HAL
open science

Reliable Trust Estimation in ad hoc Networks using Confidence Interval

Mouhannad Alattar, Francoise Sailhan, Julien Bourgeois

To cite this version:

Mouhannad Alattar, Francoise Sailhan, Julien Bourgeois. Reliable Trust Estimation in ad hoc Networks using Confidence Interval. Conference on Security of Internet of Things (SecurIT), Aug 2012, Kerala, India. pp.1-8. hal-03033122

HAL Id: hal-03033122

https://hal.science/hal-03033122v1

Submitted on 1 Dec 2020

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Reliable Trust Estimation in ad hoc Networks using Confidence Interval

Mouhannad Alattar
FEMTO-ST,
University of Franche-Comté,
Montbéliard, France
fName.lName@univ-fcomte.fr

Françoise Sailhan
Cédric Laboratory,
CNAM,
Paris, France
fName.lName@cnam.fr

Julien Bourgeois
FEMTO-ST,
University of Franche-Comté,
Montbéliard, France
fName.lName@univ-fcomte.fr

ABSTRACT

In an *ad hoc* network, trust systems usually rely on both local and remote evidence in order to build a trust relationship. Remote evidences are usually gathered and concatenated with each other so as to collaboratively infer a trust relation. However, the ever-changing topology and the high versatility of wireless links (coming from e.g., presence of interferences and noise) imply that evidence is partial. Hence the resulting trust relation varies from time to time, depending on e.g., the connectivity. To tackle this issue, we exploit the notion of confidence interval that corresponds to an interval estimate of a parameter (e.g., mean, standard deviation) that characterizes the evidence population. We herein consider this interval to indicate the reliability of the trust estimate and inflect the decision making of an intrusion detection system. More particularly, this detection system detects attacks threatening a routing protocol. This detector distinguishes itself by adopting a mechanism of punishment based on the level of harm produced by the intruder. In a nutshell, this harmfulness is measured by the number of misuse goals (e.g., route disruption, exhausting resources) that could be realized as a result of the intrusion. Performance evaluations of our intrusion detector along with the confidence measure have been also conducted.

Keywords

Trust, Confidence interval, ad hoc routing, Misuse detection.

1. INTRODUCTION

A Mobile *ad hoc* network (Manet for short) is an infrastructure less network, which is generally composed of limited resources nodes (e.g., laptops, smart phones, PDA). Intermediate nodes relay packets so as to increase the communication range of the nodes. Compared to infrastructure-based network, Manet is vulnerable to security threats because of the absence of centralized administration/security enforcement points e.g., switches and routers, from which preventive strategies can be launched [18]. Thus, Intrusion Detection Systems (Ids for short) [9] should be coupled with the preventive techniques that rely on e.g., firewall so as to constitute a second line of defense. In Manet, the development of Idss faces two challenges. First, an Ids gathers continually the evidences from other nodes in order to correlate it and hence detect attacks. This operation is bandwidth- and energy-consuming. Second, the misbehaving/compromised nodes may supply incorrect evidences so as to praise an attacker or accuse a legitimate node. To tackle these issues, we propose a log-based intrusion detection system that is further coupled with an entropy-based trust system. We exemplify our system on the Optimized Link State Routing (Olsr) routing protocol [6] focusing on a link spoofing attack. The general idea lies in analyzing the local logs so as to find evidence of suspicious activity. This activity is modeled as a series of events that match, potentially partially, an attack signature. When local evidences are not sufficient, additional second-hand evidences are requested. Each second-hand evidence is then pondered with the trustworthiness of its source in order to reduce the impact of false accusations, false praises.

Note that gathering second-hand evidence is extremely costly in term of resources consumption: it involves requesting all the neighbors of the suspicious nodes. In order to precisely quantify the amount of evidence that needs to be collected while keeping to a minimum the number of false positives and false negatives, we rely on the confidence interval which corresponds to a standard measure of precision used in statistic. This idea distinguishes itself by the fact that it permits to define a range of confidence (e.g., 95%) within which one is confident that the true value lies, i.e., that the decision on whether an intrusion detection takes place. As a result, the Ids gathers second-hand evidence as long as the width of the confidence interval does not fall into a specific threshold (i.e., the accuracy becomes sufficient).

The reminder of this paper is organized as follows. We first survey the attacks that target routing protocols (§2). Then, we present our intrusion detection system (§3) as well as the proposed trust system (§4). We further evaluate the performance of these latter (§5). An overview of the related work is given in (§6). Finally, we conclude with a summary of our results (§7).

2. VULNERABILITIES

Ad hoc routing protocols constitute a key target for attackers because: (i) no security countermeasure is speci-

fied/implemented as a part of the published RFCs, (ii) the absence of a centralized infrastructure complicates the deployment of preventive measures e.g., firewalls, and (iii) devices operate as routers, which facilitates the manipulation of messages and more generally the compromising of the routing. We hereafter illustrate our presentation by exemplifying attacks on a proactive protocol, OLSR [6].

2.1 Background on OLSR

OLSR aims at maintaining a constantly updated view of the network topology on each device. One fundamental principle is the notion of multipoint relay (MPR): each device selects a subset of 1-hop neighbors, the MPRs, that are responsible for forwarding the control traffic. The idea is to select the minimum number[1] of MPRs that cover 2-hops neighbors so as to reduce the number of nodes retransmitting messages and hence keep to a minimum the bandwidth overload. In practice, a node N selects MPRs among the 1-hop neighbors that are announced in periodic heartbeat messages, termed *hello* messages. Then, a *Topology Control* (TC) message intended to be diffused in the entire network, is created by the selected MPR(s). In this message, a MPR declares the nodes (including N) that selected itself to act as a MPR. Then, any device can compute the shortest path, represented as a sequence of MPRs, to any destination. In addition, recent versions of the 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 regularly by MPRs so that one another maps multiple interfaces with a main address. Thus, a unique identification is provided. Additional extensions have been devised in compliance with the above-summarized core functions. Examples include (i) dealing with the nodes that commit (or not) to carry the traffic for others, and (ii) supporting the interconnection of an OLSR MANET with another routing domain. With the former, nodes advertise their 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). In particular, any gateway with associated host(s) and/or network(s) generates periodically a HNA (Host and Network Association) message including those host(s) and/or network(s) (i.e., the related network address and netmask); this message being further disseminated by MPRs. Overall, these core and auxiliary functionalities are together subject to various attacks.

2.2 Attacks Targeting OLSR

OLSR is vulnerable to a wide range of attacks, which are hereafter sub-classified according to the action which is undertaken on the routing [15]:

- *Drop attacks* consist in dropping routing message(s).
- *Active forge attacks* generate novel and deceptive routing message(s).
- *Modify and forward attacks* modify a received routing message(s) before forwarding it.

[1]Redundant MPRs may be selected to increase the availability.

We hereafter detail each of those attacks.

Drop attack comes from a node that drops a message instead of relaying it. Threatened messages are restricted to the messages that are created and relayed by MPR, i.e., TC, MID, and HNA messages. Consider a host H that sends a message which intended to be forwarded. This message is received by the intruder I that drops it. In practice, I drops a message if I does not forward it within the maximum allowed period. Convenient drop is due to a packet that is empty, expired (as indicated by the time to live field), duplicated or out of sequence. In addition, restrictive forwarding may apply; only the MPR(s) forward messages. Otherwise, remaining drops come from either a selfish/faulty node or an attacker. An attempt to drop any packet is termed *black hole* whereas selective dropping is named *gray hole*. Rather than dropping traffic, an opposite behavior consists in introducing falsified routing information.

Active forge comes from a node that injects novel and deceptive routing messages. Among others, the broadcast storm aims at exhausting resources (e.g., energy). For this purpose, an intruder I forges a large number of control messages within a short period of time. This attack may be conducted in a distributed manner with several nodes colluding so as to emit (a large number of) messages. In order to reduce the visibility of this attack, I typically masquerades itself. In practice, the masquerade lies in sending a message including a switched identification. Note that this case should be distinguished from a node that holds several interfaces and advertises them in the dedicated MID message. Apart from masquerading, identity spoofing may be intended to create conflicting route(s) and potentially loop(s). This spoofing attack may also be coupled with a modification of the willingness field so as to impact the selection of MPR. MPRs are selected among the nodes with the highest willingness and in case of multiple choices, the node providing a reachability to the maximum number of 2-hops nodes is primarily selected. For instance, a node whose willingness is minimal (resp. maximal), is never (resp. always) selected as a MPR. In addition, active forges cover the message tempering with incorrect adjacent links (*hello* messages), topology information (TC messages), and network interfaces (MID) and routes (HNA messages). With the former, I forges a *hello* message, which declares 1-hop and symmetric neighbors differing from the real one. When forging this set of symmetric neighbors, the attacker has 3 options:

- declaring a non-existing node as a symmetric neighbor, implies that I (or another misbehaving node) is further selected as a MPR. Indeed, if I advertises a non-existing node, I ensures that no other (well-behaving) MPR claims being a 1-hop symmetric neighbor of that node. Recall that MPRs are selected so that all the 2-hops and symmetric neighbors are covered, I is hence selected as a MPR. Note that this is verified as long as no other misbehaving neighbor claims the same. Overall, inserting at least one non-existing neighbor guaranties that a misbehaving node is selected to act as a MPR. In addition to the above, the connectivity of I increases.
- declaring that an existing node is a symmetric 1-hop neighbor (whereas it is not the case) increases artificially the connectivity of I. If no (well-behaving) MPR

covers the declared node, then at least one misbehaving node is selected as a MPR. This typically characterizes an attempt to create a blackhole: I introduces a novel path that provisions the blackhole.

- omitting an existing 1-hop symmetric neighbor, decreases artificially the connectivity of the attacked node and I.

Overall, such a falsification of the neighboring adjacency perverts the topology seen by nodes and may impact the selection of MPR(s). Another alternative refers to a node I declaring itself as a MPR although it has not been selected as a MPR forehand: I forges a (TC) message including incorrect 1-hop symmetric neighbor(s), including at least the MPR selector(s) that corresponds to the neighbors that have selected I as a MPR. In particular, possible falsifications lie in inserting a non-existing node or an incorrect but existing node or also omitting an existing node. Due to the lack of space, we do not detail herein each of these cases. Upon the reception of a falsified TC message, routing tables are corrupted and may contaminate any interconnected routing domain if a gateway exports those OLSR routes. Note that the gateway may also forge itself wrong routes. 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 omitting advertising reachable nodes. Symmetrically, an intruder may import incorrect routes to the OLSR domain. Overall, the forge attacks (e.g., route spoofing attacks) necessitate to tamper message while keeping it syntactically correct. In other words, bogus messages can be forged, hence creating implementation-dependent effects. Generally speaking, similar tampering may be performed by a MPR relaying control messages.

Modify and forward attacks are characterized by an intermediate that captures the control message and replays or/and modifies this message before forwarding it. Replaying a message includes delaying (i.e., forwarding latter potentially in another area) and repeating this message. As a result, routing tables are updated with obsolete information. Both attacks can be performed in a distributed manner with two intruders: one records the message from one region so as to replay it in another region (i.e., the one of the colluding intruder); this leads to the creation of a *wormhole*. In order to stay invisible, both intruders may keep the identification unchanged. Note that sequence numbers constitute a standard mechanism that provides protection against replay attacks. Based on those numbers, a node identifies freshest information, prevents duplicates and replaying while indicating insertion/deletion. In counterpart, there usage may be hijacked. For instance, an intruder I may forwards the message including an increased sequence number. Thus, the source assumes that I provides the freshest route.

3. INTRUSION DETECTION

In order to deal with such attacks, we propose IDAR, a distributed, log- and signature-based intrusion detection system that periodically collects the OLSR logs. These logs characterize the activities of OLSR (e.g., packet reception, MPR selection). Note that additional logs, e.g., 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 predefined signatures; a signature is thought as a partially ordered sequence of events that characterizes a misbehaving activity. Detection is potentially not only a memory but also a bandwidth-consuming: it may involve not only examining logs but also requesting others to collect/correlate additional intrusion evidences. Thus, this activity should be carefully-planned, i.e., initiated only when sufficient suspicion exists and terminated as soon as a result is obtained. Toward this goal, evidences are classified so that depending on their level of gravity, additional in-depth detection is whether performed. They fall into the following groups:

- *Suspicious-evidence-group* contains the evidences necessary to identify a node as suspicious,

- *Initial-evidence-group* contains the evidences necessary to identify a suspicious node and launch a networked investigation,
  - *Confirming-evidence-group* contains the evidences that confirm the occurrence of an attack. This results in terminating the investigation and declaring the suspicious node as an intruder.

- *Canceling-evidence-group* contains the evidences that eliminate the suspicion, which ends the investigation.

These groups are populated with the evidence extirpated from logs. If an evidence belonging to the initial-evidence-group is discovered then an advanced investigation is launched so as to confirm (confirming-evidence-group) or infirm (canceling-evidence-group) intrusion; both resulting in terminating the investigation. Relying on these groups, the gradual evolving of the attack and of its related detection are easily followed. In addition, its compact form facilitates the lightweight discovering of long-terms intrusions. But before delving into the functioning of the above groups, let first exemplify the proposed intrusion detection system with the link spoofing attack we purposely developed.

3.1 Link Spoofing Detection

Link spoofing aims at inflecting the MPR selection; such selection is triggered upon a change in the symmetric 1- and 2-hops neighbors. Rather than launching an in-depth investigation upon these changes, we minimize the investigation by initiating it only when the event that occurs is relevant to a link spoofing attack. We ignore changes in the 1-hop neighborhoods (e.g., node apparition) because they are observed by the node itself and are hence not subject to remote falsification; a cornerstone of a link spoofing. In addition, changes in the 2-hops neighborhood are considered if they impact the MPR selection. Those include:

- A replaced MPR (Evidence 1, $E1$ for short) means that a change in the covering of 1-hop neighbors leads to this replacement. This comes from 1-hop neighbor(s), possibly the replacing MPR, that increase(s)/decreases it/their coverage(s) to the detriment of the replaced MPR.

- No MPR replacement takes place but a previously-selected MPR is detected as misbehaving. Messages are dropped, forged or misrelayed by that MPR ($E2$). Overall, a link spoofing also covers the case wherein an intruder continues to advertise identical 1-hops neighbors despite recent changes in the neighborhood. Note that contrary to the other evidences listed here, this case is not event-driven and should be handled by launching periodical/random checks.
- a MPR is the only one providing the connectivity to node(s) ($E3$).
- a MPR does not cover its adjacent neighbor(s) ($E4$).
- a MPR provides connectivity to a non-neighbor ($E5$).

$$\begin{array}{cc} (E1 \vee E2)\ ,\ \text{optional}(E3) & \\ \Downarrow & \Downarrow \\ E4 \vee E5 & (!E4 \wedge !E5) \\ \Downarrow & \Downarrow \\ \text{The suspicious MPR} & \text{The suspicious MPR} \\ \text{is an intruder.} & \text{is well-behaving.} \end{array} \quad (1)$$

The occurrence of $E1$ or $E2$ is the starting point for further investigation; $E1$, $E2$ hence belong to the initial-evidence-group. Note that a MPR that is the only one providing the connectivity to node(s) ($E3$) is suspicious but this condition is not sufficient to launch an investigation: (i) this situation is typical in a sparse network and (ii) 2 nodes within communication range often fail in communicating due to the unpredictable nature of wireless transmission resulting from, e.g., obstacles, noises. Thus, diagnosing $E3$ is especially difficult under no specific assumption. Overall, the occurrence of either $E1$ or $E2$ and optionally $E3$ leads to a collaborative investigation which consists in interrogating the neighbor(s) of the suspicious MPR so as to discover whether the suspicious MPR does not cover its neighbors ($E4$) or advertises a distant node ($E5$).

3.2 Collaborative Investigation

A collaborative investigation works as follows: the investigator interrogates the 1-hops neighbor(s) of the suspicious MPR so as to glean additional evidences concerning the intrusion. If all the requested nodes confirm (resp. infirm), then the MPR is suspected (resp. defined as well-behaving). Note that, part of those requested nodes may express different opinions, which may result from e.g., the nodes mobility or misbehavior. In order to tackle this issue, their respective reputations is taken into account, as described in Section 4. In practice, the interrogation of a 2-hops neighbors, denoted Ai, consists in sending a request to Ai. Note that wherenever possible, this request should not go through both the suspicious MPR I or any of its colluding intruder I'_j. This avoidance is necessary so as to prevent I and I'_j from dropping the request and/or simply forging a defective answer. For this purpose, a 1-hop neighbor (primarily the MPR) that covers the requested 2-hops neighbors is provided the request. If no answer is obtained (i.e., when the related time-out elapses), then the demand is sequentially transferred through the rest of the covering 1-hop neighbors (as aforementioned, MPRs being primarily selected). Note that this verification is performed within a thread so that the investigation of one node (and the result waiting) is not blocking to others. If no neighbor is left, then a (multi-hops) alternative path is researched in the routing table to reach Ai. Based on the answer provided by all the Ai, the suspicious MPR is defined either as well-behaving or not. Note that if the number of answers is not sufficient, then the suspicious MPR is tagged as unverified. Such a collaborative detection may be especially resource consuming. It involves interrogating all the neighbors A_i of the suspicious node and this interogation may involve multi-hops communications. As a result, the greater gets the netwok density, the greater gets the bandwidth usage. This calls for keeping to a minimum the number of interogations while garantying that the investigation is not affected.

3.3 Optimised Investigation

In order to support lightweight intrusion detection, we propose to probabilistically interrogate the nodes (that are participating in the verification). This consists in randomly and uni-formally selecting some node among the 2-hops neighbors of the suspicious MPR. The advantage of that method is that fewer nodes are contacted comparing to a deterministic approach. Let b be the number of neighbors of the suspicious MPR, the expected cost, in terms of number of messages to send, is equals to $p.b$, with p defining the probaility of a node to be interrogated. Nevertheless, multi-hops interrogation should be considered.

3.3.1 Probabilistic Confidence-based Optimization

A probabilistic verification involves requesting in a random manner a subset of neighbor.

3.3.2 Level of Confidence

The *confidence interval* [19] corresponds to an estimated range which is likely to include an unknown population parameter, the estimated range being calculated from a given set of sample data. Herein, based on a partial set of evidences $e_1, ..., e_n$ (namely the sample), our objective lies in estimating a range wherein the overall population of evidences is likely to fall. This range; called the confidence interval, is given by $[\overline{Detect}^{A,I}_{\Delta t} - \varepsilon, \overline{Detect}^{A,I}_{\Delta t} + \varepsilon]$ with ε defining the allowed margin of error. The likelihood that the overall population falls into the confidence interval corresponds to a probability, e.g., 95%, named confidence level (cl for short). This confidence level is a configuration parameter of the trust system. Given a standard deviation σ and the standard density z of a normal law, the margin of error ε is calculated as follows:

$$\varepsilon = z\frac{\sigma}{\sqrt{n}} \quad (2)$$

With a $\sigma = \sqrt{\frac{\sum_{i=0}^{n}(\bar{m} - e_i^{A,S_i})^2}{n-1}}$ and $\bar{m}$ corresponding to the mean. It follows that the higher is the confidence level, the wider gets the confidence interval. Overall, the confidence interval depends of three factors: the required confidence level that is parameterized, the number of observations that is provided, the spread of the observations. As such it permits to guide the decision of establishing if an attack takes place whereas a limited number of recommendations is provided by potentially well-behaving and misbehaving nodes. If the confidence interval is too wide then more evidence

should be provided to estimate the trustworthiness in routing (or other operation) decisions. For this purpose, we apply the following rule:

$$\begin{cases} I \text{ is well-behaving} & \text{if} \quad \gamma \leq Detect_{A,I} - Ci \leq 1 \\ I \text{ is intruder} & \text{if} \ -1 \leq Detect_{A,I} + Ci \leq -\gamma \\ I \text{ is unrecognized} & \text{if other} \end{cases} \tag{3}$$

When the investigation falls into the unrecognized range then more evidence should be collected in order to confirm/refuting the existence of the intrusion. However, linking the investigation to the trustworthiness of the nodes increases the accuracy of detection as it will be shown in the next section.

4. TRUSTED AND LOW-RESOURCE CONSUMING INTRUSION DETECTION

In order to prevent as much as possible misbehaving nodes from foiling the intrusion detection while supporting a ligh-weight detection intrusion, we propose to:

- evaluate the trustworthiness of the node(s) that provide second-hand observations. The objective is to favor the observations provided by trustworthy nodes while being detrimental to misbehaving nodes. A trust system aims at expressing the opinion about another node based on the actions of that node. It permits to decide whether a detection provided by a possibly unknown node is advisable or not [17].
- a probabistically select the nodes that are participating in collecting the observations. For this purpose, we measure the level of confidence on the detection.

But before delving into the details, let us first introduce the trust esttablishment.

4.1 Trust establishment

We propose a distrubted trust system that establishes trust relations between the devices depending on the past behavioral evidences. A trust relation $T_{A,I}$ that is established between two nodes A and I represents to which extend A beliefs that I acts as expected. This belief is built according to I 's previous activities [3]: based on the evidence that is collected, the system evaluates the trustworthiness. Generally speaking, several properties should be taken into account during the establishment of the trustworthiness:

- **Properties 1**: the beneficial activity that is performed by a node increases the trustworthiness of that node, whereas a harmful activity decreases this trustworthiness. Examples of beneficial activity includes the normal relaying of the traffic. In contrast, an harmful activity related to e.g., an intrusion or the supplying of an incorrect answer/feedback to an investigation request.
- **Properties 2**: the degree of gravity (versus reputability) of a harmful (versus beneficial) activity influences the risk for other nodes and hence should be reflected in the establishment of the trust. For instance, relaying correctly the packets is less reputability than supplying correct answer to an investigation request.
- **Properties 3**: the risk characterized by the imminence of the intrusion decreases drastically the trustworthiness.
- **Properties 4**: fresh activities should be privileged in opposition to stale activities.
- **Properties 5**: first hand evidences (i.e., evidences that are gleaned by the node itself) are privileged comparing to the second hand information which are subject to controversial.

The above properties are enforced as follows. A node A calculates the trust $T_{\Delta t}(A, I)$ of a node I based on the n evidences $e_1^{A,I}, ..., e_i^{A,I}, ..., e_n^{A,I}$ about I that are collected by A during a time slot Δt:

$$T_{\Delta t}^{A,I} = \sum_{j=0}^{n} \alpha_j e_j^{A,I} + \beta T_{\Delta(t-1)}^{A,I} \tag{4}$$

Beneficial and harmful evidences $e_j^{A,I}$ take respectively positive and negative values (property 1). A weighting factor α_j pondered $e_j^{A,I}$ so as to reflect the degree of gravity/reputability of this evidence (property 2) as well as the risk (property 3). Meanwhile, a forgetting factor β permits to privilege fresh evidences rather than the stale evidences that were computed at the previous time slot $\Delta(t-1)$ (property 4). When the observations of A are not sufficient, additional evidences provided by other nodes are gleaned. These evidences are less reliable than the local observations. Thus, an uncertainty is involved. To compute such an uncertainty, we rely on the *entropy*, a measure of uncertainty stated in information theory [8]. As in [14, 23], we establish trust through a third party (termed *concatenated propagation*) and through recommendations provided by multiple sources (called *multipath propagation*). Let a *recommendation* $R_{A,S}$ represent how much A trusts the recommendations generated by S. A builds its belief about I according to a third party S's recommendation as follows:

$$Tc_{\Delta t}^{A,I} = R_{\Delta t}^{A,S} T_{\Delta t}^{S,I} \tag{5}$$

When multiple nodes $S_1, S_2, ..., S_m$ generate a recommendation, A builds its belief about I as follows:

$$Tm_{\Delta t}^{A,I} = \sum_{i=1}^{m} w_i . R_{\Delta t}^{A,S_i} . T_{\Delta t}^{S_i,I} \tag{6}$$

where $w_i = \frac{1}{\sum_{j=0}^{m} R_{A,S_j}^{\Delta t}}$

Let us illustrate the establishment of the trust relationship by considering a link spoofing attack that is taking place between a suspicious node I and its neighbor S. In order to establish whether I is launching a link spoofing, the neighbors $S_1, ..., S_m$ of I are interrogated so as to provide the second-hand evidences noted $e^{S_1,I}, ..., e^{S_i,I}, ..., e^{S_m,I}$. These evidences are aggregated (Formula 7), i.e., each evidence $e_i^{S_i,I}$ is pondered with a weighting factor w_i along with the

trust T_{A,S_i} that the investigator A shares with the requested neighbour S_i:

$$Detect_{\Delta t}^{A,I} = \sum_{i=1}^{m} w_i T_{A,S_i} e_i^{S_i,I} \quad (7)$$

with $w_i = \frac{1}{\sum_{j=0}^{m} T_{A,S_j}}$.

An evidence $e_i^{S_i,I}$ is either equal to (1) - The link which is advertised by I is correct: I does not carry a spoofing attack - or in the contrary to (-1) - the advertised link is wrong. Note that if a requested node S_i does not return an answer (before the waiting time elapses) then $e_{S_i} = 0$. As a result, a link spoofing is detected when $Detect_{\Delta t}^{A,I}$ is nearly equal to (-1). Once stated, this result is used to update the trustworthiness of I and $S_1, ..., S_m$.

5. PERFORMANCE EVALUATION

We hereafter evaluate the performance of the trust system ; one may find in [2] a detailed evaluation of the intrusion detection. Experiments are performed as follows. We consider, unless specified, 16 nodes including 1 attacker that performs a link spoofing attack and 4 colluding misbehaving nodes (i.e., 26.3% of the overall nodes). These misbehaving nodes, also called liars, provide incorrect answers in order to foil the detection. Initially, each node is assigned with a random trust value. The attacker launches a link spoofing attack that, unless specified, takes place during the overall experiment. Similarly, misbehaving nodes supply incorrect answers during the overall experiments. The evolution of trustworthiness as seen by the node under attack is presented in Figure 1. The constant maintaining of the well-behaving and misbehaving explains the (monotonous) ascending versus descending rate of the node trustworthiness. One may also note the defensive nature of our trust system: the trust associated to a liar falls dramatically regardless of the initial trust, whereas the well-behaving nodes owning a low initial trust gain moderately the trustworthiness. Then, if the attack ceases (Figure 2), both liars and well-

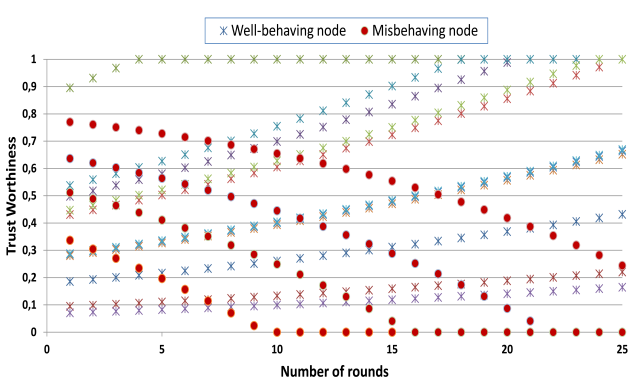

Figure 1: Trustworthiness

behaving nodes recover due to the forgetting factor. One may note that the nodes with high or medium initial trust values reach the default trust value (herein 0.4) in the latest rounds. Nodes with low initial trust values recover slowly. This represents again the defensive nature of our trust system which demands a long misconduct-less duration before trusting a former liar. In order to evaluate the impact of

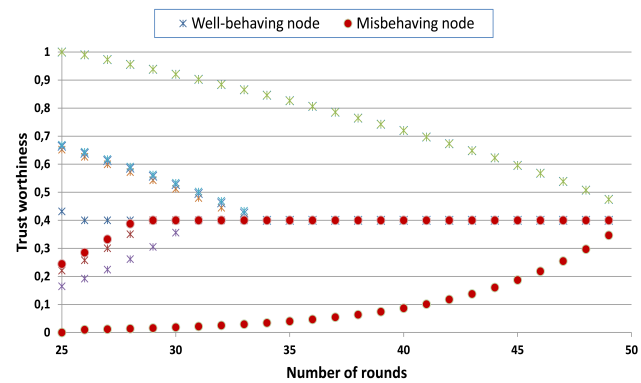

Figure 2: Impact of the Forgetting Factor on the Trustworthiness

the liars, the result of the detection is provided for several percentages of liars (Figure 3). Note that a detection result approaching (-1) reflects the certainty of an attack. As expected, the greater the number of liars, the slower the detection. After 10 rounds, the detection result falls even when

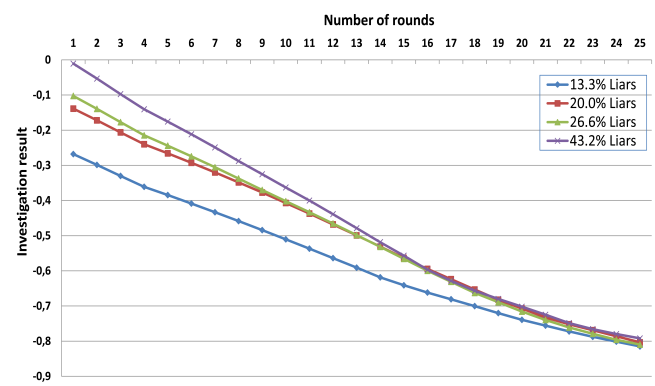

Figure 3: Impact of Liars on the Detection

the liars constitute 43.2% of the nodes. More precisely, during the last rounds, the detection converges to -0.8 regardless of the percentage of liars because the trustworthiness of those liars diminishes. Thus, liars do no longer influence on the detection in the last rounds. Figure 4 presents the impact of the number of nodes on the margin of error CI which is used to estimate the confidence interval. When 4 nodes participate to the investigation , there exists a significant drop in the margin of error. This results from the decline of the trustworthiness that is characterizing liars: the standard deviation of the returned answers, which are pondered with the trustworthiness, decreases. The smaller is the deviation in the pondered answers, the smaller is the margin of error. In the other side, a small sample data leads to a high margin of error. Overall, a node cannot misbehave and keep a high trustworthiness. Thus, the impact of liars on the detection decreases. It is worth to be mentioned that an investigation does not necessary interrogates all the available nodes : estimating the confidence interval permits us to find a trade-off

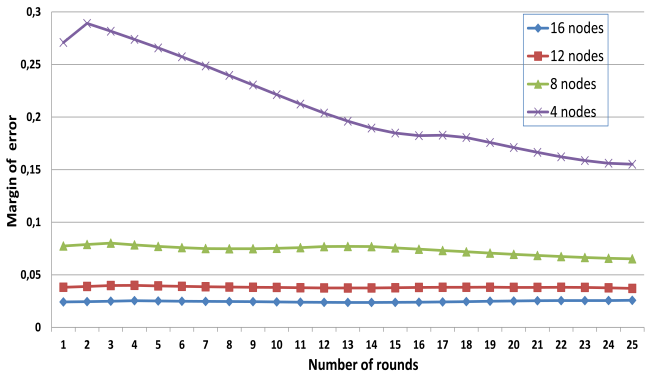

Figure 4: Impact of Liars on the Detection

between the accuracy of/confidence in the detection and the resulting traffic.

6. RELATED WORK

Systems that detect intrusions targeting *ad hoc* routing protocols are diverse in the way they analyze the intrusion. They fall into 3 categories: anomaly, specification- and signature-based detection. **Anomaly detection** constitutes the main approach. The idea is to define the correct behavior of a node and detect deviations from this behavior. This correct behavior is automatically built during an attack-less phase. In [24], attempts to falsify the routes are detected. During the training phase, the impact of the movement on the percentage of updates in the routing table is analyzed. Then, during operation, an actual percentage of updates differing from the predicted one, is defined as an anomaly; the distinguish is provided by the Support Vector Machine (Svm) Light [12] classifier or a rule-based engine Ripper [7] . In [13] and resp. [4], a blackhole (and resp. dropping) attack targeting the Aodv protocol (resp. a secured version of Aodv) are detected by investigating features, e.g., the number of route requests and route replies as well as the average difference of sequence numbers[2]. If the distance between observed features and the average ones, exceeds a given threshold, then an intrusion is detected. More sophisticated Cross-Features Analysis (Cfa) [11] is applied relying on the C4.5 [16] decision tree classifier so as to detect both blackhole and packet dropping on Aodv and Dsr protocols. Cfa and C4.5 are also used for Olsr [5]. Rather than establishing automatically a correct behavior, **specification-based** systems hand-code this behavior relying on the protocol specification. Then, the system detects a violation of constraints circumventing this behavior. Example of constraints defining the correct behavior of Olsr[10, 22] includes the fact that a Mpr and a node that selects the Mpr must be adjacent. These constraints are modeled using semantic properties [22], rules [10], or finite state machines [20]. **Signature-based detection** models the way an intruder penetrates the system by defining intrusion signatures. Then, any behavior that is close to this predefined signature is flagged as intrusion. Finite-state-machine is used to detect network flooding, dropping and spoofing attacks, which target Aodv [21]. Sensors observe the traffic and match it against predefined signatures. They also exchange Mac and Ip addresses to detect identity spoofing realized by by a node emitting a packet identified with Mac or Ip addresses differing from those registered. Rule-based signatures are specified to detect attack on Olsr [1] in opposition to the legitimate behavior depicted by a prior specification-based Ids [22]. This detection is further coupled with a trust system: a node mistrusts another that does not conforms to the predefined rules.

[2]Increased sequence numbers are known as a sign of blackhole attack.

7. CONCLUSION

The open medium of communication and the collaborative structure of *ad hoc* networks facilitates the performing of intrusions and attacks. It follows that several/intrusions attacks have been surveyed in the literature. They mostly operate against routing protocols due to the central role of those protocols, which consists in determining multi-hops paths among the devices. In order to detect such intrusions/attacks, many approaches have been proposed. Intrusion/attack may be identified as a deviation of the correct behavior (anomaly detection); this correct behavior is either hand-specified given a protocol description, e.g., [10] or automatically built/analyzed using e.g., machine learning or data mining techniques, e.g., [7]. Alternatively, signatures identifying the way the intruder penetrates the system are further used so as detect an intrusion/attack that is close to that predefined signature. We propose such a signature-based intrusion/attack detector that takes advantage of the logs that are generated by the routing protocol so as to detect intrusion attempts. From a practical point of view, this implies no change in the implementation of the routing protocol and does not necessitate to inspect the traffic as it is the case with other (aforementioned) systems. Such a Idss faces two challenges:

- keeping to a minimum the number of investigations (and hence the computational/bandwidth usage related to gleaning attack/intrusion evidences) while maximizing the detection accuracy,
- dealing with false accusations coming from misbehaving nodes.

In order to precisely quantify the amount of evidences that needs to be collected, we rely on the confidence interval which corresponds to a standard measure of precision used in statistic. This idea distinguishes itself by the fact that it permits to define a range of confidence (e.g., 95%) within which one is confident that the true value lies, i.e., the decision on whether an intrusion detection takes place. Meanwhile, we propose an entropy-based trust evaluation in order to combat colluding attacker(s) that attempt to get an increased influence on the detection by providing fake evidences. This trust system evaluates periodically the trustworthiness of each node. This trustworthiness is then used to ponder the gathered evidences. We further provide an evaluation of the performance of the probabilistic trust system and confidence-enabled intrusion detector. This evaluation assesses the defensive nature of our trust system and the reduced impact of misbehaving nodes. Finally, the confidence interval constitutes an interesting indicator that permits to guide the evidence gleaning and more precisely to cease the

gleaning as soon as possible.
In the near future, more experiences are planned in order to evaluate the impact of mobility on trustworthiness evaluation. In addition, we envisage to evaluate the influence of using the confidence interval on the detection and the resource consumption in a simulated MANET.

8. REFERENCES

[1] A. Adnane, R. Sousa, C. Bidan, and al. Autonomic trust reasoning enables misbehavior detection in OLSR. In *ACM SAP*, 2008.

[2] M. Alattar, F. Sailhan, and J. Bourgeois. Log-based intrusion detection for MANET. In *IWCMC'12, 8-th IEEE Int. Conf. on Wireless Communications and Mobile Computing*. IEEE Computer Society, 2012.

[3] F. Azzedin and M. Maheswaran. Evolving and managing trust in grid computing systems. In *Proceedings of the IEEE Canadian Conference on Electrical & Computer Engineering (CCECE 02)*, 2002.

[4] L. Bononi and C. Tacconi. Intrusion detection for secure clustering and routing in mobile multi-hop wireless networks. *International journal of information security*, 6, 2007.

[5] J. Cabrera, C. Gutierrez, and R. Mehra. Ensemble methods for anomaly detection and distributed intrusion detection in mobile ad-hoc networks. *Information Fusion*, 9, 2008.

[6] T. Clausen and P. Jacquet. Optimized link state routing protocol (OLSR). IETF experimental RFC 3626, October 2003.

[7] W. Cohen. Fast effective rule induction. In *ICML*, 1995.

[8] T. M. Cover and J. A. Thomas. *Elements of information theory*. John Wiley and Sons, 2006.

[9] D. E. Denning. An intrusion-detection model. *IEEE TRANSACTIONS ON SOFTWARE ENGINEERING*, 13, 1987.

[10] F.Cuppens, N.Cuppens-Boulahia, S.Nuon, and al. Property based intrusion detection to secure OLSR. In *IEEE ICWMC*, 2007.

[11] Y. Huang, W. Fan, W. Lee, and al. Cross-feature analysis for detecting ad-hoc routing anomalies. In *IEEE ICDCS*, 2003.

[12] T. Joachims. *Making large-scale support vector machine learning practical*. MIT Press Cambridge, 1999.

[13] S. Kurosawa, H. Nakayama, N. Kato, and al. Detecting blackholes attack on AODV-based mobile ad hoc networks by dynamic learning method. *International journal of network security*, 5(3), 2007.

[14] M.N.K.Babu, A.A.Franklin, and C.S.R.Murthy. On the prevention of collusion attack in olsr-based mobile ad hoc networks. In *Networks, 2008. ICON 2008. 16th IEEE International Conference on*, 2008.

[15] N. Peng and S. Kun. How to misuse AODV: a case study of insider attacks against mobile ad-hoc routing protocols. *Ad Hoc Netw.*, 3(6), 2005.

[16] J. Quinlan. *C4.5: programs for machine learning*. M. Kaufmann, 1993.

[17] S. Ruohomaaand and L. Kutvonen. Trust management survey. In *Proceedings of the Third international conference on Trust Management*. Springer-Verlag, 2005.

[18] S.Corson and J.Macker. Mobile ad hoc networking (manet): Routing protocol performance issues and evaluation considerations, 1999.

[19] M. Smithson. *Confidence Intervales*. Sage University Papers Series on Quantitative Applications in Social Sciences, 2003.

[20] C. Tseng, S. Tao, P. Balasubramanyam, and al. A specification-based intrusion detection model for OLSR. In *RAID*. 2008.

[21] G. Vigna, S. Gwalani, K. Srinivasan, and al. An intrusion detection tool for AODV-based ad hoc wireless networks. In *ACSAC*, 2004.

[22] M. Wang, L. Lamont, P. Mason, and al. An effective intrusion detection approach for OLSR MANET protocol. *IEEE ICNP workshop*, 2005.

[23] YL.Sun, S.Member, Z.Han, and al. Information theoretic framework of trust modeling and evaluation for ad hoc networks. *IEEE Journal on Selected Area in Communications*, 24, 2006.

[24] Y. Zhang and W. Lee. Intrusion detection in wireless ad-hoc networks. In *ACM MobiCom conference*. ACM, 2000.