices, designates the ad hoc routing protocol as one of the favorite targets for the attackers [13]. Our attempt is not restricted to describing a bunch of attacks. Instead, we categorize and detail each attack relying on a representation/formalism that captures the complexity and temporal dependencies between each of the constituting subtasks. While describing attack, we attempt to circumvent the general form of this attack so as to keep to a minimum the detection that fails due to a varying attack. Based on these modeled attacks, we further implement one attack, challenge and derive appropriate intrusion detection.

Recent works show that attack may be identified by detecting a deviation to the correct behavior (anomaly detection); this correct behavior is either hand-specified relying on a protocol description (typically a Request For

Comment (RFC for short)) as in, for example, [14] or automatically built/analyzed using machine learning or data mining techniques, for example, [15]. The difficulty involved in automatically modeling the behavior of such dynamic network leads to a large number of false positives that may be reduced by coupling automatic and specification-based anomaly detection [16]. Moreover, the used techniques are mostly characterized by their intensive calculation and, henceforth, the extensive consumption of resources [2]. An alternative consists in describing the way an intruder penetrates the system (by establishing intrusion signature) so as to detect any behavior that is close to this previously defined signature. Little attention has been centered on signature-enabled detection (also called misuse detection) in MANET [17], although it guarantees a high rate of detection along with a limited number of false alarms [18].

We propose IDAR, a log-, signature-based Intrusion Detector dedicated to ad hoc routing protocols. IDAR does not necessitate sniffing/inspecting the traffic as it is the case with the majority of other IDSs. Thus, it avoids the permanent strain of energy, bandwidth [19], and computational power [20] that accompanies traffic sniffing and analyzing. Our system rather takes advantage of the audit logs that are generated by the routing protocol so as to detect evidences of intrusion attempts. In practice, a sequence of events are extracted from logs so as to be matched against a set of predefined intrusion signatures—a signature is thought of as a pattern of events that characterizes an intrusion.

Main challenges stem from the need to keep to a minimum the number of diagnostics and the computational load related to the intrusion identification while minimizing the traffic generated when gleaning intrusion evidences. This calls for developing a lightweight and distributed intrusion detection system that scopes with the cooperative nature of ad hoc networks and the device resource constraints. Towards this goal, IDAR is designed to be a distributed and cooperative detection system, which parses logs as close as possible from the device that generates them so as to diminish the number of long-distant communications. Furthermore, we propose to categorize the intrusion evidences according to their level of gravity/suspicion. Such categorization enables us to carefully plane the diagnostic; an in-depth diagnostic is initiated only when a sufficient degree of suspicion exists and terminated as soon as a result is obtained. Thus, we guarantee keeping to a minimum the number/the duration of diagnostics. The performance of IDAR is evaluated in a simulated MANET that is coupled with virtual machines. Such coupling permits us measuring both the detection accuracy and the amount of consumed resources.

The reminder of this paper is organized as follows. We first survey attacks on ad hoc network, breaking down the successive steps that characterize conquering attacks (Section 2). Grounded upon the defined intrusion signatures, we present a distributed, log- and signature-based intrusion detection system (Section 3) and evaluate its performance (Section 4). Then, we conclude this paper with a summary of our results along with directions for future works (Section 6).

2. Vulnerabilities

Physical, data link, and network layers are all subject to vulnerabilities. Whereas in the physical and data link layers, vulnerabilities are common to IEEE 802.11 wireless networks, vulnerabilities in network layer are specific to ad hoc networks. More specifically, Zeroconf and routing protocols constitute the main target of attacks [13]. The reason is threefold. First, no security countermeasure is specified or implemented as a part of the drafts or RFCs (request for comments) proposed through the Internet Engineering Task Force MANET (http://www.ietf.org/dyn/wg/charter/manet-charter.html) or Zeroconf (http://www.zeroconf.org/) working groups. Second, the absence of a centralized infrastructure complicates the deployment of preventive measures, for example, firewalls or key/authentication infrastructures. Third, any device may operate as router, which facilitates the manipulation of multihops messages as well as the compromising of the routing functionality. Attacks on those protocols fall into two main categories, passive versus active [21]. With the former, an intruder intercepts the traffic in order to reveal useful information (e.g., the roles played by the nodes and their locations) whereas with the latter, an unauthorized action is attempted [22]. Active attacks are further sub-classified according to their unauthorized actions as follows [23].

(i) *Drop attack* consists in dropping routing message(s).

(ii) *Modify and forward attack* modifies received routing message(s) before forwarding it.

(iii) *Active forge attack* proactively generates deceptive routing message(s).

Although the above attacks threaten both routing and Zeroconf protocols, we hereafter concentrate on routing protocol and illustrate our presentation by exemplifying attacks on a proactive protocol called OLSR.

2.1. Background on the OLSR Protocol. OLSR aims to maintain a constantly updated view of the network topology on each device. One fundamental is the notion of multipoint relay (MPR): each device selects a subset of the 1-hop neighbors, called MPRs, that is responsible for forwarding the control traffic in the entire network. The basic idea is to select the minimum number of MPRs that cover the two-hops neighbors so as to reduce the number of nodes retransmitting control messages and hence keep to a minimum the bandwidth overload. Note that a redundant MPR may be selected as to increase the reachability but this leads to increase the overhead. In practice, a node N selects the MPRs among the 1-hop neighbors that are announced (in addition, link layer information provided by, e.g., the IEEE 802.11 protocol, may be used by a node to update its own routing tables) in periodic heartbeat messages, termed hello messages. Then, a *Topology Control* message (TC for short), intended to be diffused in the entire network (the message Time To Live field (TTL for short) can be used so as to limit message diffusion), is created by the selected MPR(s). In TC message, a MPR declares the nodes (including N) that selected itself to act as MPR. Thanks to TC messages, any device computes the shortest path (in

term of the number of hops) to any destination, such path being represented as a sequence of MPRs. In addition to the above, last versions of the protocol specification support a node holding several network interfaces which are declared (if many) in a so-called MID (Multiple Interface Declaration) message. This message is broadcasted on a regular basis by MPRs so that one another maps multiple interfaces of a given node with the main address of this latter, hence permitting a unique identification. The aforementioned functionalities compose the core of OLSR. Additional extensions have been devised in compliance with the above-summarized OLSR specification. Examples include (i) dealing with the nodes that commit (or not) to carry the traffic for others and (ii) supporting interconnection of an OLSR MANET with another routing domain, for example, OSPF-enabled routing domain. With the former, node advertises (in hello message) its willingness to carry/forward traffic. With the latter, OLSR is extended to import (and, resp., export) the routes provided by other routing protocols (resp., OLSR). For this purpose, any gateway with associated host(s) and/or network(s) generates periodically a HNA message including those host(s) and/or network(s) (i.e., the related network address and the netmask); this message is further disseminated by MPRs. Such an auxiliary function is enabled by providing a basic layout of any OLSR packet and by ignoring unknown (and hence not handled) packets. Overall, these core and auxiliary functionalities are together subject to a variety of attacks.

Recently, a new version of OLSR (so-called OLSRv2) is specified in an Internet-draft [24]. OLSRv2 distinguishes itself from its predecessor by considering the link metric rather than hop count for selecting the shortest path. It is supposed that OLSRv2 would have more modular and flexible architecture that facilitates add-ons extensions for, for example, security, QoS, and multicast [25]. It will also possess a simplified packet format and reduced-size messages due to address compression [26]. However, both, OLSR and OLSRv2, retain the same basic algorithms and mechanisms.

It is worth mentioning that several enhanced versions of OLSR have been proposed. Some of them tackle the security issue and use signature, hash chain, or encryption schemes in order to provide the security to OLSR [27, 28]. In general, these versions aim to ensure (i) the integrity of the routing information, that is, preventing the unauthorized modifications of the routing messages by the intermediate nodes on the path from the source to the destination and (ii) the node-to-node authentication in order to prevent identity spoofing. However, they do not address, for example, the case where the source node is itself compromised and hence it generates incorrect routing messages. Thus, they are still incapable of preventing some types of attack [29]. Therefore, there is always a need to employ a detection mechanism so as to handle the attacks that would success penetrating the prevention techniques.

2.2. Attacks Targeting the OLSR Protocol. The attacks threatening OLSR are hereafter detailed and classified according to the model introduced in [30]. This model provides the level of expressiveness necessary to specify the relationship between the actions and their related consequences. We further enrich this model with temporal annotations (Table 1). As a consequence, complex attacks, their constituting actions and consequences are temporally and successfully depicted and categorized as drop attack (Section 2.2.1), active forge attack (Section 2.2.2), and modify and forward attack (Section 2.2.3).

Table 1: Notations.

Communication	
$Y \xrightarrow{M_t} X$	At t, Y sends a message M that is received by X
$Y \xrightarrow{M_t}$	At t, Y sends a message M
$Y \overset{M_t}{\nrightarrow}$	Y does not send a message M at t
$\xrightarrow{M_t} X$	X receives a message M at t
$\overset{M_t}{\nrightarrow} X$	X does not receive a message M at t
Parameters	
$\triangle t$, $\triangledown t$	Period of time
$\mathcal{F}$	Set of malicious nodes
NS_X	Set of 1-hop neighbors of X
sq	Sequence number
hc	Hop count
w_X	Willingness of X to forward packets on behalf of others
MPR_x	MPRs of X
$\text{Sel}_{\text{MPR}_x}$	MPR selectors set of X
Messages	
Hello	Hello message
TC	Topology Control message
MID	Multiple Interface Declaration
HNA	Host and Network Association
CM	Control Message
RM	Received Control message
FM	Control message intended to be forwarded

2.2.1. Drop Attack. In practice, a drop attack consists of an intruder that drops a control message instead of relaying it. This dropping has an impact only if the dropped control message is intended to be forwarded; as illustration, a suppressing of a hello message that is broadcasted over one hop, has no consequence. Thus, with OLSR, threatened messages are restricted to the messages that are created and/or broadcasted by a MPR so as to be rediffused by other MPRs, that is, Topology Control (TC), Multiple Interface Declaration (MID), and Host and Network Association (HNA) messages. More particularly, let us consider a host S that sends a control message which is intended to be forwarded. This message, which is originated at t ($S \xrightarrow{\text{FM}_t} I$), is received by I that drops it. In practice, I drops a control message if I does not forward this message during a period $|t' - t|$ less than the maximum allowed (any control message is characterized by an emission interval as well as a holding time in the routing tables. These two related values participate in setting a validity time that

should be herein considered in conjunction with the message type) period $\triangle t$:

$$S \xrightarrow{\mathrm{FM}_t} I, I \overset{\mathrm{FM}_{t'}}{\nrightarrow}, \left|t' - t\right| > \triangle t \Downarrow I \in \mathcal{I}. \tag{1}$$

Such a dropping is naturally applied on any packet that is empty, expired, duplicated, or out of order. In addition, restrictive forwarding is provided, meaning that in practice (only) the 1-hop neighbor(s) of S that have been selected by S to act as MPR(s), forward S's messages. Apart from the aforementioned conditions that lead to a convenient dropping, the remaining reason motivating a dropping is threefold. An intruder (or a compromised node), a selfish or a malfunctioning node disturbs the routing calculation by dropping a control message that should be forwarded. The attempt to drop any packet is termed *black hole*. A selective dropping (named *gray hole*) is detected by taking into account additional fine-grained or discriminative criteria, including the choice of specific final/1-hop-away destination or source, percentage/rate of packets received or forwarded, and message type. Rather than dropping the traffic, an opposite misbehavior consists in introducing falsified routing information.

2.2.2. Active Forge Attack. An active forge attack comes from a node that introduces novel deceptive routing messages. Among others, the *broadcast storm* stems from forging control messages so as to exhaust resources (e.g., energy) and saturate the communication medium. For this purpose, an intruder I forges a large number of control messages CM within a short period of time $\bigtriangledown t$ (see Expression 2). This attack may be local (e.g., targeting a 1-hop neighborhood) or global (e.g., relying on the multihops broadcasted messages). It may also be conducted in a distributed manner with several colluding nodes so as to simultaneously emit (a large number of) control messages. Note that in order to prevent a broadcast storm resulting from the involuntary synchronization of some emitting nodes, a delaying of the control messages is recommended in the RFC; a jitter (typically, a random value) is subtracted from the interval at which control messages are generated (and also forwarded):

$$I \xrightarrow{\mathrm{CM}_t}, I \xrightarrow{\mathrm{CM}'_t}, \left|t' - t\right| < \bigtriangledown t \Downarrow I \in \mathcal{I}. \tag{2}$$

A special type of *broadcast storm* is *routing table overflow* attack that threatens especially the proactive routing protocols (e.g., OLSR) as they periodically update the routing information [31]. Herein, one or several colluding intruders prevent the well-behaving nodes from discovering new existing routes by dumping the network with route advertisements to nonexisting nodes. In general, *broadcast storm* constitutes a kind of denial of service that is characterized by a high visibility. Therefore, it is typically combined to masquerading—although less intrusive attacks may also be preferred. In practice, masquerading lies in sending a control message CM including a switched identification ($I \xrightarrow{\mathrm{CM}(I)_t}, I \xrightarrow{\mathrm{CM}(S)_t}$). Note that this case is distinguished from a node that holds several interfaces and that advertises those later in a dedicated MID message. Apart from masquerading a denial of service, identity spoofing may be intended to create conflicting route(s) and potentially routing loop(s). In this latter case, spoofing the identity of a node S, belonging to the network, is necessary. As pointed out in [32], the spoofing attack may also be coupled with a modification of the willingness field so as to impact the MPR selection (Expression 3). In practice, if I emits a hello message including an originator address already assigned to another node S ($I \xrightarrow{\mathrm{hello}(S)} D$), then I modifies the local topology as seen by its 1- and 2-hops neighbors. In addition, if I modifies the willingness field w'_S of S (with $w'_S \neq w_S$), then the selection of S as MPR is impacted; recall that MPRs are selected among the nodes with highest willingness, and in case of multiple choices, the node which provides a reachability to the maximum number of nodes is primarily selected. For instance, a node whose willingness attribute set to $will\_never = 0$ (resp., $will\_always = 7$), is never (resp., always) selected as MPR; that is, $w'_S = will\_never \Rightarrow S \notin \mathrm{MPR}_D$ (versus $w'_S = will\_always \Rightarrow S \in \mathrm{MPR}_D$):

$$S \xrightarrow{\mathrm{hello}(S,w_S)_t} D, I \xrightarrow{\mathrm{hello}(S,w'_S)t'} D, \left|t' - t\right| < \triangle t, w_S \neq w'_S \Downarrow I \in \mathcal{I}, \quad S \text{ is spoofed}, \quad w'_S = will\_never \Longrightarrow S \notin \mathrm{MPR}_D, \quad w'_S = will\_always \Longrightarrow S \in \mathrm{MPR}_D. \tag{3}$$

Active forge attacks are not restricted to identity spoofing (possibly coupled with a tampering of the willingness field). They also cover the tampering of control messages including incorrect adjacent links (hello messages), topology information (TC messages), and network interfaces (MID and HNA messages). In the following, we detail each of those. In the first case (Expression 4), I forges a hello message, which declares a list of 1-hop and symmetric neighbors NS'_I differing from the real set NS_I. A symmetric 1-hop neighbor, or simply symmetric neighbor, corresponds to an adjacent node with which communication is bidirectional. In practice, it means

that the hello message exchanged by two nodes is heard by both of these later:

$$\begin{gathered} I \xrightarrow{\text{hello}\left(NS_I'\right)_t} S, NS_I' \neq NS_I \\ \Downarrow \\ I \in \mathcal{F}. \end{gathered} \tag{4}$$

Whenever forging the set of neighbors NS_I', the attacker has three options.

(1) Declaring a nonexisting node as a symmetric neighbor implies that I (or another misbehaving node) is further selected as MPR (Expression 5). Indeed, if I advertises a non-existing node N ($N \notin \mathcal{N}$ with $\mathcal{N}$ defining the set of nodes composing the OLSR network (according to the OLSR RFC [12], messages can be flooded into the entire network (with a maximum network diameter defined by the message Time To Live field, TTL for short), or flooding can be limited to nodes within a diameter (defined in terms of number of hops) from the originator of the message. For the sake of clarity, let $\mathcal{N}$ represent the network in both cases)), I ensures that no other (well-behaving) MPR claims being a 1-hop symmetric neighbor of N. Recall that the set of MPRs is selected so that all the 2-hops and symmetric neighbors are covered; I is hence selected as MPR:

$$\begin{gathered} S \xrightarrow{\text{hello}\left(NS_S\right)_t}, I \xrightarrow{\text{hello}\left(NS_I'\right)_{t'}} S, \left|t' - t\right| < \triangle t, \\ \exists N \in NS_I' \ni: N \notin \mathcal{N} \cap NS_I \\ \Downarrow \\ I \in \mathcal{F}, \\ \exists I' \in \mathcal{F} \cap NS_S \ni: I' \in \text{MPR}_S, \\ \text{Card}\left(NS_I' \setminus \left(NS_I' \cap \mathcal{N}\right)\right) > 0. \end{gathered} \tag{5}$$

This assertion is verified as long as no other misbehaving neighbor of S is claiming the same. Overall, inserting at least one non-existing neighbor ($\exists N \in NS_I' \ni: N \notin \mathcal{N} \cap NS_I$) guarantees that a misbehaving node I' (with $I' \in \mathcal{F}$) is selected to act as MPR of S ($\exists I' \in \mathcal{F} \cap NS_S \ni: I' \in \text{MPR}_S$). In addition to the above, the connectivity of I is also artificially increased (Card($NS_I' \setminus (NS_I' \cap \mathcal{N})) > 0$).

(2) Declaring that an existing node is a symmetric 1-hop neighbor whereas that node is far away (i.e., is not a neighbor) or is a neighbor but that is not symmetric ($\exists X \in NS_I' \cap \mathcal{N} \ni: X \notin NS_I$). This claiming increases artificially the connectivity of I, that is, Card($(NS_I' \setminus NS_I) \cap \mathcal{N}) > 0$:

$$\begin{gathered} S \xrightarrow{\text{hello}\left(NS_S\right)_t}, I \xrightarrow{\text{hello}\left(NS_I'\right)_{t'}} S, \left|t' - t\right| < \triangle t, \\ \exists X \in NS_I' \cap \mathcal{N} \ni: X \notin NS_I \\ \Downarrow \\ I \in \mathcal{F}, \\ \text{Card}\left(\left(NS_I' \setminus NS_I\right) \cap \mathcal{N}\right) > 0, \\ \nexists A \in \mathcal{N} \setminus \mathcal{F} \ni: A \in NS_S \wedge X \in NS_A \\ \Downarrow \\ \exists I' \in \mathcal{F} \ni: I' \in \text{MPR}_S. \end{gathered} \tag{6}$$

If no (well-behaving) MPR covers S ($\nexists A \in \mathcal{N} \setminus \mathcal{F} \ni: A \in NS_S \wedge X \in NS_A$), then at least one misbehaving node (e.g., I) is selected as MPR of S ($\exists I' \in \mathcal{F} \ni: I' \in \text{MPR}_S$). Such insertion typically characterizes an attempt to create a black hole; I introduces a novel path toward M that whenever selected provisions the black hole.

(3) Omitting an existing 1-hop neighbor and symmetric node, P ($\exists P \in NS_I \ni: P \notin NS_I'$), decreases artificially the connectivity of both P and I ($NS_I \not\subseteq NS_I'$). Note that if P has no other connectivity than the one obtained through a misbehaving node, P gets isolated:

$$\begin{gathered} S \xrightarrow{\text{hello}\left(NS_S\right)_t}, I \xrightarrow{\text{hello}\left(NS_I'\right)_{t'}} S, \left|t' - t\right| < \triangle t, \\ \exists P \in NS_I \ni: P \notin NS_I' \\ \Downarrow \\ I \in \mathcal{F}, \\ \exists I' \in \mathcal{F} \cap NS_S, NS_I \not\subseteq NS_I'. \end{gathered} \tag{7}$$

Overall, falsifying the neighboring adjacency by inserting (existing or nonexisting) neighbor(s) and/or omitting real neighbors potentially perverts the local topology seen by S (and more generally by one another) and impacts the selected MPR(s) by S. Nevertheless, note that in order to be selected as MPR of S (or to prevent its selection), there is no need for I to falsify the neighboring adjacency. Recall that, in a hello message, a field, termed *willingness*, designates the node's willingness to carry traffic on behalf of others; I hence prevents (resp., ensures) its selection as MPR, by simply setting its willingness field to the value *will_never* (resp., *will_always*). On the whole, the MPR selection is impacted by either falsifying the topological information included in hello message or by making use of the willingness attribute. Another (perhaps more straightforward) alternative refers to a slander node I declaring (resp., not declaring) itself as MPR although it has not (resp., has) been selected as MPR

[33]. For this purpose (Expression 8), I forges a *Topology Control* (TC) message including an incorrect set of 1-hop symmetric neighbors that have selected I as MPR (so-called MPR selector set). Depending on the level of redundancy required, a MPR advertises:

(1) the MPR selector(s) $\mathrm{Sel}_{\mathrm{MPR}_I}$ of I;

(2) the MPR selector(s) along with the MPR of I: $\mathrm{Sel}_{\mathrm{MPR}_I} \cup \mathrm{MPR}_I$;

(3) the 1-hop neighbors NS_I of I (hence including the MPR selectors and the MPR of I).

Let $\mathscr{A}_I$ represent the set advertised in the TC message: $\mathscr{A}_I = \mathrm{Sel}_{\mathrm{MPR}_I} \vee (\mathrm{Sel}_{\mathrm{MPR}_I} \cup \mathrm{MPR}_I) \vee NS_I$. This attack consists in I advertising an incorrect set $\mathscr{A}'_I$ differing from the real one $\mathscr{A}_I$:

$$\begin{gathered} I \xrightarrow{\mathrm{TC}(\mathscr{A}'_I)_t} S, \mathscr{A}'_I \neq \mathscr{A}_I \\ \Downarrow \\ I \in \mathscr{I}. \end{gathered} \tag{8}$$

In particular, possible falsifications lie in either I inserting a non-existing node, or inserting an existing node but non-MPR selector, or omitting a node belonging to $\mathscr{A}_I$. Upon the reception of a falsified TC message, routing table is corrupted. This corruption contaminates the OLSR network and also any interconnected routing domain. Indeed, a node, well-behaving or not, acting as a gateway exports the wrong OLSR routes. Symmetrically, an intruder may also import incorrect routes to the OLSR domain. This latter attack, termed *sinkhole*, involves an intruder I defining itself as a gateway that provides an access to associated host(s) and/or network(s). This gateway generates periodically a HNA message including those host(s) and/or network(s) (i.e., the related address(es) and netmask(s)). This attack constitutes a generalization of the previously defined forging of corrupted TC messages; a node advertises either non-existing or existing but unreachable nodes or omits advertising reachable nodes. The similarity with the TC active forging lead us not detailing it hereafter.

The aforementioned forge attacks (e.g., link or route spoofing attacks, sinkhole) necessitate to tamper specific message fields while keeping this message syntactically correct. More generally, bogus control messages can be forged, hence creating an implementation-dependent effect. Generally speaking, similar tampering may be performed by an intruder that has acted as MPR prior forwarding a falsified control message.

2.2.3. Modify and Forward Attacks. Modify and forward attacks are characterized by an intermediate, which captures the victim's control message and replays, modifies, or drops this message before forwarding it. In the following, we depict the two former cases, ignoring the message dropping that has been already described. Replaying a control message includes delaying the emission of this message by recording and forwarding it later (potentially in another area) or repeating this message. As a consequence, routing tables are updated based on obsolete information. Note that each message contains a field indicating the period of time during which this message is considered as valid, and hence it is used to update the routing table. Both attacks can be systematic (i.e., targeting any multihops control traffic) or selective. They may also be performed in a distributed manner (Expression 9) with two intruders: one recording the control message from one region so as to replay it in another region (i.e., the one of the colluding intruder). Without loss of generality, let I_1 (resp., I_2) be the node that records it (resp., replays it):

$$\begin{gathered} S \xrightarrow{\mathrm{CM}(S)_t} I_1, I_1 \xrightarrow[\mathrm{enc}]{\mathrm{CM}(S)} I_2, I_2 \xrightarrow{\mathrm{CM}(S)_{t'}} Y, \\ S \notin NS_Y, \bigtriangledown t \leq \left| t' - t \right| < \bigtriangleup t \\ \Downarrow \\ S \in NS_Y \\ \Downarrow \\ I_1, I_2 \in \mathscr{I}. \end{gathered} \tag{9}$$

In practice, I_1 tunnels the control traffic by, for example, encrypting the routing message and/or relying on an alternative network interface (that may be advertised or not). Then, I_2 replays it. This attack leads to the creation of a *wormhole* whose length depends on the distance separating the two intruders. Note that, in order to stay invisible, both I_1 and I_2 may keep the identification field unchanged; the source is S rather than I_1 or I_2. This possibility corresponds to a masquerading. Sequence numbers constitute a standard mechanism that provides protection against replay attacks given that the sequence delivery is not required by OLSR. In counterpart, their usage can be hijacked so that the destination drops the message rather than using it. In practice, an intruder I increases (a wraparound mechanism is implemented; when the sequence number reaches an extreme boundary, it is reset to zero) (resp., decreases) the value of the sequence number sq ($S \xrightarrow{\mathrm{CM}(S)_{t,\mathrm{sq}}} I, I \xrightarrow{\mathrm{CM}(S)_{t',\mathrm{sq}'}}$, $\mathrm{sq}' \neq \mathrm{sq}$) so that the destination assumes that S is providing the freshest (resp., an obsolete) route and, therefore, ignores the subsequent (resp., the actual) control message(s):

$$\begin{gathered} S \xrightarrow{\mathrm{CM}(S)_{t,\mathrm{sq}}} I, I \xrightarrow{\mathrm{CM}(S)_{t',\mathrm{sq}'}}, \mathrm{sq}' \neq \mathrm{sq}, \left| t' - t \right| < \bigtriangleup t \\ \Downarrow \\ I \in \mathscr{I}. \end{gathered} \tag{10}$$

An intruder I may also corrupt the routing table by maliciously modifying a received control message before forwarding it. This modification consists in tampering either the contents of the message or the identification of its source. The former case is similar to the forge attack, described above, wherein the intruder tampers the MPR selector set in TC message, the OLSR routes in HNA message or the interface(s) identification in MID message. While in the latter case, an intruder I forwards the packet containing the control message without changing the source address [34]. Consequently,

two nodes consider themselves as 1-hop neighbors whereas they are not in the communication range of each other. An intruder may also disrespect the flooding mechanism in OLSR. This happens when the intruder retransmits a control message that is received from a non-MPR selector node [32]:

$$I \notin \mathrm{MPR}_S, S \xrightarrow{\mathrm{CM}_t} I, I \xrightarrow{\mathrm{CM}_{t'}}, \left|t' - t\right| < \triangle t \Downarrow I \in \mathcal{I}. \tag{11}$$

Overall, attacks targeting OLSR protocol are classified into 3 types:

(i) *drop* attack that consists in totally or selectively dropping the control messages, for example, dropping TC message in order to foil route calculation;

(ii) *active forge* attack that attempts to poison OLSR's functionalities by introducing novel deceptive control message, for example, a hello message including an incorrect neighbor set so as to foil MPR selection;

(iii) *modify and forward* attack lies in an intruder that captures and modifies a control message before forwarding it (e.g., increasing the sequence number of a TC message; hence the subsequent TC messages are ignored).

An intruder launches the attack either in a standalone manner or in collusion with other intruder(s), constituting what so-called a *Byzantine* attack (e.g., *wormhole*), together usually coupled with masquerading. Detecting these attacks is far to be a trivial task because a minor deviation on an attack makes it undetectable. In addition, an attack can be composed of several subattacks. In order to tackle these issues, we describe/model the attack as general as possible, hence circumventing possible deviations. We then propose an intrusion detection system that detects composed attacks as much as their parties.

3. Intrusion Detection

We propose a distributed, log- and signature-based intrusion detection system named IDAR. In addition to providing a high detection accuracy, IDAR aims to maintain the available resources in MANET. For this purpose, the evidences of attack are extracted from the log and further classified according to their level of gravity. Such classification helps in planning the diagnostic and henceforth minimizing the computation overhead related to attack identification. During the diagnostic, evidences are matched to predefined intrusion signatures. The establishment of an intrusion signature and further the diagnostic operation are exemplified with the link spoofing attack we purposely developed.

3.1. Resource-Aware Evidence Gathering. IDAR distinguishes itself from other IDSs by extracting the signs of attack from the logs instead of sniffing the traffic. Thus, the consumption of energy and computational power resulting from evidence gathering and analyzing is minimized. In fact, sniffing traffic necessitates that the wireless network interface stays in mode promiscuous at all time, thus, the node can overhear all the packets within its transmission range. But promiscuous mode leads to consuming more energy and hence reducing rapidly the lifetime of the node/network. Indeed, a nondestination node in the radio range of either the sender or the receiver overhears some or all of their traffic. For the IEEE 802.11 MAC protocol, a nondestination node in discarding (i.e., nonpromiscuous) mode can enter into a reduced energy consumption mode and discard others' traffic [35]. Such mode requires less energy than the idle mode, which is the default mode in ad hoc network. While a nondestination node operating in promiscuous mode listens to all the traffic, whether or not it is the intended destination. The traffic is further received as if it was a broadcast traffic; thus, additional energy consumption is associated with promiscuous mode operation. Figure 1 represents a part of the experimental measurements realized in [19] about the energy consumption of an IEEE 802.11 2 Mbps wireless card. It shows the significant difference in energy consumption between promiscuous and nonpromiscuous modes. Note that the consumption in the nonpromiscuous mode is negative because it requires less energy than the reference idle mode. Besides increasing the amount of consumed energy, sniffing traffic imposes a huge computational overhead [20]. Indeed, the packet-level analysis, which is applied on the sniffed packets, strains significantly the available resources, that is, memory and CPU processing. Moreover, since all the traffic in the radio range is sniffed, many of the analyzed packets would be redundant and add nothing to the detection. In order to avoid this permanent strain of resources, IDAR does not sniff the traffic but it rather collects periodically the local logs. In particular, it focuses on the portion of logs that characterizes the

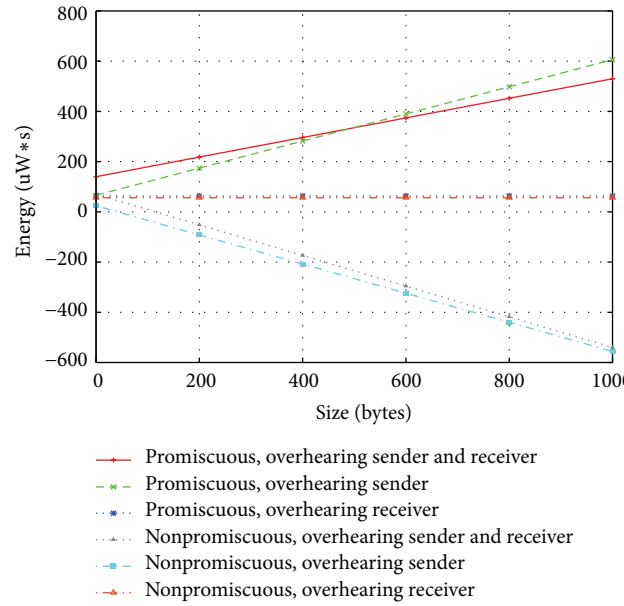

Figure 1: Energy consumption in promiscuous and nonpromiscuous modes.

activities of the routing protocol (e.g., packet reception, MPR selection). Note that additional logs, for example, system-, security-related logs, could be integrated and correlated. Once parsed, a log is used so as to detect a sign of suspicious activity. This consists in matching the log against a predefined intrusion signature. An intrusion signature is thought as a partially ordered sequence of events that characterizes an intrusion. Such procedure is potentially not only memory but also bandwidth consumer. Indeed, it involves examining local logs as well as performing an in-depth diagnostic where other nodes are requested so as to collect additional attack evidences, correlate, and match them against the defined intrusion signatures. The in-depth diagnostic offers a global view about the suspicious nodes, and therefore it increases the accuracy of the detection. But, it is a costly operation in terms of resources. Thus, the in-depth diagnostic must be carefully planned, that is, should be initiated only when a sufficient degree of suspicion exists and terminated as soon as a result is obtained. For this purpose, we propose to classify attack evidences so that depending on their level of gravity, the in-depth diagnostic may be performed. According to our classification, an evidence falls into one of the following four groups.

(i) *Initial-evidence group* contains the evidences that lead to launch an in-depth diagnostic over the network.

(ii) *Suspicious-evidence group* contains the evidences that lead to identify a node as suspicious.

(iii) *Confirmed-evidence group* contains the evidences that confirm the occurrence of an attack. This results in terminating the diagnostic and declaring the suspicious node as intruder.

(iv) *Cancel-evidence group* contains the evidences that eliminate the suspicion and stop the diagnostic.

These groups are populated with the evidences that are extracted from the log. If an evidence belonging to the *initial-evidence group* is discovered, then an in-depth diagnostic is launched so as to confirm (i.e., discovering an evidence belonging to *confirmed-evidence group*) or infirm (i.e., discovering an evidence belonging to the *cancel-evidence group*) the intrusion; both lead to the termination of the diagnostic. Relying on these groups, the evolution of any attack and its related detection is easily followed. In addition, its compact form facilitates the lightweight discovering of long-term intrusions.

3.2. Link Spoofing Attack. A link spoofing attack lies in falsifying hello message(s) so as to modify the local topology perceived by adjacent nodes. This attack influences the MPR selection. In particular, this attack owns a global impact: the MPR position provides to the intruder the possibility to eavesdrop, tamper, misrelay, or drop the traffic. As discussed in Section 2 (and further detailed in Expression 4), an intruder realizes a link spoofing attack through one of the following three cases.

(1) It advertises a non-existing and symmetric node. Thus, the intruder guarantees (unless another attacker advertises the same non-existing node) being selected as a MPR because this non-existing node is uniquely covered by the intruder.

(2) It advertises existing but nonneighboring node(s). The intruder is selected as a MPR if the advertised node(s) is (are) not already covered by another (well-behaving or malicious) MPR.

(3) It keeps under wraps neighboring and symmetric node(s). In this case, the connectivity of the intruder and consequently its chance of being selected as a MPR are both decreased. Used in a standalone manner, this attack aims to decrease the connectivity of one or several nodes; a complete isolation necessitates that no other (well-behaving) MPR covers that node(s).

We develop an attack (Expression 12) wherein an intruder falsifies a hello message that contains both a non-existing node (Case 1) and existing but non-neighboring nodes (Case 2). The reason that motivates this choice is twofold. First, by advertising a nonexisting node N, I ensures being selected as a MPR by the victim S ($\exists N \in NS'_I \ni: N \notin \mathcal{N} \cap NS_I$). Note that with the last versions of the RFC, a node (i.e., potentially an intruder) may dictate its selection as a MPR by advertising (in a hello message) its high willingness to relay messages. Nevertheless, previous versions (and their related implementations) ignore this case. Regardless of the version of the protocol, our intruder (as aforementioned previously) guarantees being selected as MPR by advertising a nonexisting and symmetric node. Second, by announcing common neighbors with another node L (Case 2), I increases the probability that L is not selected as a MPR (I replacing L) and henceforth increases its proper ascendancy. Note that the attack we developed does not consider the third case that involves the reduction of the intruder's connectivity. Nevertheless, attack signature deals with it:

$$
\begin{gathered}
S \xrightarrow{\text{hello}\left(NS_S\right)_t} I, I \xrightarrow{\text{hello}\left(NS'_I\right)_{t'}} S, \left|t' - t\right| < \triangle t, \\
\exists N \in NS'_I \ni: N \notin \mathcal{N} \cap NS_I, \\
\exists L \in \text{MPR}_S \ni: \left[NS_L \setminus NS_I\right] \subseteq \left[NS'_I \setminus NS_I\right] \\
\Downarrow \\
I \in \mathcal{I}, L \notin \text{MPR}_S, \\
\exists I' \in \mathcal{I} \ni: I' \in \text{MPR}_S, \\
\text{Card}\left(NS'_I \setminus \left(NS'_I \cap \mathcal{N}\right)\right) > 0 \\
\text{Card}\left(\left(NS'_I \setminus NS_I\right) \cap \mathcal{N}\right) > 0.
\end{gathered}
\tag{12}
$$

Herein, a key challenge stems from the need to generate a hand-coded intrusion signature that models such attack and henceforth permits to detect it.

3.3. Signature Establishment. As mentioned before, a link spoofing attack aims to inflect the MPR selection; such

Table 2: Evidence characterizing a link spoofing attack.

Initial-evidence group	
$E1$	A MPR is replaced by a 1-hop neighbor or by an already selected MPR
$E2$	A MPR behaves maliciously, for example, it tampers, misrelays, or drops the control messages
Suspicious-evidence group	
$E3$	A MPR is the only node that provides connectivity to one (or more) 2-hops neighbor(s)
Confirmed-evidence group	
$E4$	A MPR advertises a partial set of 1-hop neighbors
$E5$	A MPR advertises nonneighboring node(s) as 1-hop neighbors(s)
Cancel-evidence group	
$E6$	A MPR is defined as an intruder/well-behaving node

selection is triggered upon a change in the symmetric 1- and 2-hops neighborhood. Rather than launching an in-depth diagnostic upon every change in the 1- or 2-hops symmetric neighborhood, we keep to a minimum the number of these diagnostics, and henceforth the amount of consumed computational power and bandwidth, by initiating it only at the occurrence of an event related to a link spoofing attack. More precisely, we ignore the changes in the 1-hop neighborhood (e.g., apparition of 1-hop neighbor) because they are observed by the node itself. Thus, they are not subject to the remote falsification which is the cornerstone of a link spoofing attack. In contrary, changes in the 2-hops neighborhood are considered as long as they impact the MPR selection. In practice, the evidences that reveal a link spoofing attack (Table 2) are broken down into:

(i) a MPR replacement (Evidence 1 or $E1$ for short) that results from a change in the covering of the 1-hop neighbors; one (or several) 1-hop neighbor(s) (possibly the replacing MPR) increase(s) its (their) coverage to the detriment of the replaced MPR. A replacement MPR is a 1-hop neighbor that is excluded from the MPRs set even though it provides the same connectivity in both the previous and current MPR selection rounds.

(ii) No MPR replacement takes place but an already selected MPR is detected as misbehaving node. For instance, a misbehaving MPR may drop, falsify, or misrelay the control messages ($E2$). Note that an evidence of a special interest herein refers to a node promoting itself as a MPR without having been selected as a MPR. A misbehaving MPR includes also the case wherein a MPR continually advertises the same 2-hops neighbors set despite being no more valid. Contrary to other cases, this one is not event driven and should be handled based on random or periodical checks;

(iii) a MPR is the only one that covers one or several nodes ($E3$);

(iv) a MPR covers partially its adjacent neighbor(s) ($E4$);

(v) a MPR provides connectivity to a nonneighboring node ($E5$).

The occurrence of either $E1$ or $E2$ constitutes the starting point of an in-depth diagnostic. $E1$ and $E2$ belong to the *initial-evidence group*. The act of being the only MPR that provides the connectivity to node(s) ($E3$) is suspicious but is not sufficient to launch an in-depth diagnostic because this situation is typical in a sparse network. Furthermore, two nodes within the covering range of each other often fail in communicating due to the unpredictable nature of wireless transmission resulting from, for example, obstacles and noises. Thus, diagnosing $E3$ is especially difficult under no specific assumption. Overall, the occurrence of $E1$ or $E2$ and optionally $E3$ leads to an in-depth diagnostic (Expression 13). In practice, the 1-hop neighbor(s) of the suspicious MPR are interrogated. Note that part of the interrogated nodes may express a different opinion that results from malicious node or an obsolete routing information. This calls for taking into account their respective reputation (as we plan in our near future work):

$$\begin{array}{c} (E1 \vee E2) \\ \Downarrow \qquad\qquad\qquad \Downarrow \\ (E4 \vee E5), \text{optional}\,(E3) \qquad (!E4 \wedge !E5) \\ \Downarrow \qquad\qquad\qquad \Downarrow \\ \text{The suspicious MPR is an intruder.} \qquad \text{The suspicious MPR is well-behaving.} \end{array} \tag{13}$$

If the obtained answers confirm (resp., infirm) that the suspicious MPR covers partially its neighbors ($E4$) and/or advertises a distant node as 1-hop neighbor ($E5$), then the MPR is declared as an intruder (resp., well-behaving) and the diagnostic is terminated ($E6$). Obviously, the cooperation between the nodes constitutes a cornerstone in our in-depth diagnostic.

3.4. Cooperative Diagnostic. The in-depth diagnostic of a link spoofing attack consists in verifying the existence of a symmetric and neighboring relationship between a suspicious MPR and its advertised 1-hop neighbors (Algorithm 1). More precisely, this diagnostic follows the following steps. First, the MPRs that have been replaced by other MPR or 1-hop neighbor are identified (lines 1-2) because such replacement represents the initial evidence of a link spoofing attack ($E1$ in Table 2). For this purpose, the function *GetReplaced-Mpr* is called (Line 1). It establishes the MPRs that have been

```
(1)  OldMprs = GetReplaced-Mpr();
(2)  SuspiciousMprs = GetReplacing-Mpr();
(3)  for (suspicious ∈ SuspiciousMprs) do
(4)     InterrogatedNodes = GetCommon2HopsNeighors
        (suspicious, OldMprs);
(5)     for (2HopsNeighbor ∈ InterrogatedNodes) do
(6)        if (LinkExistence (2HopsNeighbor, suspicious) ==
           false) then
(7)           Generate-Alarm (suspicious);
(8)        end if
(9)     end for
(10)    Cancel-Suspicious (suspicious);
(11) end for
```

Algorithm 1: In-depth diagnostic.

replaced by comparing the current MPRs and the penultimate MPRs. Then, the MPRs sharing 1-hop neighbor(s) with a replaced MPR are identified (function *GetReplacing-MPR*, line 2). Those identified MPRs are tagged as suspicious and are subject to further analysis (lines 3–11); the objective is to verify whether some spoofed links have been advertised so as to replace the MPR. It follows that the 1-hop neighbors that are common to a replaced MPR and a suspicious one are determined (*GetCommon2HopsNeighors* function, line 4). Note that these 1-hop neighbors also correspond to the 2-hops neighbors of the node launching the diagnostic. They are interrogated (*LinkExistence* function) so as to verify the existence of links. The interrogation of a 2-hops neighbor, denoted by *Ai*, consists in sending a request to *Ai* asking if this latter considers the suspicious MPR as a 1-hop neighbor at a specific time t. Whenever possible, this request is sent without going through both the suspicious MPR I or any colluding intruder I'. This eluding is necessary in order to prevent I and I' from dropping the request and/or forging a deceptive answer. Therefore, the 1-hop neighbor(s) (primarily the MPR(s)) that covers the interrogated 2-hops neighbors is provided with the request. Note that the suspecting node S is aware of is provided with the connectivity between the nodes that form its two-hops neighbors. Thus, S may select the one (if existing) that covers *Ai*. If no answer is obtained (i.e., the related time-out elapses), then the request is sequentially transferred through the rest of the covering 1-hop neighbors (keeping in mind that, as aforementioned, MPRs are primarily selected). However, when no neighbor is left, then an (multihops) alternative path is researched in the routing table to reach *Ai*. Note that the verification related to a suspicious MPR is performed within an independent thread; hence, the diagnostic of one node (and the result waiting) is not a blocking to others. If *Ai* denies being a 1-hop neighbor of the suspicious MPR, then a countermeasure is triggered. More precisely, an alarm that establishes the detected attacker(s) is broadcasted (*Generate-Alarm* function, line 7). Otherwise, if no deny takes place, the suspicious MPR is no longer suspected (*Cancel-Suspicious* function, line 10). In both cases, the diagnostic terminates. Note that if no answer is provided (the inconsistent answers lead also to a claim in the degree of suspicion. However, this case is not included in this paper but it is handled in our other works), then the suspicious MPR is tagged as not verified and the degree of suspicion in this latter goes up. It is worth mentioning that the current implementation of IDAR considers the link spoofing attack. However, considering other attacks is not complicated and requires only specifying the evidences to be searched in the logs and the information to be exchanged between the nodes during the in-depth diagnostic.

4. Architecture and Performance Evaluation

In this section, we introduce the key architectural components of IDAR (Section 4.1). Then, we present the evaluation of IDAR performance (Section 4.2) and further discuss the obtained result (Section 4.3).

4.1. IDAR Architecture. In order to cope with the dynamic nature of MANET, IDAR is both distributed and cooperative. The proposed architecture (Figure 2) requires that every device participates in the detection (The delegation of some detection operations (e.g., logs analyzing) may be shared among devices. But, it will not be necessarily less expensive in terms of resources consumption.) More precisely, each device contains an instance of IDAR, which independently detects the signs of suspicion and cooperates with other instances so as to conduct the diagnostic in a broader range. IDAR is implemented in *Perl* (http://www.perl.org/) and conceptually structured into 4 components.

(i) *Coordinator* that orchestrates all the components. As such, it constitutes the hearth of IDAR. In practice, it parses dynamically the OLSR logs so as to extract signs of suspicion and match these latter against predefined intrusion signatures. The *intrusion signatures* are represented as conditional rules (*if condition then state*). Furthermore, it triggers the communication and the alarm notification in order to launch advanced diagnostic.

(ii) *Communication manager* that gathers information about, for example, the adjacent links as required by the diagnostic manager. Meanwhile, it answers the diagnostic requests. This component runs into a separated thread so that other IDAR operations are not blocked.

(iii) *Knowledge database* includes the information that is extirpated from the logs and is provided by the communication manager. This database is realized in *Mysql* (http://www.mysql.com/). Stored information encompasses, for example, the 1-hop and 2-hops neighbors and the MPRs.

(iv) *Alarm notifier* is responsible for alarming the network when an intrusion is detected. Note that we are planning to include more countermeasures in the future, for example, eliminating the routes containing an intruder.

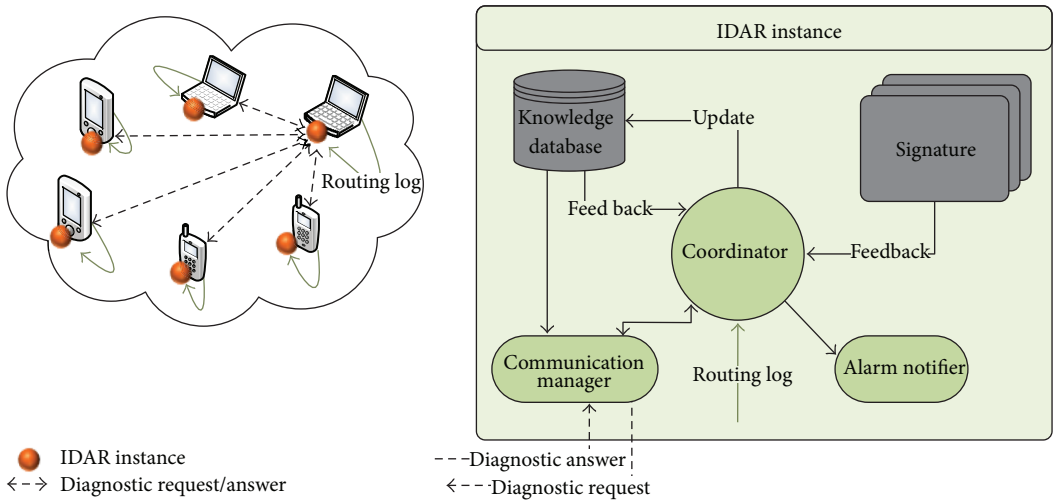

Figure 2: IDAR architecture.

All the above components have been developed so as to support intrusion detection, offering a high rate of detection along with conserving the resources.

4.2. Performance Evaluation. In order to evaluate the performance of IDAR, a mobile ad hoc network has been simulated using the network simulator NS3 (http://www.nsnam.org/) [36]. Each node in the simulated network is further coupled with a Linux Container (LXC) virtual machine (http://lxc.sourceforge.net/) [37]. The reason that motivates this coupling is twofold. First, NS3 offers the possibility to simulate a large-scale mobile ad hoc network. Second, LXC, an operating-system-level virtualization tool, permits to run multiple isolated machines (also called *containers*) on a single modified hosting kernel (up to 1024 containers over a single hosting kernel). Each container owns its proper resources (e.g., process tree, network interface, and IP address). Thus, the resource consumption (e.g., memory usage) can be isolated and measured. In practice, an instance of IDAR is installed on each container that appears as a standalone/separated machine. Figure 3 exemplifies the coupling between nodes in NS3 and LXC containers. From the IDAR perspective (as well as from any application installed on the container), the emission and reception of packets is done through the network interface (*eth0*). While the container contains a simulated ethernet card (*veth*), which is connected through an *ethernet bridge* towards a kernel tap device (*tap*). Note that for each container-node coupling, new *tap* and *ethernet bridge* should be defined in the hosting kernel. The *ethernet bridge* implements the forwarding of link-level frames. The interface *tapRouter* [38] is implemented in NS3 and is used to exchange packets between the *tap* device and the *WiFi* device, which enables NS3 nodes to communicate over a IEEE 802.11- and OLSR-enabled MANET. Overall, the outgoing packets from *eth0* in the container are delivered to the *WiFi* device at the corresponding node in the simulated network and vice versa. Simulation is carried using the aforementioned platform, wherein the hosting device is equipped with 32 GB of memory and 2 Intel(R) Xeon(R) (6) Core @2.40 GHz CPUs. Containers hold a Fedora 12 operating system. We consider a MANET (Table 3) constituted of $N = 30$ nodes split into 25 well-behaving nodes and 5 intruders. While most of other IDSs are evaluated against one intruder, we show that IDAR can handle several intruders in the same time. Intruders launch repeatedly the implemented link spoofing attack (as described in Expression 12). In our experiments, the number of successful intrusions varies between 15 and 33 according to the parameters of simulation scenario (as described in Expression 12). Nodes communicate via IEEE 802.11a and are characterized by a transmission range T_x of 90 m. They randomly move, following the so-called *RandomWalk2d* [39] mobility pattern provided by NS3: each node moves according to a randomly chosen direction at a given speed that is the same for all the nodes. When a node hits the network boundaries (unless specified, the network area is defined by a squared area of $S = 310 \times 310\,\text{m}^2$), it rebounds following a reflexive angle. Data traffic is further modeled using the *V4PingHelper* application in NS3; each node exchanges 56-byte ICMP (http://www.ietf.org/rfc/rfc792.txt) echo requests to one another and waits for 1 s before sending it again. We use OLSR as the underlying routing protocol, preserving the configuration parameters promoted in the RFC 3626 (e.g., hello message interval is 2 s).

For each experimental scenario, the simulation is launched 5 times, each one lasts for 140 s. We compare the launched attacks with the detected ones and present the average of results. We aim at evaluating the performance of IDAR in terms of:

(i) intrusion detection rate that reflects the capacity of detecting successful intrusions (a successful intrusion causes the replacement of a legitimate MPR of the victim by an intruder);

(ii) false positive rate that measures how many times a legitimate node is wrongly designated as an intruder;

Table 3: Simulation parameters.

Simulation platform	
Simulator	NS3 + LXC virtual machines
Container operating system	Fedora 12
Hosting device memory	32 GB
Hosting device CPU	2 Intel(R) Xeon(R) (6) Core @2.40 GHz
Simulated MANET	
Number of mobile nodes	30
Number of intruders	5
Topology	$310 \times 310\ m^2$
Transmission range	90 m
Maximum bandwidth	1 Mbps
Traffic	*V4PingHelper*: 56 bytes Icmp packets
Mobility model	*RandomWalk2d*
Maximum speed	8 m/s (or 28.8 km/h)
Simulation time	140 s

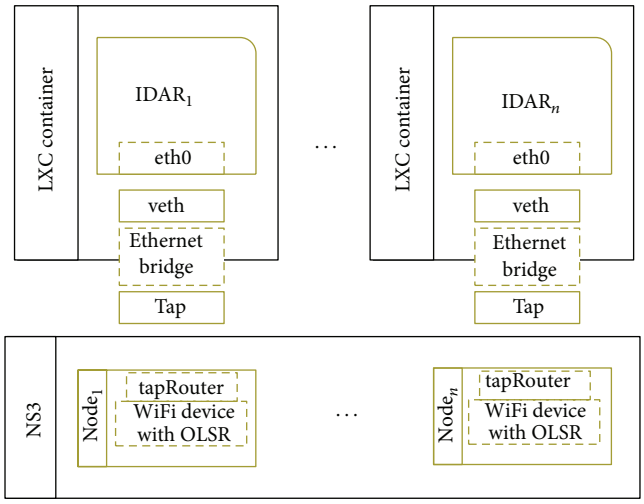

Figure 3: Experiment platform.

(iii) detection overhead which represents the additional network traffic that is generated because of IDAR. Note that further benchmarks are also provided in terms of memory usage.

Based on those performance indicators, IDAR is further evaluated with regard to the network density (Section 4.2.1) and node mobility (Section 4.2.2).

4.2.1. Network Density. In order to evaluate the scaling properties of IDAR, we vary the density of the network. The density of the network corresponds to the average number of neighbors, which is defined as in [40] by $(N \times \pi \times T_x^2)/S$. We selected an interval of density that ranges from 6 up to 16 neighbors, which includes the intruder, the victim, and the remaining neighbors. Figure 4(a) shows that the detection rate keeps higher than 80% regardless of the network density. This demonstrates the relatively limited impact of the network density on IDAR performance. More precisely, the intrusion detection rate slightly rises from 93.5% up to its maximum 96.3% when the density varies from 6 up to 8 neighbors: the greater is the number of neighbors, the higher is the visibility of the attack. Then, a slow diminish is observed; dense networks (with a density exceeding 8) are characterised by a high collision rate that causes a decline in the detection rate. Similarly, Figure 4(b) highlights that the average percentage of false positives is neglectful (under 5.9%). The analysis of several cases of false positive leads us to find that a false positive occurs when two nodes own a different live-times for the bidirectional link that connects these later; consequently, one of the two nodes still confirms the existence of this link while the other one denies it. To override this problem, the period during which those links are considered valid should be amplified as in [41]. Figure 4(c) presents the average memory usage caused by IDAR. This usage gently fluctuates between 17.6 MB and 22.2 MB. It is also worthy of mention that contrary to OLSR, our system does not assign additional functionality to MPR (detection may be launched by one another); IDAR provides an homogeneous load balancing among the nodes. As illustrated in Figure 4(d), the traffic caused by IDAR is very low (under 0.5% for a network density inferior to 8). It then slowly rises to reach 1.3% when the network density is 16. In order to understand the slight fall when the density reaches 14, we compare the average number of suspicious cases with regard to a varying density. This average was 70.8 (resp., 65.8) with density equals to 12 (resp., 14). Consequently, the number of diagnostic-related packets is smaller at a density of 14 than at a density of 12.

4.2.2. Mobility. In order to isolate the influence of the mobility, the network density is set to 8. The detection rate falls when the speed increases (Figure 5(a)). This results from the ever changing topology that results in increasing the number of broken links and the related drops of diagnostic-related packets. However, IDAR still provides a high detection rate; for example, the average of detection rate is 70.7%

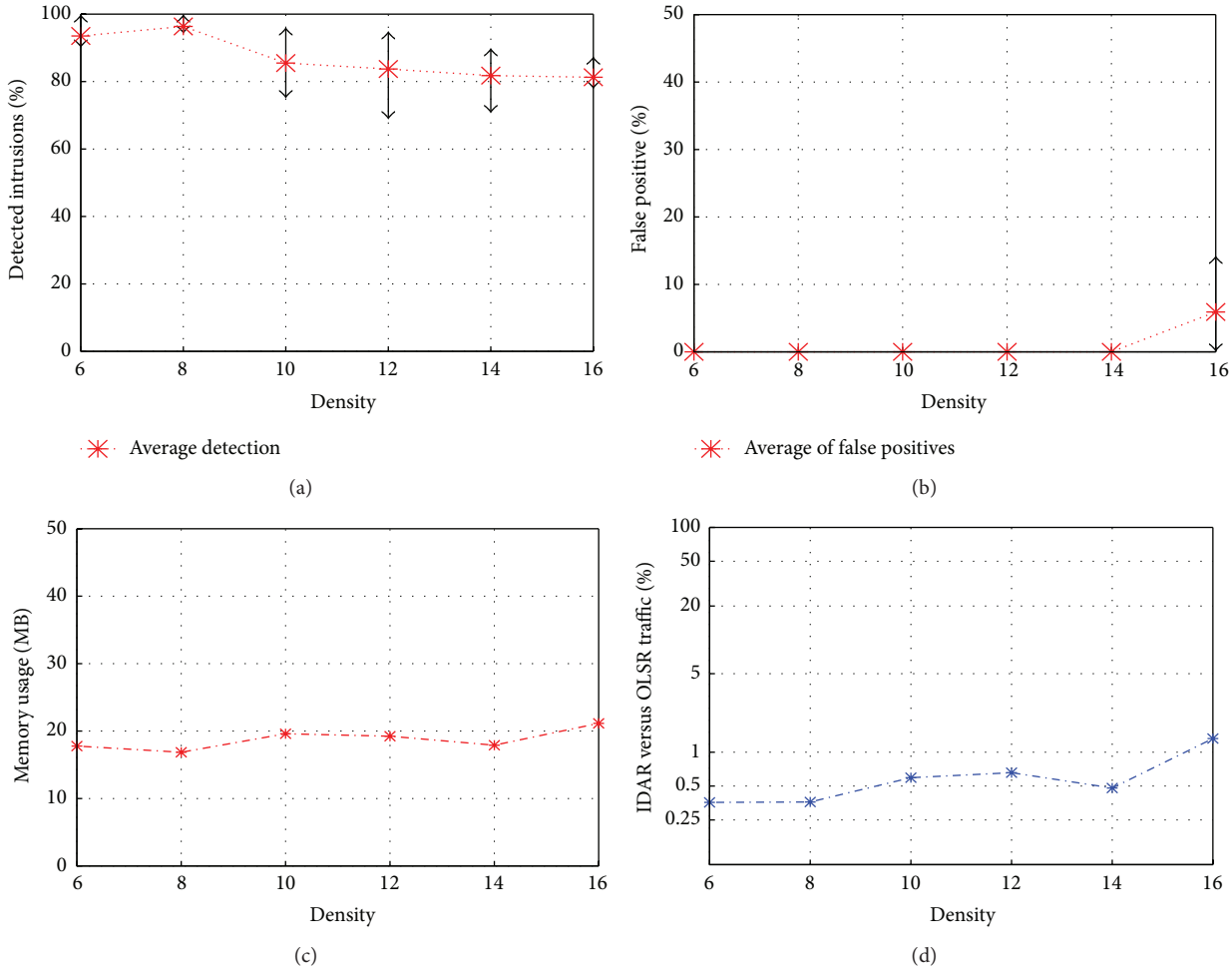

Figure 4: Intrusion detection rate (a), false positives rate (b), memory usage (c), and traffic (d) depending on the network density.

with a moving speed equals to 8 m/s (i.e., around 28.7 kilometers/hour equivalent to 17.89 miles/hour). Figure 5(b) shows that the higher is the moving speed, the greater it gets false positives, mainly because of out of date feedback. In fact, when a node moves away, some of its 1-hop neighbors are no longer neighbors but are still announced as such and are considered as potential attackers. A higher increase of the false positive rate is observed when the node speed ranges from 4 m/s up to 6 m/s because the instable node covering is given its transmission range, which leads to an increased update of its 1-hop neighbors. The memory usage (Figure 5(c)) rises staidly from 17.6 MB to 26.5 MB, which is reasonable even for resource-limited devices. The impact of the mobility on the bandwidth is shown in Figure 5(d). The IDAR traffic comparing to the OLSR traffic significantly grows from 0,36% up to 3.1% for a speed ranging from 0 m/s up to 2 m/s. Then, the bandwidth usage gradually grows and reaches almost 6% when the moving speed is equal to 8 m/s. This is motivated by the fact that increasing the number of verified links leads to exchange more diagnostic-related messages in the network. In order to understand the reason behind such variable growth of IDAR traffic, we counted the number of the verified suspicious neighbor links. We found that the average number of the verified links significantly grows from 45.2 to 201.4 when the velocity rises from 0 m/s to 2 m/s. This average then gradually grows and takes the values 240.8, 331.4, and 406 when the velocity equals to 4 m/s, 6 m/s, and 8 m/s, respectively. The limited number of the suspicious neighbor links when the nodes are static is justified by the fact that the attacker has a limited choice to falsify its neighbors links. Instead, when the velocity is greater than 2 m/s, there is a continuous change in the neighborhood and the attackers can attack more (diversified) targets. Meanwhile, the continuously changing topology implies an increased number of investigations.

4.3. Performance Discussion. It is clear that the density of the network holds less influence than the mobility on our IDS. More mobility in MANET causes more dropped diagnostic packets, and hence less diagnostic is concluded with a final result, that is, confirming or refuting the suspiciousness. However, comparing the launched attacks to the discovered suspicious events shows that the majority of attacks are tagged as suspicious events and considered for further diagnostic. Note that the mobility has a negative influence

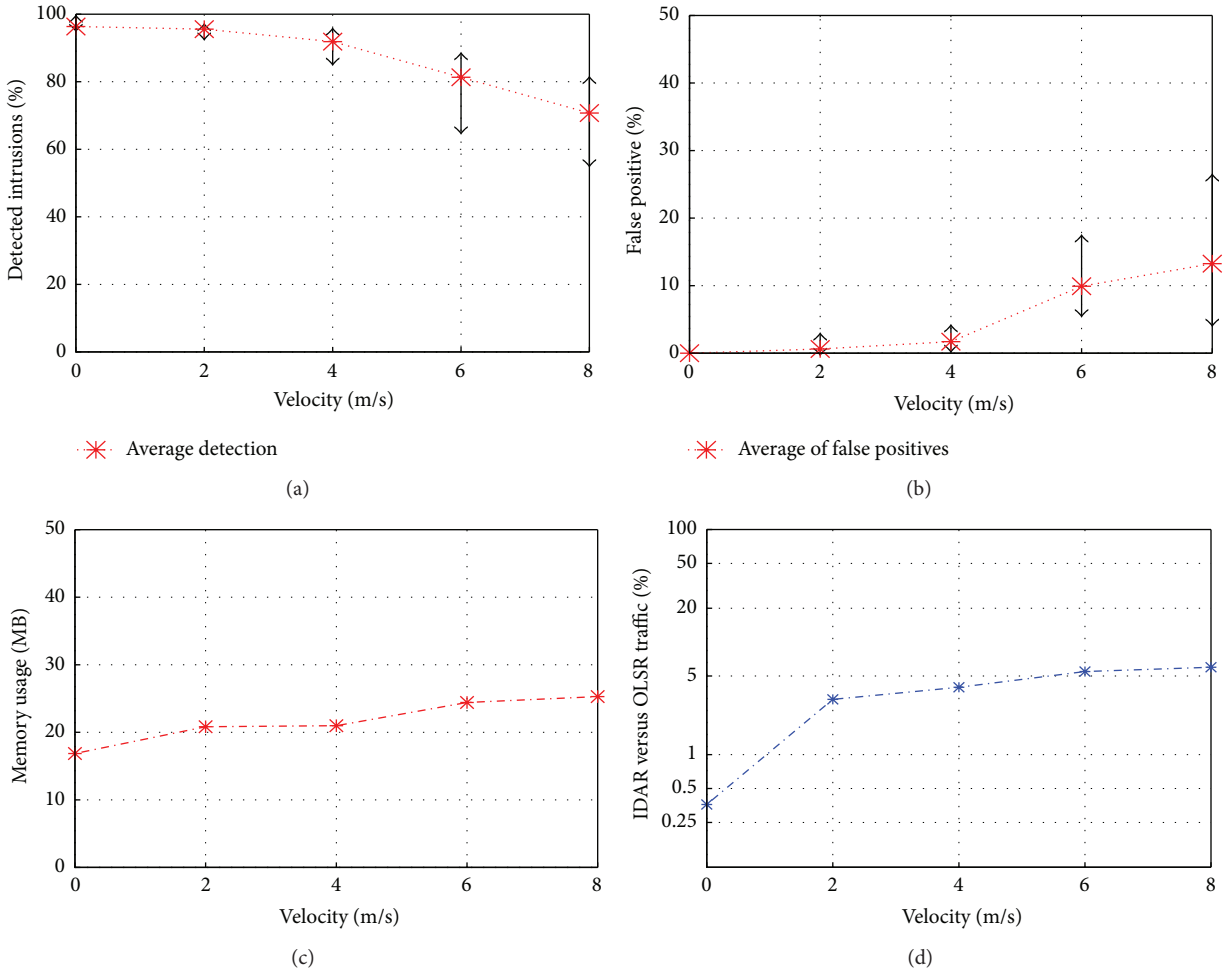

Figure 5: Intrusion detection rate (a), false positives rate (b), memory usage (c), and traffic (d) depending on the network mobility.

on all the IDSs especially the cooperative ones where there is an exchange of evidences/alerts. Table 4 provides a short performance comparing between IDAR and other known IDSs that are dedicated to ad hoc routing protocol. Like the majority of these IDSs, the performance of our system is challenged against a specific attack: link spoofing attack, in a simulated MANET (According to [42], around 75% of mobile ad hoc published papers use simulation for test issues.) Two evaluation factors are common to almost all the listed IDSs: the detection rate/capacity and the positive false rate. Even though it was challenged in a MANET wherein the intruders constitute 16.6% (Note that this percent is the 3rd highest registered percent) of the nodes, IDAR provides one of the highest detection rate. Regarding the positive false rate, we can notice that several IDSs, especially those that are dedicated to DSR protocol, show better results than our system. However, this latter still generates a limited number of false positives. It is also possible to reduce the false positives rate by amplifying the TTL (Time To Live) parameter of a bidirectional link so as to reduce the impact of obsolete neighborhood-related information during the detection. Recall that most of the generated false positives in IDAR are generated because of the difference in the time-live of a bidirectional link between the two ends of this link. In such case, this link will be expired in one end, while the other end will keep confirming its existence. It is worth mentioning that, except our system, the listed IDSs that aim to detect the attacks targeting OLSR protocol do not present neither the detection rate nor the false positives rate. But they merely prove the capacity of detecting the addressed attack(s). In addition, IDAR is one of the rare IDSs that are evaluated in terms of resources consumption. Indeed, we measure both the increment in memory usage and traffic overhead resulting from detection operations. IDAR triggers a small increment in the devices' memory usage (between 17.6 MB and 26.5 MB). Therefore, it can be tolerated by the resource-limited devices. It further triggers the lowest traffic overhead compared to the listed IDSs. IDAR presents a good scalability and offers a high detection rate even in the high density networks where a large amount of routing traffic should be monitored (Figure 4(a)). This scalability is reinforced by (i) the limited traffic overhead, (ii) the use of routing logs so as to extract the evidences instead of sniffing the traffic, (iii) the distributed nature of detection such that each node is responsible for detecting the attacks targeting it, and (iv) the unnecessity of modifying the used routing protocol.

5. Related Work

Systems that detect intrusion targeting ad hoc routing protocol are extremely diverse in the way they analyze intrusion. They fall into the three key categories.

(i) *Anomaly detection* system defines the correct behavior of the node/network so as to detect deviations to this behavior. This correct behavior is automatically built during an attackless training phase. The detection accuracy depends on the ability to (i) describe the correct behavior and (ii) distinguish between anomalous and unexpected behaviors.

(ii) *Specification-based detection* system hand-codes the legitimate function/operation and then searches for a violation to this operation.

(iii) *Signature-based detection* system first describes the way an intruder penetrates the system by defining an intrusion signature. Then, any behavior that is close to this predefined signature is flagged as intrusion.

Hereafter, we detail examples of each of these categories. Anomaly-based detection constitutes the main approach used to detect attacks. The IDS proposed by Zhang et al. [7] constitutes a de facto standard in MANET. It aims at detecting the attempts to falsify the routes provided by the AODV [61], DSR [62], and DSDV [63] routing protocols. During the training phase, the impact of movement on the percentage of changes in the routing table is analyzed. Note that this movement (i.e., velocity, direction, and position) is provided by a Global Positioning System (GPS). Then, during the operation phase, an actual percentage of changes differing from the predicted one, is defined as anomaly. In practice, the distinguishing between anomalous and normal behaviors is provided by the Support Vector Machine (SVM) Light [64] classifier or by RIPPER [15], a rule-based engine. The hierarchical IDS proposed in [47] detects *blackhole* and routing request flooding attacks targeting AODV protocol. Here, MANET is arranged into clusters, whereas the node with the highest residual energy and number of connections is elected as a cluster-head. Each cluster-head monitors the traffic inside its cluster so as to identify the abnormal values of (i) the percentage of changes in the routing table and (ii) the propagation of routing/data packets. Such abnormalities are identified thanks to (1-SVM) classifier [65], a deviation of SVM that needs to be trained with either normal or abnormal scenario but not both of them. In [48] (resp., [66]), a blackhole (and resp., dropping) attack targeting AODV protocol (resp., a secured version of AODV protocol including authentication and reputation estimation) are detected by investigating features, for example, the number of route requests and route replies as well as the average difference of sequence numbers (Largely increased sequence numbers are known as a sign of blackhole attack). Then, simple anomaly detection is applied: if the distance between the actually observed features and the average ones that are recorded during the training exceeds a given threshold, then an intrusion is detected. More sophisticated cross-features analysis (CFA) [67] is applied to detect both blackhole and packet dropping on AODV and DSR protocols. Features including, for example, the reachability between 2 nodes and the number of delivered packets, are analyzed within time windows. This analysis attempts to quantify the relation existing between one feature f_i and the others $f_1, \ldots, f_{i-1}, f_{i+1}, \ldots, f_k$ (with $k+1$ defining the number of features analyzed). During the operation phase, the probability of matching between the observed feature f_i' and the predicted feature f_i, which is established based on $f_1', \ldots, f_{i-1}', f_{i+1}', \ldots, f_k'$, is calculated by a decision tree classifier named C4.5 [68]. An anomaly refers to an average of matching probability fewer than a given threshold. CFA and C4.5 are also used in [69] with traffic-related features (e.g., number of received/transferred packets) and OLSR- or AODV-related features (e.g., MPR updates for the former and route discovery for the latter). A modified Markov chain-based IDS is proposed in [58] so as to detect route disruption attack targeting DSR protocol. Here, the detection depends on identifying unpredicted value of two features: the percentage of changes in the number of routes and the percentage of changes in the number of hops. In practice, a Markov chain is based on the immediate previous n consecutive values of a feature as to predict the $(n+1)$th value. If the difference between a predicted and an observed value exceeds a given threshold, an intrusion is concluded.

Rather than establishing automatically a correct behavior, specification-based detection system hand-codes the correct behavior of the routing protocols based on the protocol specification (IETF draft or RFC). Then, the system attempts to detect a violation of constraints circumventing this behavior. Four key constraints define the correct behavior of OLSR ([14, 70]).

(1) Neighbor relation must be reciprocal (i.e., 2 neighbors must hear the hello message sent by each other).

(2) The MPRs and the nodes that select the MPR (i.e., the MPR selectors) must be adjacent.

(3) A node that finds itself advertised as a MPR selector in a TC message must be adjacent to the originator of this message.

(4) Nodes receive TC messages without modifications from MPR(s).

Each node sniffs the traffic in its radio range so as to discover the violations of any of the above constraints. These constraints are modeled by semantic properties in [70] (resp., rules in [14]) so as to detect link spoofing attack on TC (and resp., hello messages); both experiment an ability to detect a link spoofing by observing a violation of the third condition over a simulated network composed of 11 nodes. Finite-state machines are used to describe similar constraints on OLSR [41] and the behavior of AODV protocol [52, 71]. In [41], a centralized detection of link spoofing, man-in-the-middle, and deny of service attack, is performed. While in [52, 71], incorrect hop counts/sequence numbers are analyzed by distributed sensors, which sniff and group the packets per request-reply flow so as to estimate the forwarding path per flow. If a route request/reply is illegally modified or forwarded via a nonexpected path, then an alarm is triggered. In [60], similar FSMs are used to define constraints on, both, the route discovery in AODV and the packet forwarding operations.

Table 4: Comparing IDAR performance with other IDSs.

IDS	Routing protocol	Number of nodes	Intruders (%)	Moving speed	Detection rate	False positives rate	Increment in memory usage	Increment in traffic overhead
IDAR	OlSR	30	16.6%	0–8 m/s	70.7%–96.3%	0%–13.2%	17.6–26.5 MB	47.8–621.5 KB (0.36%–5.97% of OlSR traffic)
[43]	OlSR	30	3.3%, 6.6%, 10%	—	—	—	—	6%–12% of OlSR traffic
[41]	OlSR	10	10%	0 m/s	Detection is possible	—	—	—
[44]	OlSR	7	14.3%	0 m/s	Detection is possible	—	—	—
[45]	OlSR	5	20%	0 m/s	Detection is possible	—	—	—
[14]	OlSR	11	9%	0 m/s	Detection is possible	—	—	—
[46]	AODV	30	3.3%	0–5 m/s	30.67%–90%	0%–20%	—	—
[47]	AODV	50	2%	0–20 m/s	90%	5%	—	—
[48]	AODV	30	3.3%	1–20 m/s	70%–82%	12%–18%	—	—
[49]	AODV	20, 50	10%	1–10 m/s	96.4%–98.7%	0.79%–0.93%	—	900 kB
[50]	AODV	52	3.8%	0–20 m/s (Fixed IDSs)	100%	0%–0.6%	—	—
[51]	AODV	5, 21	20%, 23.8%	0–10 m/s	50%–100%	0%–30%	—	—
[52]	AODV	30	—	0–20 m/s	70%–100%	0.5%–15%	—	—
[53]	AODV	50	Random	0–20 m/s	$79 \pm 10\%$–$92 \pm 3\%$	$5 \pm 1\%$–$32 \pm 8\%$	—	—
[54]	AODV	10, 20, 50	—	0–10 m/s	0%–100%	0%– $\gg$ 100%	0.2%–0.6%	—
[55]	AODV	20–200	—	0–20 m/s	30%–95%	0%–16%	—	—
[56]	AODV	50	—	1–20 m/s	99.8%–100%	0.83%–8.46%	—	—
[7]	AODV	—	—	—	$88.48 \pm 4.14\%$–$97.1 \pm 0.32\%$	$1.45 \pm 0.72\%$–$20.2 \pm 6.27\%$	—	—
	DSDV	—	—	—	$85.23 \pm 3.28\%$–$90.61 \pm 2.99\%$	$5.37 \pm 3.10\%$–$26.3 \pm 5.49\%$	—	—
	DSR	—	—	—	$85.2 \pm 2.38\%$–$99.1 \pm 0.37\%$	$0.03 \pm 0.04\%$–$15.3 \pm 4.08\%$	—	—
[57]	DSR	50	40%	0–20 m/s	—	—	—	11%–31.3% of data traffic
[58]	DSR	30	3.3%	3–5 m/s	60%–100%	2.5%–50%	—	—
[59]	DSR	30	—	—	75%	1.2%	—	—
[60]	DSR	50	2%	0–20 m/s	83.7%–97.4%	1.3%–7.2%	—	—

A signature-based system distinguishes itself by modeling the misbehavior (rather than the correct behavior) so as to identify if a sequence of observed events matching an intrusion signature. AODVSTAT [54] uses state-based signatures in order to detect dropping and spoofing attacks along with network flooding. Few sensors sniff the traffic and match it against predefined signatures. They also exchange periodically MAC and IP addresses so as to detect an identity spoofing which is characterized by a node emitting a packet identified by MAC and IP addresses differing from those registered for this node. A dropping attack refers to a node that fails to replay route request/reply. Finally, a node sending a number of packets exceeding a given threshold is identified as willing to exhaust resources. In [45], intrusion signatures are specified in opposition to the legitimate behavior of OLSR depicted in conjunction with specification rules similar to those defined in [70]. In particular, a node N mistrusts two neighbors L and M that conform to the following rules: M advertises L as a neighbor while L does not advertise M as a neighbor (or reciprocally), both L and M send TC messages while L's neighbors are a subset of M's neighbors (or reciprocally). In addition, N mistrusts M if N which is not a neighbor of M finds itself as a MPR selector in a TC of N (constraint 3). In [44], a FSM is employed to model the signature of hello message fabrication in OLSR protocol in an agent-based IDS. In practice, each node uses a Simple Network Management Protocol (SNMP) agent so as to collect audit data from the Management Information Base (MIB). After that, events are extracted from the collected audit data and are matched to the attack signature. The addressed attack aims at breaking the link between a victim node and its neighbors, and thus, a DoS takes place. To that end, the attacker impersonates the identity of the victim and sends a fake hello message advertising one of the victim symmetric neighbors with lost link status. Upon receiving the fabricated message, the neighbor changes the status of its link with the victim to "heard" and stops routing packets through the victim.

5.1. Synthesis and Discussion. A great majority of the literature is focused on anomaly detection while scarce effort investigates specification- and signature-based intrusion. This naturally calls for consolidating the effort on specification- and signature-based detection while following the *habitus* of wired network which consists in coupling these detection systems with one another, for example, in [72] wherein a combination of anomaly- and signature-based detection is realized so as to increase the detection rate. Almost all of the IDS focuses only on the detection accuracy, that is, providing a high detection rate along with limited false alarms. However, they do not consider the criticality of maintaining the resources and henceforth extending the lifetime of the node/network. Our IDS takes into account the necessity of reducing to minimum, both the communication and computation overload related to intrusion identification. The absolute majority of the IDSs is simulated (including the system we are proposing) rather than being either developed or experimented on testbeds or relying on empirical data. The reason that explains this situation is twofold. First, the deployment of critical MANET is limited/not advertised. Second, to the best of our knowledge, experiences of intrusion in MANETs are not reported: intrusions are in fact developed/simulated as proofs of concept. Nevertheless, not only routing protocols but also the characterization of the intrusion together leverage on the experience gained in designing routing protocols for wired networks and dealing with related misbehaviors. In particular, the similarity existing between the OSPF and OLSR protocols implies that many threats (e.g., sequence numbers, identity spoofing) are shared. In other words, intrusion may be inspired from the one targeting wired networks. Meanwhile, the development of intrusion detection and the envisioning of attacks are accelerated.

6. Conclusion

We survey and classify the attacks that target ad hoc routing protocols focusing on the OLSR protocol. In order to facilitate the definition of intrusion signatures, we extend a description model—an attack is expressed as the preconditions and the resulting consequences—and enrich it with temporal annotations. Once hand-coded, these signatures are utilized by IDAR, a log-based, distributed intrusion detection system dedicated to operate in mobile ad hoc networks. IDAR distinguishes itself by analyzing the logs generated by a routing protocol and extracts intrusion evidences so as to compare these latter against predefined intrusion signatures. For this purpose, evidences are categorized into four groups according to their degree of suspicion/gravity and hence to their ability to activate/deactivate the diagnostic. We further develop a link spoofing attack on the OLSR protocol, build the related detection rules, and evaluate the performances of IDAR relying on the NS3 simulator coupled with LXC virtual machines. Overall, the experiments figure out a high rate of intrusion detection and low false positives rate even under increased mobility and density. Meanwhile, resource consumption and network overhead result from the diagnostic are low and adapted to the resource-constrained devices. Note that ongoing effort is provided as to evaluate IDAR in a real MANET consisting of resource-constrained devices. Still, detecting intrusion is not a trivial task due to the large number of evidences to tackle and the resource-consuming diagnostic. A compromise between detection accuracy and resource consumption could be found. For this purpose, we are working on a new statistical mechanism for gathering the evidences. This mechanism aims at enhancing the scalability of our system by restricting the interrogation to a limited subset of nodes during the diagnostic. Moreover, until now, we assume that honest parties are passive; they alarm others but do not react. We are considering the coupling with countermeasures (e.g., blacklisting) and the exploring of lightweight binary consensus and trust establishment among the nodes that participate in the intrusion detection.

References

[1] Z. Zhao, H. Hu, G. Ahn, and R. Wu, "Risk-aware mitigation for manet routing attacks," *IEEE Transactions on Dependable and Secure Computing*, vol. 9, no. 2, pp. 250–260, 2012.

[2] B. Sun, L. Osborne, Y. Xiao, and S. Guizani, "Intrusion detection techniques in mobile ad hoc and wireless sensor networks," *IEEE Wireless Communications*, vol. 14, no. 5, pp. 56–63, 2007.

[3] S. Sesay, Z. Yang, and J. He, "A survey on mobile ad hoc wireless network," *Information Technology Journal*, vol. 3, no. 2, pp. 168–175, 2004.

[4] P. Brutch and C. Ko, "Challenges in intrusion detection for wireless ad-hoc networks," in *Proceedings of the Symposium on Applications and the Internet Workshops (SAINT '03)*, pp. 368–373, 2003.

[5] S. Khan, N. Alrajeh, and K. Loo, "Secure route selection in wireless mesh networks," *Computer Networks*, vol. 56, no. 2, pp. 491–503, 2012.

[6] S. Corson and J. Macker, "Mobile ad hoc networking (manet): routing protocol performance issues and evaluation considerations," IETF RFC2501, 1999.

[7] Y. Zhang, W. Lee, and Y. Huang, "Intrusion detection techniques for mobile wireless networks," *Wireless Networks*, vol. 9, no. 5, pp. 545–556, 2003.

[8] B. C. Cheng and R. Y. Tseng, "A context adaptive intrusion detection system for MANET," *Computer Communications*, vol. 34, no. 3, pp. 310–318, 2011.

[9] T. Eissa, S. A. Razak, and M. D. A. Ngadi, "Towards providing a new lightweight authentication and encryption scheme for MANET," *Wireless Networks*, vol. 17, no. 4, pp. 833–842, 2011.

[10] A. Irshad, W. Noshairwan, M. Shafiq, S. Khurram, E. Irshad, and M. Usman, "Security enhancement for authentication of nodes in MANET by checking the CRL status of servers," *Communications in Computer and Information Science*, vol. 78, pp. 86–95, 2010.

[11] D. E. Denning, "An intrusion-detection model," *IEEE Transactions on Software Engineering*, vol. 13, no. 2, pp. 222–232, 1987.

[12] T. Clausen and P. Jacquet, "Optimized link state routing protocol (OLSR)," IETF Experimental RFC 3626, 2003.

[13] P. Albers, O. Camp, J. Percher et al., "Security in ad hoc networks a general intrusion detection architecture enhancing trust based approaches," 2002.

[14] F. Cuppens, N. Cuppens-Boulahia, S. Nuon, and T. Ramard, "Property based intrusion detection to secure OLSR," in *Proceedings of the 3rd International Conference on Wireless and Mobile Communications (ICWMC '07)*, Guadeloupe, France, March 2007.

[15] W. Cohen, "Fast effective rule induction," in *Proceedings of the International Conference on Machine Learning (ICML '95)*, 1995.

[16] S. Sarafijanovic and J.-Y. le Boudec, "An artificial immune system for misbehavior detection in mobile ad-hoc networks with virtual thymus, clustering, danger signal and memory detectors," in *Proceedings of the 3rd International Conference on Artificial Immune Systems (ICARIS '04)*, 2004.

[17] S. Sen and J. Clark, *Intrusion Detection in Mobile Ad Hoc Networks in Guide to Wireless Ad Hoc Networks*, Springer, London, UK, 2009.

[18] D. Subhadrabandhu, S. Sarkar, and F. Anjum, "A Framework for misuse detection in ad hoc networks—part II," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 2, pp. 290–303, 2006.

[19] L. M. Feeney and M. Nilsson, "Investigating the energy consumption of a wireless network interface in an ad hoc networking environment," in *Proceedings of the 20th Annual Joint Conference of the IEEE Computer and Communications Societies*, pp. 1548–1557, April 2001.

[20] A. P. Lauf, R. A. Peters, and W. H. Robinson, "A distributed intrusion detection system for resource-constrained devices in ad-hoc networks," *Ad Hoc Networks*, vol. 8, no. 3, pp. 253–266, 2010.

[21] N. A. Alrajeh, S. Khan, J. Lloret, and J. Loo, "Secure routing protocol using cross-layer design and energy harvesting in wireless sensor networks," *International Journal of Distributed Sensor Networks*, vol. 2013, Article ID 374796, 11 pages, 2013.

[22] S. Sen, J. Clark, and J. Tapiador, *Security Threats in Mobile Ad Hoc Networks, I. S. of Self-Organizing Networks*, Auerbach Publications, 2010.

[23] P. Ning and K. Sun, "How to misuse AODV: a case study of insider attacks against mobile ad-hoc routing protocols," *Ad Hoc Networks*, vol. 3, no. 6, pp. 795–819, 2005.

[24] T. Clausen, C. Dearlove, P. Jacquet, and U. Herberg, "The optimized link state routing protocol version 2," IETF Active Internet-Draft Draft-Ietf-Manet-Olsrv2-15, 2012.

[25] U. Herberg and T. Clausen, "Security issues in the optimized link state routing protocol version 2 (olsrv2)," *International Journal of Network Security & Its Applications*, vol. 2, no. 2, 2010.

[26] Y. Owada, T. Maeno, H. Imai, and K. Mase, "OLSRv2 implementation and performance evaluation with link layer feedback," in *Proceedings of the International Wireless Communications and Mobile Computing Conference (IWCMC '07)*, pp. 67–72, New york, NY, USA, August 2007.

[27] F. Hong, L. Hong, and C. Fu, "Secure OLSR," in *Proceedings of the 19th International Conference on Advanced Information Networking and Applications (AINA '05)*, pp. 713–718, March 2005.

[28] L. N. A. M. Hegland, P. Spilling, and Q. Kure, "Hybrid protection of olsr," in *Electronic Notes in Theoretical Computer Science*, 2006.

[29] P. Kunwar, S. Sofat, and D. Bansal, "Comparison of secure olsr routing protocol," *International Journal of Engineering Science*, vol. 3, no. 6, p. 5049, 2011.

[30] A. Adnane, C. Bidan, and R. T. de Sousa Junior, "Trust-based countermeasures for securing OLSR protocol," in *Proceedings of the 7th IEEE/IFIP International Conference on Embedded and Ubiquitous Computing (EUC '09)*, pp. 745–752, August 2009.

[31] B. Wu, J. Chen, J. Chen, J. Wu, and M. Cardei, "A survey of attacks and countermeasures in mobile ad hoc networks," in *Wireless Network Security*, Springer, New York, NY, USA, 2007.

[32] C. Adjih, D. Raffo, and P. Mühlethaler, "Attacks against OLSR: distributed key management for security," in *Proceedings of the OLSR Interop Workshop*, 2005.

[33] G. Cervera, M. Barbeau, J. Garcia-Alfaro, and E. Kranakis, "Mitigation of topology control traffic attacks in OLSR networks," in *Proceedings of the 5th International Conference on Risks and Security of Internet and Systems (CRiSIS '10)*, October 2010.

[34] P. M. Jawandhiya, M. M. Ghonge, M. S. Ali et al., "A survey of mobile ad hoc network attacks," *International Journal of Engineering Science and Technology*, vol. 2, no. 9, pp. 4063–4071, 2010.

[35] L. M. Feeney, "An energy consumption model for performance analysis of routing protocols for mobile ad hoc networks," *Mobile Networks and Applications*, vol. 6, no. 3, pp. 239–249, 2001.

[36] G. Riley and T. Henderson, "The ns-3 network simulator," in *Modeling and Tools for Network Simulation*, Springer, Berlin, Germany, 2010.

[37] S. Bhattiprolu, E. Biederman, S. Hallyn et al., "Virtual servers and checkpoint/restart in mainstream linux," *SIGOPS Operating Systems Review*, vol. 42, no. 5, pp. 104–113, 2008.

[38] J. Zhang, J. Xing, and Z. Qin, "Taprouter: an emulating framework to run real applications on simulated mobile ad hoc network," in *Proceedings of the 44th Annual Simulation Symposium (ANSS '11)*, pp. 39–46, 2011.

[39] D. Broyles, A. Jabbar, and J. P. G. Sterbenz, "Design and analysis of a 3-d gauss-markov mobility model for highly-dynamic airborne networks," in *Proceedings of the International Telemetering Conference (ITC '10)*, San Diego, CA, USA, 2010.

[40] J. Haerri, F. Filali, and C. Bonnet, "Performance comparison of AODV and OLSR in vanets urban environments under realistic mobility patterns," in *Proceedings of the IFIP Med-Hoc-Net Workshop*, 2006.

[41] C. H. Tseng, T. Song, P. Balasubramanyam, C. Ko, and K. Levitt, "A specification-based intrusion detection model for OLSR," *Recent Advances in Intrusion Detection*, vol. 3858, pp. 330–350, 2006.

[42] S. Kurkowski, T. Camp, and M. Colagrosso, "Manet simulation studies: the incredibles," *SIGMOBILE Mobile Computing and Communications Review*, vol. 9, no. 4, pp. 50–561, 2005.

[43] A. Abdalla, I. Saroit, A. Kotb et al., "Misbehavior nodes detection and isolation for MANETs OLSR protocol," *Procedia Computer Science*, vol. 3, pp. 115–121, 2011.

[44] R. Puttini, J. Percher, L. Me et al., "A modular architecture for distributed ids in manet," in *Proceedings of the International Conference on Computational Science and Its Applications III*, 2003.

[45] A. Adnane, R. T. de Sousa, L. Mé, and C. Bidan, "Autonomic trust reasoning enables misbehavior detection in OLSR," in *Proceedings of the 23rd Annual ACM Symposium on Applied Computing (SAC '08)*, pp. 2006–2013, March 2008.

[46] W. Wang, H. Man, and Y. Liu, "A framework for intrusion detection systems by social network analysis methods in ad hoc networks," *Security and Communication Networks*, vol. 2, no. 6, pp. 669–685, 2009.

[47] H. Deng, R. Xu, J. Li, F. Zhang, R. Levy, and W. Lee, "Agent-based cooperative anomaly detection for wireless ad hoc networks," in *Proceedings of the 12th International Conference on Parallel and Distributed Systems (ICPADS '06)*, pp. 613–620, July 2006.

[48] S. Kurosawa, H. Nakayama, N. Kato et al., "Detecting blackholes attack on AODV-based mobile ad hoc networks by dynamic learning method," *International Journal of Network Security*, vol. 5, no. 3, pp. 338–346, 2007.

[49] W. Wang, H. Wang, B. Wang et al., "Energy-aware and selfadaptive anomaly detection scheme based on network tomography in mobile ad hoc networks," *Information Sciences*, vol. 220, pp. 580–602, 2013.

[50] M.-Y. Su, "Prevention of selective black hole attacks on mobile ad hoc networks through intrusion detection systems," *Computer Communications*, vol. 34, no. 1, pp. 107–117, 2011.

[51] N. Marchang and R. Datta, "Collaborative techniques for intrusion detection in mobile ad-hoc networks," *Ad Hoc Networks*, vol. 6, no. 4, pp. 508–523, 2008.

[52] B. V. R. N. Yadav, B. Satyanarayana, and O. B. V. Ramanaiah, "An efficient intrusion detection system for mobile ad hoc networks," in *Emerging Trends in Computing, Informatics, Systems Sciences, and Engineering*, vol. 151, Springer, New York, NY, USA, 2013.

[53] Y. Huang and W. Lee, "Attack analysis and detection for ad hoc routing protocols," in *Recent Advances in Intrusion Detection*, vol. 3224, Springer, Berlin, Germany, 2004.

[54] G. Vigna, S. Gwalani, K. Srinivasan, E. M. Belding-Royer, and R. A. Kemmerer, "An intrusion detection tool for AODV-based ad hoc wireless networks," in *Proceedings of the 20th Annual Computer Security Applications Conference (ACSAC '04)*, pp. 16–27, December 2004.

[55] M. Ayachi, C. Bidan, and N. Prigent, "A trust-based ids for the aodv protocol," in *Information and Communications Security*, vol. 6476, pp. 430–444, 2010.

[56] S. Şen and J. A. Clark, "A grammatical evolution approach to intrusion detection on mobile ad hoc networks," in *Proceedings of the 2nd ACM Conference on Wireless Network Security (WiSec '09)*, pp. 95–102, March 2009.

[57] S. Marti, T. J. Giuli, K. Lai, and M. Baker, "Mitigating routing misbehavior in mobile ad hoc networks," in *Proceedings of the 6th Annual International Conference on Mobile Computing and Networking (MOBICOM '00)*, pp. 255–265, August 2000.

[58] B. Sun, K. Wu, and U. Pooch, "Routing anomaly detection in mobile ad hoc networks," in *Proceedings of The 12th International Conference on Computer Communications and Networks (ICCCN '03)*, 2003.

[59] S. Bose, S. Bharathimurugan, and A. Kannan, "Multi-layer integrated anomaly intrusion detection system for mobile adhoc networks," in *Proceedings of the International Conference on Signal Processing, Communications and Networking (ICSCN '07)*, pp. 360–365, February 2007.

[60] P. Yi, Y. Zhong, and S. Zhang, "A novel intrusion detection method for mobile ad hoc networks," in *Advances in Grid Computing*, vol. 3470, pp. 1183–1192, Springer, Berlin, Germany, 2005.

[61] C. Perkins, E. Belding-Royer, and S. Das, "Ad hoc on-demand distance vector (AODV) routing," IETF Experimental RFC 3561, 2003.

[62] D. Johnson and D. Maltz, "Dynamic source routing in ad hoc wireless networks," in *Mobile Computing*, pp. 153–181, Kluwer Academic, 1996.

[63] C. Perkins and P. Bhagwat, "Highly dynamic destinationsequenced distance-vector routing (dsdv) for mobile computers," in *Proceedings of the Conference on Communications Architectures, Protocols and Applications (SIGCOMM '94)*, vol. 24, pp. 234–244, 1994.

[64] T. Joachims, *Making Large-Scale Support Vector Machine Learning Practical*, MIT Press, Cambridge, Mass, USA, 1999.

[65] Y. Yajima and T. Kuo, "Efficient formulations for 1-svm and their application to recommendation tasks," *Journal of Computers*, vol. 1, no. 3, pp. 27–34, 2006.

[66] L. Bononi and C. Tacconi, "Intrusion detection for secure clustering and routing in Mobile Multi-hop Wireless Networks," *International Journal of Information Security*, vol. 6, no. 6, pp. 379–392, 2007.

[67] Y. Huang, W. Fan, W. Lee et al., "Cross-feature analysis for detecting ad-hoc routing anomalies," in *Proceedings of the 23rd International Conference on Distributed Computing Systems (ICDCS '03)*, pp. 478–487, May 2003.

[68] J. Quinlan, *C4. 5: Programs for Machine Learning*, Morgan Kaufmann, Boston, Mass, USA, 1993.

[69] J. B. D. Cabrera, C. Gutiérrez, and R. K. Mehra, "Ensemble methods for anomaly detection and distributed intrusion detection in Mobile Ad-Hoc Networks," *Information Fusion*, vol. 9, no. 1, pp. 96–119, 2008.

[70] M. Wang, L. Lamont, P. Mason, and M. Gorlatova, "An effective intrusion detection approach for OLSR MANET protocol," in *Proceedings of the 13th IEEE International Conference on Network Protocols Workshop (ICNP '05)*, pp. 55–60, November 2005.

[71] C. Y. Tseng, P. Balasubramanyam, C. Ko, R. Limprasittiporn, J. Rowe, and K. Levitt, "A specification-based intrusion detection system for AODV," in *Proceedings of the 1st ACM Workshop on Security of Ad Hoc and Sensor Networks*, pp. 125–134, October 2003.

[72] S. A. Razak, S. M. Furnell, N. L. Clarke, and P. J. Brooke, "Friend-assisted intrusion detection and response mechanisms for mobile ad hoc networks," *Ad Hoc Networks*, vol. 6, no. 7, pp. 1151–1167, 2008.