WIRELESS COMMUNICATIONS AND MOBILE COMPUTING
Wirel. Commun. Mob. Comput. 2008; **8**:689–704
Published online 9 January 2007 in Wiley InterScience
(www.interscience.wiley.com) DOI: 10.1002/wcm.493

Struggling against selfishness and black hole attacks in MANETs

Djamel Djenouri[1]*,† and Nadjib Badache[2]
[1]*Basic Software Laboratory, CERIST Center of Research, Ben-aknoun, BP 143, Algiers 16030, Algeria*
[2]*Computer Science Department, USTHB University, Algiers, Algeria*

Summary

Since mobile ad hoc networks (MANETs) are infrastructureless and multi-hop by nature, transmitting packets from any node to another usually relies on services provided by intermediate nodes. This reliance introduces a new vulnerability; one node could launch a *Black Hole DoS attack* by participating in the routing protocol and including itself in routes, then simply dropping packets it receives to forward. Another motivation for dropping packets in self-organized MANETs is resource preservation. Some solutions for detecting and isolating packet droppers have been recently proposed, but almost all of them employ the promiscuous mode monitoring approach (watchdog (WD)) which suffers from many problems, especially when employing the power control technique. In this paper we propose a novel monitoring approach that overcomes some WD's shortcomings, and improves the efficiency in detection. To overcome false detections due to nodes mobility and channel conditions we propose a Bayesian technique for the judgment, allowing node redemption before judgment. Finally, we suggest a social-based approach for the detection approval and isolation of guilty nodes. We analyze our solution and asses its performance by simulation. The results illustrate a large improvement of our monitoring solution in detection versus the WD, and an efficiency through our judgment and isolation techniques as well. Copyright © 2007 John Wiley & Sons, Ltd.

KEY WORDS: ad hoc networks; security; selfish misbehavior; black hole attack

1. Introduction

The infrastructureless multi-hop nature of MANET causes vulnerabilities to packet dropping attacks (causing a Denial of Service (DoS)) and selfish misbehavior. A node could launch a DoS attack, by simply participating in the routing protocol to include itself in routes then dropping data packets it is asked to forward. This kind of DoS attack is termed black hole attack [1]. On the other hand, in many MANET's applications where nodes typically do not belong to a single authority and do not pursue a common goal, forwarding packets for other nodes is not in the direct interest of anyone. Indeed, each node tries to save its resources, particularly its battery power and bandwidth, and might behave *selfishly* or *non-cooperatively* by not forwarding packets originated from other nodes, while using their resources to send its own packets toward

*Correspondence to: Djamel Djenouri, Basic Software Laboratory, CERIST Center of Research, Ben-aknoun, BP 143, Algiers 16030, Algeria.

†E-mail: ddjenouri@mail.cerist.dz

remote recipients. In the rest of this paper we call both the malicious and the selfish behaviors misbehavior on packet forwarding. Regardless of the motivation of the dropper, this misbehavior threatens the Quality of Service (QoS) and the service availability in the network.

The first solution dealing with this problem is the watchdog (WD) [2], which has been implemented with a source routing protocol Dynamic Source Routing (DSR) and relies on monitoring neighbors in the promiscuous mode. The WD suggests that each node in the source route monitors its successor by overhearing the channel. A monitoring node accuses the monitored one of misbehaving when it detects that this latter drops more than a given number (threshold) of packets. This basic technique of monitoring has no *communication* overhead when nodes do not misbehave. Nevertheless, it suffers from some problems, especially when using the power control technique employed by some new power-aware routing protocols following the WD's proposal [3,4]. Note that the power control technique consists of adapting the transmission power according to the distance separating the transmitter and the receiver, instead of using a fixed full power that covers the maximum physical power range. Assume three aligned nodes: A, B, and C such that A sends B a packet and monitors its forwarding to C, and that B uses the power control technique. When A is closer to B than C, B could circumvent the WD by using a transmission power strong enough to reach A but less than the one required to reach C, which is power efficient for B. On the other hand, when C is closer to B than A and B behaves correctly but uses the power control technique, A could not overhear B's forwarding to C and would wrongly notice packet droppings, which might result in false detections when the counter exceeds the configured threshold. Further, packet collisions either at C or A during the monitoring could cause problems. When B's forwarding causes a collision at C, the former could circumvent to A by not retransmitting the packet. On the other side, if B's forwarding results in a collision at A, this latter could falsely note a packet dropping of B. In addition to the problems related to detection, the WD has a lack of punishment drawback. After a detection of a misbehaving on some route, the source node will be informed, and future packets will be routed around the misbehaving one to avoid dropping. However, the WD does not prevent nodes from misbehaving, as it does not employ any isolation or punishment strategy. Recent solutions, that will be presented later, have dealt with this problem and proposed punishment policies, together with methods to exchange information on misbehaving nodes. Still, almost all these solutions rely on the WD's promiscuous monitoring technique in their monitor components, thus inherent all its drawbacks.

In this proposal, we extend and complete the random two-hop ACKs monitoring approach introduced in Reference [5], and provide a comprehensive solution that integrates the random two-hop ACKs based monitoring module with the judgment and isolation ones. Our novel monitoring approach overcomes the WD's problems with reasonable overhead. Like the WD, it works with a source routing protocol. Regarding the judgment issue, we propose a Bayesian approach for node accusation, enabling node redemption before judgment. Finally, we suggest a social-based approach to approve detections and *safely* isolate guilty nodes. The aim of this approach is to consider and avoid false accusation attacks (rumors) vulnerability, as well as decreasing false positives that might be caused by channel conditions and nodes mobility. Our solution deals with all kinds of packet droppers, including as well as selfish malicious nodes (insiders and outsiders) launching a black hole attack. It also deals with any Byzantine attack involving packet dropping in any of its steps, for example, wormhole (tunneling) and rushing attacks [1]. In this case, our solution detects the attacker when it drops packets. However, dealing with the detection of intruders (compromised nodes) is out of the scope of this work, but there are many IDS proposed for MANET [6] that could be used. Also, note that we are note securing the routing protocol procedures against other attacks such as spoofing. To ensure this, any of the source routing authenticated protocols largely proposed in literature [6,7] can be used.

The remainder of the paper is organized as follows: In the upcoming section our solution for misbehaving detection will be presented, followed by the ones for accusation approval and isolation in Section 3. Section 4 will be devoted to some analysis and discussions of our solutions, and Section 5 to the simulation study. Related work will be sketched in Section 6, and finally the last section will conclude the paper.

2. Packet Dropper Detection

In this section we present our solution to detect misbehaving nodes that drop packets. We deal both with data and routing control packets, and we present a separate solution for each kind of packets. Because our solution relies on a source routing protocol, it is built with DSR [8]. Note that it could be implemented with any source routing protocol, especially the secure (authentication-enabling) ones [7].

DOI: 10.1002/wcm

2.1. Data Packets

2.1.1. Monitoring

Like the WD, in our solution each node A monitors its successor B in the source route and checks whether this latter forwards to C each packet node A provides such that C is B's successor in the source route and A could be either the source or any intermediate node. This process is repeated on each couple of hops until reaching the final destination. We define a new kind of feedbacks we call two-hop ACK [9], as an ACK that travels two hops. In our context, node C acknowledges packets sent from A by sending this latter via B a two-hop ACK. Node B could, however, escape from the monitor without being detected by simply sending A a *falsified* two-hop ACK. Note that performing in this way is power economic for B, since sending a short packet like an ACK consumes too less energy than a data packet. To get over this vulnerability we use an asymmetric cryptography based strategy as follows:

Node A generates a random number and encrypts it with C's public key (PK), then appends it in the packet's header. When C receives the packet it retrieves the number, decrypts it using its secret key (SK), encrypts it using A's PK, and putting it in a two-hop ACK it sends back to A via B. In the first hop (C,B) the ACK is not transmitted in a separate packet, but piggybacked to the ordinary MAC ACK. This inclusion and mining of the MAC ACK reduces the number of two-hop ACK packets as much as half compared with a separate transmission on each hop. When A receives the ACK it decrypts the random number and checks whether it matches with the one it has generated, in order to validate B's forwarding regarding the appropriate packet. However, if B does not forward the packet A will not receive the two-hop ACK, and it will be able to detect this misbehavior after a timeout. This strategy requires a key distribution mechanisms enabling a security association between each pair of nodes. To ensure this distribution, a mechanism like the chain of trust [10] can be used. Note that the same keys could be employed for other security purposes at the other layers.

The WD's problems presented in the first section are mitigated with this approach, as long as B's forwarding validation at A is not only related to B's transmission, but to C's reception. Nevertheless, the problem with this first solution is that it requires a two-hop ACK for each packet on each couple of hops, which might result in important overhead. To decrease this cost we propose to *randomize* the ACK requests [5], *viz*. A does not ask C an ACK for each packet, but upon sending a packet to forward it *randomly* decides (with a coefficient p) whether it asks an ACK or not, then it conceals this decision in the packet. A simple way to conceal the decision is to exploit the random number. For instance, when the node decides to ask an ACK it selects an even number, and an odd number when it decides not to ask the ACK. This *random* selection strategy prevents the monitored node from deducing which packets contain ACK requests. Note that getting such information allows a misbehaving to drop packets with no requests without being detected. The coefficient p is continuously updated as follows: It is set to 1 (the initial value) when a timeout exceeds without receiving the requested ACK, and decreased to the trust value p_{trust} as soon as the requested ACK is received. Setting p to p_{trust} gives trust to well-behaving nodes, while setting p to 1 enforces the ACK request after a lack of an ACK. This random approach allows to achieve all by the same performance in misbehaving true detections (true positives) like the ordinary two-hop ACK, as we will see later.

2.1.2. Bayesian approach for judgment

The new monitoring method (random two-hop ACK) allows to confirm the correct forwarding of packets. Though, when a monitoring node notices that some packet has been dropped over a link it should not directly accuse the monitored as misbehaving, since this dropping could be caused by collisions or nodes' mobility. Indeed, a threshold of tolerance should be fixed. Hereafter, we propose a Bayesian approach allowing nodes to decide about the behavior of each other. In this approach, well-behaving of nodes improves their reputation, whereas intentional or unintentional packet dropping decreases it. The Bayesian approach [11] is a mathematical estimation method that consists of estimating a parameter the observations of which follow a Bernouli distribution by a Beta distribution. It has the advantage of not needing a memory. That is, only the latest updates are safeguarded, and not all the observations. The Bayesian approach for nodes reputation regarding packet forwarding in MANET has already been used by Buchegger and Le-Boudec [12], but their solution requires periodic transmissions of huge control packets. Since misbehaving is usually exception rather than the norm, information exchange in our solution is limited to negative impressions. It is simpler and engenders no overhead when nodes well-behave.

Each node A thinks that each other node B misbehaves with a probability θ, which is a random

DOI: 10.1002/wcm

variable estimated by a Beta distribution Beta(a, b). Initially with no prior information, θ is assumed uniform in (0,1), which is identical to Beta(1,1). As observations (that follow a Bernoulli distribution with a parameter θ) are made, a and b are updated as follows:

$$a = a + u, \quad b = b + 1 - u$$

where $u = 1$ if the observation consists of a dropping, and 0 otherwise. A dropping in our solution is the lack (non-reception after a timeout) of a required two-hop ACK. If the monitor does not ask a two-hop ACK, then the observation is considered as non-dropping. After as many observations as the decision could be made (θ could be approximated by the mathematical expectation $E(\text{Beta}(a, b))$), B will be judged. This is denoted by the decision (or stationary) point, while the number of observations is expressed by $a + b$. Upon reaching this point, B will be accused as misbehaving as soon as: $E(\text{Beta}(a, b)) > E_{\max}$. Note that:

$$E(\text{Beta}(a, b)) = a/(a + b).$$

$E_{\max}$ could be fixed to 0.5, or for more efficiency it should be estimated empirically for each network as follows:

(1) Make simulations with no misbehaving and compute E at each node for different scenarios that estimate the network.
(2) Retrieve the maximum value in all scenarios from the decision point then consider it as $E_{\max}$.

In mathematical estimation methods, the decision (stationary) point is the one upon which the difference between two subsequent observations could be negligible. One usual choice is that fulfilling the following condition:

$$\text{Var}(\text{Beta}(a, b)) < \epsilon.$$

Such that Var is the mathematical variance, and ϵ is a very small positive. Note that:

$$\text{Var}(\text{Beta}(a, b)) = \frac{a \times b}{(a + b + 1) \times (a + b)^2}$$

However, this choice is inappropriate here, since Var(Beta) is not monotonous with $a + b$. We use the following variance-like function, which is indeed decreasing with a+b:

$$\text{Max}\left(\frac{b}{(a + b) \times (a + b + 1)}, \frac{a}{(a + b) \times (a + b + 1)}\right)$$

In Buchegger's approach [12], every node periodically broadcasts in its neighborhood its view of θ regarding all the other nodes. Nodes use these information (known as second hand information) to update their own opinion on nodes' behavior. To decide about the acceptance of the provided information, each node performs complicated tests on the trustworthiness of the provider. The problem with this proactive solution is that it causes an important overhead, even if nodes well-behave. Our approach is rather reactive, thus no such information are exchanged. Indeed, each node performs monitoring separately and informs the others as soon as a misbehaving is *approved*, as we will see in the next section with more details.

2.2. Control Packets

As for routing control packets we do not use the previous Bayesian-based approach, because it requires many observations before making any decisions. Contrary to data packets, this requirement is not appropriate for control packets, since there are few packets of such kind compared with the first one. Further, dropping control packets like Route REQuests (RREQs) and Route REPly (RREP) should not be tolerated, as it completely excludes selfish nodes from routes. Therefore, we should be more sever in the judgment regarding this kind of packets.

For RREQs packets (which are broadcast), each node monitors a RREQ it forwards or launch as a source. The monitoring starts from the reception of the RREQ (or its launch if the node is the source) and ends after a timeout from its retransmission. For each RREQ, the transmitter monitors all its neighbors. It should either receive (or overhear) the RREQ or a RREP from them, except from which it received the RREQ if the node is not the source. If none of these packets is received from a neighbor B, then the monitor notices a packet dropping for B. When a node observes that another node B drops more than a given *sever* threshold number of packets it judges B as misbehaving.

For the other control packets which are not broadcast but directed to one recipient, namely route error (RERR) and RREP, the two-hop ACK approach is used for monitoring as in the data packets. Nevertheless, we do not use the Bayesian approach and still be as sever as with RREQs in judgment, since dropping these

DOI: 10.1002/wcm

two kinds of packets is also critical. Dropping a RREP prevents a selfish from being included in routes (like dropping RREQ), while dropping a RERR allows a malicious node to launch a DoS attack by preventing the destruction of broken routes. In the next section we illustrate the actions to be taken when some node makes a negative judgment on another, for both data and control packets.

3. Misbehaving Approval and Isolation

Isolating a misbehaving node means:

- do not route packets through it, to avoid losing them
- do not forward packets for it, to punish it

A node A that judges some other node B as misbehaving should not isolate it unilaterally, but must ensure its isolation by all nodes. This is because when A unilaterally isolates B, the others could consider A as misbehaving when they realize that it does not forward packets for B. In social life, a person that accuses another must show proofs. One possible way to prove the accusation is to get witnesses against the accused person.

Identically, we suggest a testimony-based protocol (both for data and control packets) to isolate a detected node. Upon a detection, the detector informs nodes in its neighborhood about the dropper (the accused), and asks for witnesses by broadcasting a Witness REQuest (WREQ) packet. It also puts the detected node ID in a special set we call *suspicious set*. Each node receiving the WREQ investigates the issue as follows:

3.1. Directed Packets

If the packet for which the investigation is launched is a directed packet, that is, sent to one recipient, then this latter immediately sends a *signed* Witness REPly (WREP) packet to the accuser in the following two cases:

- if its suspicious set includes the accused node
- if the accused node's misbehaving expectation is close to $E_{\max}$, or the number of control packets considered dropped is close to the configured maximum threshold

Otherwise, when it has not enough experience with the accused node (B), and if B is its neighbor then it asks the successor of this latter whether it has received packets forwarded from it, by sending an ACcusation REQuest (ACREQ) packet (using a route that does not include B). But first, in order to avoid false accusations, the investigator should ensure that the accuser has really sent a packet to B to be forwarded to the appropriate successor. One possible way to do this is to check whether such a packet has been recently overheard, using the promiscuous mode. The node also should check whether B has sent the accuser an ACK *just after* overhearing the data, to ensure that the former has really received the packet and that the latter is not impressing it (as it will be illustrated later). Note that unlike the WD, the information provided from the promiscuous mode in our solution are not used for the monitoring, but only for witnessing, aiming at improving efficiency of detections. If B's successor has not recently received any packet *forwarded* from B, it sends a *signed* ACcusation REPly (ACREP) packet to the investigator, then this latter testifies for the accusation and sends the accuser a signed WREP packet.

3.2. Broadcast Packets (RREQ)

In this case the node, if it is a neighbor of B, merely checks whether it has recently received (respectively overheard) either any RREQ *forwarded* from this node, or a RREP originated from it. To do this, each node keeps the RREQs and RREPs it receives in a buffer for a short time. If neither RREQ nor RREP have been received then it testifies for the accusation and sends the accuser a signed WREP packet. But it must first ensure that the accuser node has really recently sent out a RREQ, by checking in its buffer. When the detector collects *k* validation from its neighbors, with at least one provided by direct experience (without asking the successor of B), it broadcasts in the network an accusation packet (AC) containing signatures of all the validating nodes. The requirement of at least one direct witness will be argued later. Each node receiving such a valid accusation isolates the guilty. Otherwise, if the detector fails to collect *k* validation then it does not isolate the detected node, but keeps it in the suspicious set.

4. Analysis and Discussions

4.1. Monitoring

Unlike the current detective solutions that are totaly based on the promiscuous mode monitoring (the WD),

DOI: 10.1002/wcm

our solution relies on the new technique of random two-hop ACK [5]. The monitoring node (A) monitors each packet it sends and asks an ACK with a probability p, then it validates the forwarding of the monitored node (B) when it receives an ACK from the successor of this latter (node C). This process can be generalized along the path for each subsequent two hops from the source to the destination, and efficient encryption/decryption operations have been added in order to authenticate the two-hop ACKs and secure the solution against spoofing attacks. Further, our update strategy of p gives more trust to well-behaving nodes and no trust to those that drop one packet. To ensure well-functioning of our solution, the appending of a two-hop ACK to the ordinary ACK upon a reception of a data packet with active ACK requirement should be an inviolable operation. If B correctly forwards packets but C does not send back two-hop ACKs, A will be unable to validate B's forwarding. This non-violation can be ensured by implementing the operation with a tamper-resistant hardware module. However, all the other operations, including the two-hop ACK forwarding, do not need such a requirement. Whatever the motivation of B, it is of no interest for it not to forward a short two-hop ACK after forwarding a packet, since this would cause its isolation.

Getting rid of the promiscuous-mode-based monitoring makes our solution independent of the transmission power, and resolves the WD false detection problem related to the employment of the power-control technique. Moreover, our solution resolves some WD's problems related to collisions. When a collision appears at C, B should retransmit the packet, otherwise A will not validate its forwarding. This because B's forwarding will not be validated at A until C really receives the packet and sends back the two-hop ACK, unlike the WD where the validation is only related to the first transmission of B. Remember that in the WD when A is closer to B than C, then B could save its energy and make the transmission power strong enough to be overheard by A but less than the one required to reach C. This problem is also eliminated in our solution.

As illustrated, the authentication of the two-hop ACK packet is ensured by employing encreption/decreption operations on a random number, generated by the monitor node and piggybacked to the monitored packet. These operations have minor impact, since they are applied merely on the random number and not on the whole packet holding it. Note that we avoided the use of digital signatures in order to avoid useless packet hash computation. The randomization of two-hop ACK reduces the communication overhead, while keeping the efficiency in detection good enough. In the following we first analyze this efficiency, then we will discuss the overhead. We assume in this mathematical analysis that channels are always reliable such that there is no packet loss. In the next section, we will investigate by simulation the detection efficiency in scenarios with mobility and collisions causing packet losses.

The behavior of the monitored node for each packet follows a Bernoulli distribution with a parameter θ (the probability of dropping as illustrated before). Monitoring n packets could be considered as the repetition of the previous operation (monitoring one packet) n times. Therefore, the number of packets dropped (pdr) for n packets is a random variable that is the sum of n random variables following a Bernoulli distribution with parameter θ (mathematical expectation), thus it follows a Binomial distribution with a mathematical expectation: $E(\text{pdr}) = \theta \times n$.

Theoretically, when the previous assumption of reliability is held, the ordinary two-hop ACK detects all these packets. Hereafter, our purpose is to assess the number of packets detected (pd) by the random two-hop ACK, that is, $E(\text{pd})$. The requesting coefficient of the random two-hop ACK algorithm is continuously updated. It differs from one operation (monitoring one packet) to another according to the result of the previous operation, and the previous behavior as well. We denote the value of the coefficient P set by *algorithm* upon monitoring a packet i (which is a random variable) by P_i. In the execution, the probability of asking an ACK for the packet i would be expressed by $E(P_i)$. P_i is fixed to 1 if in the previous operation the packet was dropped and detected, thus with the probability[‡] $\theta E(P_{i-1})$, otherwise it is fixed to P_{trust}, that is, with the probability $1 - \theta E(P_{i-1})$. Therefore, the mathematical expectation of P_i $(E(P_i))$ could be expressed by: $1 \times \theta E(P_{i-1}) + P_{\text{trust}} \times (1 - \theta E(P_{i-1}))$. Hence:

$$E(P_i) = P_{\text{trust}} + \theta(1 - P_{\text{trust}})E(P_{i-1}) \qquad (1)$$

The number of packets detected by the random strategy (pd) also follows a Binomial distribution, since it is the results of repeating a Bernoulli operation n

[‡] The probability of detecting the ith packet when dropped is the probability of asking an ACK in the $(i-1)$th operation. The events dropping the ith*packet* and requesting ACK for the $(i-1)$th packet are independent.

DOI: 10.1002/wcm

times with parameter θP_i. However, the only difference from the continuous requesting is that in this latter strategy P_i is not constant. We have:

$$E(\text{pd}) = \sum_{i=1}^{n} E(\theta P_i) = \theta \sum_{i=1}^{n} E(P_i) \qquad (2)$$

Note that $P_1 = 1$

Lemma 1: $\forall i \geq 1$,

$$E(P_i) = \theta^{i-1}(1 - P_{\text{trust}})^i + P_{\text{trust}} \sum_{j=0}^{i-1} \theta^j (1 - P_{\text{trust}})^j$$

Using this Lemma, formula (2) could be evolved into:

$$\begin{aligned} E(\text{pd}) = {} & \frac{\theta P_{\text{trust}}}{1 - \theta(1 - P_{\text{trust}})} n + \theta(1 - P_{\text{trust}}) \\ & \times \frac{1 - \theta^n (1 - P_{\text{trust}})^n}{1 - \theta(1 - P_{\text{trust}})} \\ & \times \left(1 - \frac{\theta P_{\text{trust}}}{1 - \theta(1 - P_{\text{trust}})}\right) \qquad (3) \end{aligned}$$

The steps of simplification and the proof of Lemma 1 are presented in the Appendix.

This probability depends on many parameters, we try to investigate it versus some usual values of P_{trust}. For $P_{\text{trust}} = 1/4$, $E(\text{pd}) \approx \frac{\theta}{4-3\theta} n$, for $P_{\text{trust}} = 1/2$, $E(\text{pd}) \approx \frac{\theta}{2-\theta} n$, and finally for $P_{\text{trust}} = 3/4$: $E(\text{pd}) \approx \frac{3\theta}{4-\theta} n$ Figure 1 illustrates the approximated detection ratio according to θ. We mean by detection ratio $E(\text{pd})/E(\text{pdr})$.

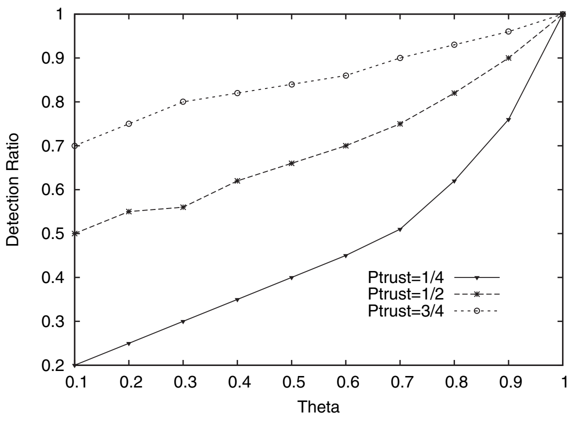

Fig. 1. Detection ratio.

The overhead (for both the ordinary and the random two-hop ACK versions) depends on the nodes' behavior. In the following we first compute the communication complexity of the two versions, then we illustrate the reduction factor (RF) provided by the random approach optimization, that is, the overhead of the ordinary two-hop ACK divided by the overhead of the random version. We consider monitoring n packets on an h hops route, where each node misbehaves with a probability θ (as assumed).

In the first solution p is constant and fixed to 1, thus the number of two-hop ACKs requested on the ith hop is simply the number of data packets arrived at this hop, we denote by Na_i. In this case, the total number of two-hop ACKs requested Np_0 is:

$$Np_0 = \sum_{i=0}^{h-2} Na_i, \quad \text{thus}: E(Np_0) = \sum_{i=0}^{h-2} E(Na_i).$$

We will use this latter formula as both Np_0 and Na_i are random variables.

On the first hop $E(Na) = n$, since the source transmits all its packets. On the second one it decreases by the number of packets dropped by the first forwarder, that is it decreases by $n\theta$ and becomes $n(1-\theta)$. On the third one this latter decreases by $n(1-\theta)\theta$ and becomes $n(1-\theta)^2$, and so on until the last hop involving a monitoring $(h-2, h-1)$, on which the number of ACK requests is $n(1-\theta)^{h-2}$. As a result:

$$E(Np_0) = \sum_{i=0}^{h-2} n(1-\theta)^i = n \times \sum_{i=0}^{h-2} (1-\theta)^i$$

As the last sum is a finite geometric series (when $\theta \neq 0$) of $h-1$ terms, with a first term equals to 1 and a ratio of $\theta(1-\theta)$, we get:

$$E(Np_0) = \begin{cases} O\left(\frac{n}{\theta}\left(1 - (1-\theta)^{h-1}\right)\right) & \text{when } \theta > 0 \\ O\left((h-1) \times n\right) & \text{when } \theta = 0 \end{cases} \qquad (4)$$

However, in the second solution P differers from a packet to another. On each hop i, the number of two-hop ACKs requested can be expressed by: $\sum_{j=1}^{Na_i} P_j$. In other words, this number (of two-hop ACKs requested in this case) is a random variable representing the sum of Na_i other random variables (P_j, $j = 1 \ldots Na_i$). Let the total number of two-hop ACKs requested in this solution be denoted by Np, its mathematical

DOI: 10.1002/wcm

expectation is expressed by:

$$E(Np) = \sum_{i=1}^{n} E(P_{1,i}) + \sum_{i=1}^{n(1-\theta)} E(P_{2,i}) + \cdots + \sum_{i=1}^{n(1-\theta)^{h-2}} E(P_{h-1,i})$$

Such that $P_{k,j}$ is the coefficient's value set by node k for the packet j. If we assume for simplicity that all the nodes generate the same sequence of random numbers, that is, $P_{k,j}$ only depends on j and not k, then $P_{k,j}$ can be simply written P_j, and:

$$E(Np) = \sum_{i=0}^{h-1} \sum_{j=1}^{n(1-\theta)^i} E(P_j) \qquad (5)$$

This can be approximated by:

$$E(Np) \approx \begin{cases} O\left(\frac{P_{\text{trust}}}{1-\theta(1-P_{\text{trust}})} \frac{n}{\theta}\left(1-(1-\theta)^{h-1}\right)\right) & \text{when } \theta > 0 \\ O((h-1) \times n \times P_{\text{trust}}) & \text{when } \theta = 0 \end{cases} \qquad (6)$$

The steps of deducing formula (6) from (5) are illustrated in the Appendix.

From formulas (6) and (4) we conclude the generale term of the communication overhead RF:

$$\text{RF} = E(Np_0)/E(Np) = \frac{1-\theta(1-P_{\text{trust}})}{P_{\text{trust}}},$$

which is $1/P_{\text{trust}}$ in the particular case when when $\theta = 0$

In the following we discuss the factor for some usual values of P_{trust}: when $P_{\text{trust}} = 1/4$ $RF \approx 4 - 3\theta$, when $P_{\text{trust}} = 1/2$ $RF \approx 2 - \theta$, and finally, when $P_{\text{trust}} = 3/4$ $RF \approx \frac{4-\theta}{3}$

Figure 2 shows this factor versus θ for the previous values of P_{trust}.

From Figures 1 and 2 we realize that $P_{\text{trust}} = 0.5$ strikes a balance between efficiency (detection ratio) and cost (RF). It decreases the complexity overhead as much as half (when nodes well-behave), while keeping the detection ratio good enough (always ≥0.5). Contrary to $P_{\text{trust}} = 0.25$ that has too low values of detection ratio for low and average misbehaving, and to $P_{\text{trust}} = 0.75$ that has too low values of RF. Thus, we fix P_{trust} to 0.5 later in our simulation study.

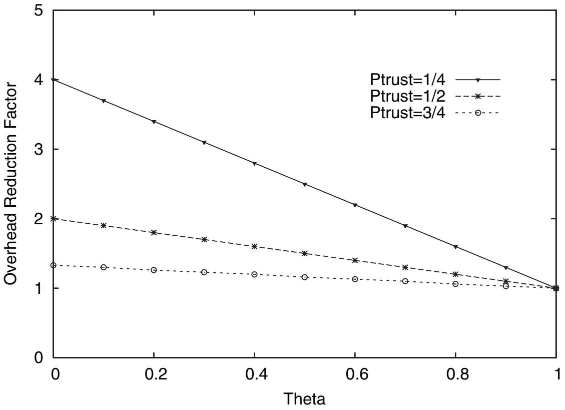

Fig. 2. Overhead reduction factor.

The previous monitoring strategy has been proposed for directed packets, which include data packets as well as RREP and RRER packets. However, it could not be applied for broadcast packets like RREQ, for which there is not a specified recipient. For this reason we proposed another strategy to monitor RREQ, where each node that sends a RREQ monitors its neighbors and validates one's forwarding either when it receives a RREP or a RREQ from it. This solution requires no communication overhead for monitoring, but needs nodes to be neighborhood-aware. Neighborhood-awareness can be achieved by employing beacons either at the MAC or at the routing protocol.

4.2. Judgment

Because a packet dropping might be unintentional due to nodes mobility and channel conditions, accusation should not be made immediately upon one dropping detection. Indeed, more observations must be noted before making a judgment. We have proposed a Bayesian approach to make such a judgment, where each node estimates each other's misbehavior with a probability that follows a Beta(a, b) distribution, whose parameters (a, b) are updated as observations are made. When enough observations with regard to a given monitored node are collected such that the judgment point is reached, the monitoring node will accuse the monitored one as soon as the estimated probability $(E(\text{Beta}(a, b)))$ exceeds the configured maximum tolerance threshold, that is, $E(\text{Beta}(a, b)) > E_{\max}$.

$$E(\text{Beta}(a, b)) > E_{\max} \longleftrightarrow \frac{a}{a+b} > E_{\max} \longleftrightarrow a > \frac{b \times E_{\max}}{1 - E_{\max}}$$

DOI: 10.1002/wcm

This latter $(b \times E_{\max})/(1 - E_{\max})$ represents the tolerable number of packets a node is allowed to drop without being accused. This maximum tolerable threshold is proportional to b, the number of packets forwarded. The more a node forward packets, the more its tolerable threshold increases. Forwarding packets after unintentional or intentional droppings that do not result in an accusation would decrease E, which allows redemption. This redemption could not be possible when setting the tolerable threshold to a fixed number of packets. Note that the strategy of dropping upto the tolerable threshold is not efficient and safe for a misbehaving, since it cannot know whether and how much the monitor will notice false observations because of channel conditions or nodes' mobility.

As for control packets, misbehaving on their forwarding is more crucial. Dropping RREQ and RREP is motivating in the context of selfish nodes. It enables a selfish node to exclude itself from routes such as to get no data packet to forward. Further, the number of control packets is generally minor compared with data packets. All these render the previous Bayesian approach ineffective with this kind of packets. Indeed, we have proposed to use a fixed number of packets threshold. The strategy of dropping upto the tolerable threshold is inefficient for the same reason cited before (of data packets).

4.3. Witness-Based Approval

To mitigate false detections and rumors vulnerability, we have proposed in the previous section a witness-based protocol. In this protocol, a node that detects and accuses another as misbehaving must prove its accusation before taking any measure against it. It should not isolate the assumed misbehaving unilaterally, because this can result in false detections against it. Upon the detection of a misbehaving, the detector launches locally in its neighborhood a call for witnesses, using a broadcast control packet that cost only one transmission. Neighbors that consider the accused node as suspicious, or those that are monitoring it and whose misbehaving estimations are close to the tolerable threshold (respectively which did not receive a RREQ if the accusation is for RREQ) testify against it by sending the requestor a signed reply packet. Those which have not enough experience with the accused node investigate this accusation and ask the successor of this latter whether it has recently received packets from it. But first, they ensure that the accuser node really sent the packet to the accused one to forward to the claimed successor. To do this they must be neighbors of the accused, otherwise they do not testify. The following example illustrates and analyzes the investigation: Assume three aligned nodes, A, B, C, and another node D in A's range as shown in Figure 3. When A accuses B not forwarding packets to C and sends a call for witnesses, D investigates the issue. But before asking C it ensures that A has really sent the packet and B has received it, by checking the data packets and ACKs overheard. This is because D could not ensure that B has received the data packet by merely overhearing it. For instance, if D is closer to A than B, A (attempting a DoS attack against B) could send the packet with a power strong enough to be overhead by D, but not by B. Requiring the ACK[§] reception from B *just after* the data ensures that B has really received the data from A. To do this, D simply safeguards the overheard packets (their headers) during a short period. This way, a node that asks the successor of the accused node has no doubt that this latter has received a data packet to forward to the successor in question. Any collision at D prevents it from testifying, but has no effect on false detections.

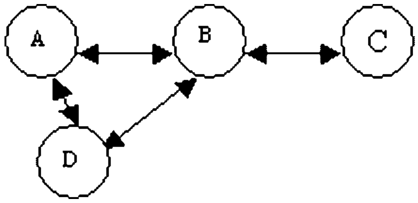

Fig. 3. Example of a nodes' connections.

Upon the reception of the ACREQ, the asked node (C) replies with a signed ACREP packet if it has not received any packet from B. A coincidental collision at C at that moment, however, would result in a false reply if A is attempting a DoS attack, then in a false testimony. Nevertheless, the requirement of at least one direct witness (testifying from its direct experience) mitigates wrong accusation caused by this kind of false testimonies. The signature of the packets prevents their spoofing, thus no node could testify using the ID of another.

The accuser node has to collect k different signatures to approve its accusation. Theoretically, $k - 1$ is the maximum number of misbehaving nodes that could exist at any time. In practice, however, it is hard to determine such a number, so it should be fixed to strike a balance between efficiency and robustness. Setting k to a high value increases the robustness of

[§] The source of this ACK should be authenticated at the MAC layer.

DOI: 10.1002/wcm

the protocol against false detections and rumors, but decreases its efficiency regarding true detections. On the other hand, a low value of *k* allows high detections, but opens the vulnerability of rumors and increases the unintentional false detections (false positives), since *k* nodes could collude to accuse maliciously (respectively wrongly) any node. This issue related to *k* will be investigated in the next section. Once the accuser collects *k* valid signatures, it broadcasts an accusation packet including all signatures through the network to isolate the guilty. This broadcast is costly, but it is not performed until a node is detected and approved as misbehaving. Except for the monitoring, our solution requires no overhead as long as nodes well-behave, as no opinions are exchanged periodically. This makes our solution reactive, unlike the other reputation-based solutions that will be presented later.

5. Simulation Results

5.1. Monitoring

To assess the performance of the proposed monitoring protocol, we have driven a GloMoSim-based [13] simulation study that we present hereafter. We have simulated a network of 50 nodes, located in an area of $1500 \times 1000\,\text{m}^2$ where they move following the random way-point model with an average speed 1 m/s, for 900 s of simulation time. To generate traffic we have used three CBR sessions between three pairs of remote nodes, each consists of continually sending a 512 bytes data packet each second. On each hop, each data packet is transmitted using a controlled power according to the distance between the transmitter and the receiver.

In this subsection we compare two versions of our monitoring protocol, 2HopACK and Random 2HopACK, as well as the WD with regard to the misbehaving detection rate, the false detection rate (rate of false accusations as misbehaving), and the number of two-hop ACKs (which represents the overhead). We measured these metrics versus the misbehaving nodes rate, which represents the rate of nodes that misbehave and drop data packets they are asked to relay. We investigate data packets dropping, and keep the investigation into control packets for our perspectives. Though, our protocol is more sever for control packets, and it makes misbehaving nodes more vigilant for this kind of packets. Each point of the plots presented hereafter has been obtained by averaging five measurements with different seeds. Note that we implemented our protocol with DSR for this simulation, like WD. Also note that WD requires no kind of ACK, so the last metric (number of two-hop ACK) concerns solely our protocol's versions. The Bayesian approach has not been used in this step since the WD does not use it. We simply used (for all protocols) a tolerance threshold of packets number, fixed empirically to 100 packets.

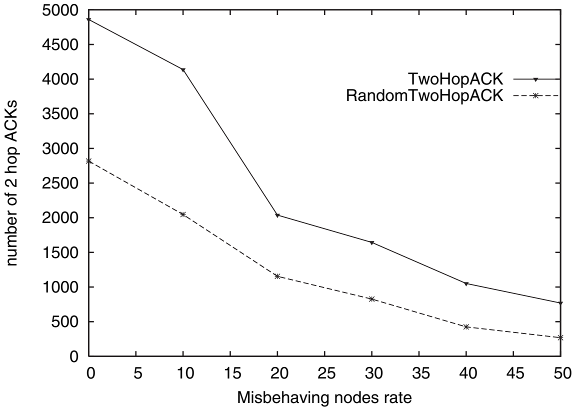

Fig. 4. Number of two-hop ACK versus misbehaving rate.

The first version of our protocol requires an ACK for each packet, while the second one uses the efficient technique of randomizing the ACK requests that reduces the overhead, especially when the misbehaving nodes rate is low as shown in Figure 4. The decrease of the packets number versus the misbehaving nodes increase in this figure for both protocols can be argued by the fact that the misbehaving increase causes more and more packets dropping during their routing, then decreases the number of packets to be monitored. The cost of this overhead decrease is a minor loss in detection efficiency, as shown in Figure 5. But we can clearly see in the same figure that both versions have better detection than WD. Figure 6 illustrates how our protocol (the two versions) decreases hugely the false detection rate compared with WD. We should

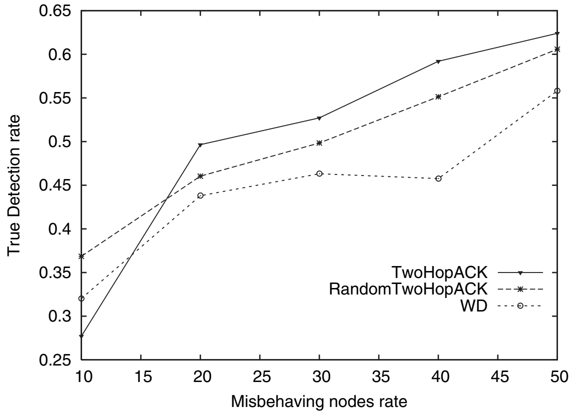

Fig. 5. True detection versus misbehaving rate.

DOI: 10.1002/wcm

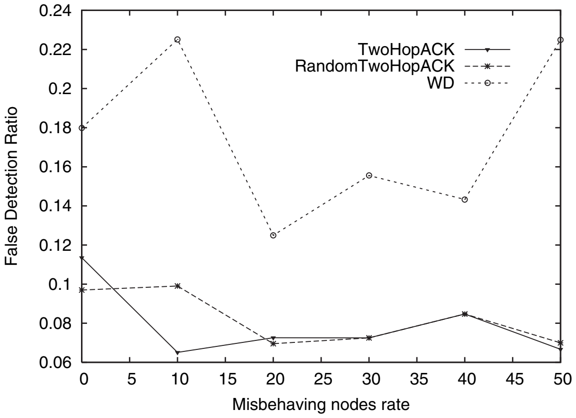

Fig. 6. False detection versus misbehaving rate.

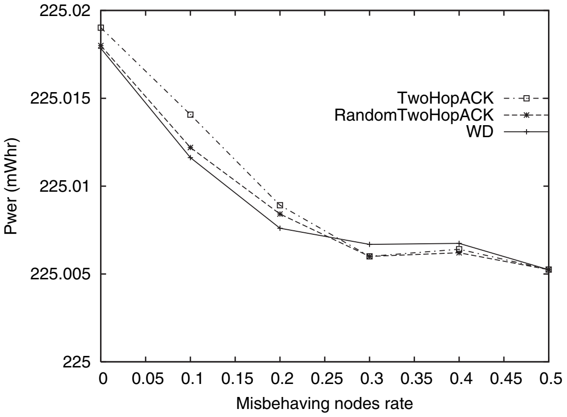

Fig. 7. Power consumption versus misbehaving rate.

point out that this metric (false detection) and the misbehaving rate are not monotonously dependent, since this latter decreases the load, which reduces both the number of nodes wrongly accused and the number of well-behaving nodes monitored (the numerator and denumerator of the false detection rate). Finally, the small difference (before the ratio 30%) between our protocol and WD regarding the power consumption represented in Figure 7 is basically due to the overhead. We can also see on the same figure that the random version reduces a little bit the power consumption. Note that when the misbehaving rate is higher than 30% we remarked a data reception ratio slightly below 100% for the two versions of our protocol (due to collisions caused by two-hop ACK), and kept 100% for the WD.[‖] Since we used CBR above UDP, these packets are lost and not retransmitted. Therefore, we argue the difference between our protocol and WD when the misbehaving ratio is above 30% by the fact that these packets are relayed until the destination in WD resulting in more consumption. Overall, our protocol clearly outperforms the WD, with a minor cost in energy consumption. However, regarding the overhead the first version of our protocol has a considerable cost. The random version decreases hugely this cost while keeping almost the same efficiency on detection, thus we use it as the monitoring protocol in the next subsection.

5.2. Isolation

Hereafter we asses the witnesses-based isolation method, and we study the impact of the parameter k (number of witnesses required) on our solution. To allow nodes achieving decision points we have raised the simulation time to 1500 s, and the number of CBR sessions to 23 as well. Still, all the other parameters have been maintained as described in the previous subsection. We compare two versions, the first with one witness required for the isolation and the second with two, respectively denoted by one-witness and two-witness. Each of which uses the random two-hop ACK for monitoring with p_{trust} fixed to 0.5 (as argued previously), and the Bayesian approach for accusation with $E_{\max}$ fixed to 0.5 and ϵ to 0.05. The comparison is performed regarding true and false isolation rates, that is, the rate of the nodes correctly (respectively wrongly) isolated, in different misbehaving rates. Note that misbehaving nodes and CBR sessions have been fixed in such a way to judge (correctly or wrongly) all the monitored nodes before the end of the simulation. That means for each monitored node at least one of the nodes monitoring it reaches the decision point, but no matter whether it judges it correctly or not.

As illustrated in Figures 8 and 9 two-witness considerably improves (decreases) the false positive

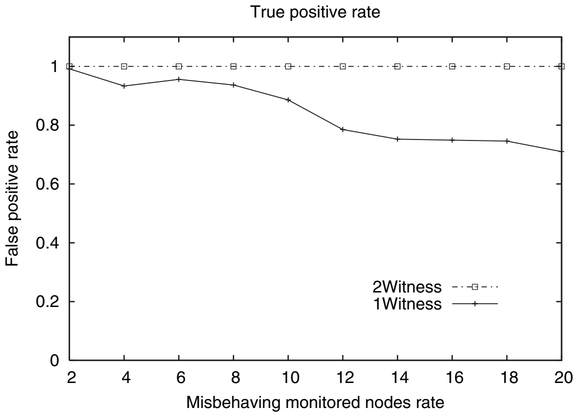

Fig. 8. True detection versus misbehaving rate.

[‖] We remarked that the reception ratio for misbehaving ratio below 30% was 100% for both versions.

DOI: 10.1002/wcm

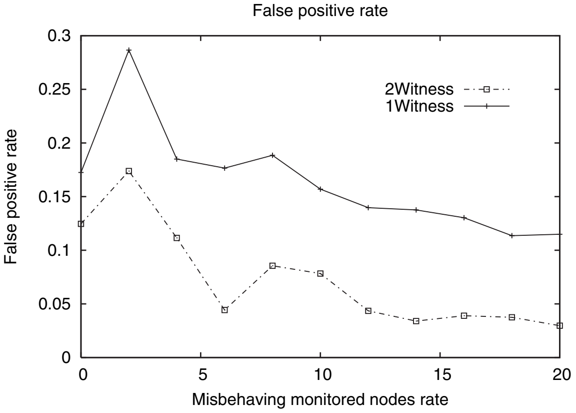

Fig. 9. False detection versus misbehaving rate.

rate, but loses a little bit on the true positive rate compared with one-witness, especially when the misbehaving rate exceeds 10%. False detections in our scenarios are due to nodes' mobility and packets' collisions that increase unintentional dropping, thus the likelihood of false accusations. The one-witness version has unacceptable values with respect of this metric, particulary when the misbehaving rate is low. Two-witness mitigates this shortcoming, and also cut down the vulnerability of collusive false accusation attack compared with one-witness, since more than two nodes have to collude to isolate a node. We point out that false detections for high misbehaving rates are low because more nodes misbehave more packets are likely to be dropped earlier through their route, which reduces the number of observations, thus the number of droppers reaching the threshold. The parameter *k* could be increased to make the solution less tolerant on false detection and false accusation attacks, but should depend on nodes' connectivity to not lose efficiency on detections. In networks with low connectivity, it should not be increased lots. Because this would prevent nodes from finding witnesses, and consequently decreases the detection efficiency. In our scenarios, we remarked that fixing *k* to 3 was not efficient at all.

6. Related Work

Marti *et al.* [2] are the first who dealt with the problem of nodes misbehavior on packet forwarding, by proposing the WD that they implemented with DSR [8]. As illustrated before, the WD relies on monitoring neighbors in the promiscuous mode, and suffers from some problems, especially when using the power control technique, employed by some new power-aware routing protocols following the WD's proposal [4,3,14].

Yang *et al.* [15] describe a unified network layer solution to protect both routing and data forwarding in the context of AODV. Michiardi and Molva [16] suggest a generic reputation-based mechanism, namely CORE, supposed to be easily integrated with any network function. CONFIDANT is another interesting reputation-based solution, proposed by Buchegger and Le Boudec [17]. In Reference [12], these authors fortify CONFIDANT with a modified Bayesian approach for its reputation system. All these solutions, however, rely on the WD technique in their monitor component. Moreover, the reputation-based solutions require periodic exchange of reputation information, which is costly and unworthy so long as nodes well-behave. Balakrishnan *et al.* [18] propose a network layer monitoring approach, that is very similar to our ordinary two-hop ACK [9]. However, in this solution the authors reduce the number of two-hop ACKs by suggesting to gather up the acknowledgment of multiple data packets in one ACK (the so called S-TWOACK), independently of the behavior of the monitored node, which differs from our random two-hop ACK [5]. Also, note that this solution is limited to the monitoring component, and does not provide any punishment strategy. More importantly, the two-hop ACK in this solution are not authentic, and the authors do not deal with the possible falsification of the two-hop ACK by the monitored node (see the second section).

Buttyan and Hubaux [19] propose an efficient preventive economic-based approach stimulating nodes to cooperate, which is modeled and analyzed in Reference [20]. The authors introduce what they called *virtual currency* or *nuglets*, along with mechanisms for charging/rewarding service usage/provision. The main idea of this technique is that nodes which use a service must pay for it (in nuglets) to nodes that provide it. Some other stimulating preventive approaches are based on game theory such as [21–23]. These preventive solutions motivate nodes to cooperate, but do not aim at detecting the misbehaving nodes, contrary to the previous ones. In Reference [24], Papadimitratos and Haas present the SMTP protocol. It is a hybrid solution that mitigates the misbehavior effects (packets loss) by dispersing packets, and detects the misbehavior by employing end-to-end feedbacks. Reference [25] is another interesting example that employs such feedbacks, and exploits the TCP ACKs through a cross-layer design to reduce the overhead. This kind of feedbacks, however, allows the detection of routes containing misbehaving nodes, but fails to

DOI: 10.1002/wcm

detect them. To overcome this problem Kargl *et al.* [26] propose *iterative probing*, enabling the detection of links containing misbehaving nodes. Nonetheless, this fortified solution fails to detect the appropriate nodes. To find the appropriate node on a link after an iterative probing the authors propose the so called *unambiguous probing*, which is based on the WD, thus suffers from its problems.

7. Conclusion

In this paper we have proposed a comprehensive solution to monitor, detect, and isolate misbehaving nodes that do not forward packets in MANETs. For monitoring, we employed the efficient technique of two-hop ACK, whose cost was reduced by the random requesting strategy. Analysis and simulation results show that the random two-hop ACK is all but as efficient as the ordinary two-hop ACK in high true and low false detection, while hugely reducing the overhead. For local judgment we have proposed a Bayesian approach, that allows redemption before making decisions, and decreases false accusations due to channel conditions and nodes' mobility. However, we have been less tolerant with control packets whose dropping is crucial. Compared with the Buchegger's Bayesian trust manager, our solution does not use any periodic packets exchange, thus it requires no overhead as long as nodes well-behave. Once a node is judged locally as misbehaving by some other node, this latter must prove its detection to ensure the isolation of the misbehaving node by all nodes. For this end we proposed a witness-based protocol, that enforces the detector to collect at least k witnesses before isolating the detected node. Fixing k is a trade-off problem. High values mitigates rumors aiming DoS attacks, as well false detections (especially for control packets with which we have been more sever), but reduces the efficiency on detections, contrary to low values. In our simulation, the protocol with two witnesses showed considerable improvement regarding false accusation while keeping the true detection good enough. This parameter could be risen to ensure more robustness, but should depend on the connectivity to keep efficiency. For instance, fixing k to 3 is not efficient in our scenarios at all, as it reduces dramatically the true positives.

In this work, we attempted to propose a new solution to struggle some problems of the current solutions. Still, the problem of efficiently and optimally detect and isolate misbehaving packet droppers in MANET is far from being resolved, and still represents an open research topic. As a perspective, we plane to make more investigations into misbehaving on control packets.

Acknowledgments

The authors would to thank their colleague Redouane Boudjemaa, as well as the anonymous referees for their reviews, comments, and suggestions that have considerably helped them to improving this paper. They would also like to thank the associated editor, Dr Mohammed Younice, for handling the paper, and for encouraging them revising earlier versions of the manuscript and resubmitting it again to WCMC.

Appendix

Proof of Lemma 1

We prove this Lemma by recurrence on i. For $i = 1$; we simply replace i by 1 in the formula, and we get $E(P_1) = 1$, which is correct since $P_1 = 1$. Now assume the formula is held for $i - 1$, then we will prove it for i. Hence by assumption:

$$E(P_{i-1}) = \theta^{i-2}(1 - P_{\text{trust}})^{i-1} + P_{\text{trust}} \sum_{j=0}^{i-2} \theta^j (1 - P_{\text{trust}})^j$$

By replacing $E(P_{i-1})$ by this expression in Equation (1) we obtain:

$$\begin{aligned} E(P_i) &= P_{\text{trust}} + \theta(1 - P_{\text{trust}}) \times \left(\theta^{i-2}(1 - P_{\text{trust}})^{i-1} + P_{\text{trust}} \sum_{j=0}^{i-2} \theta^j (1 - P_{\text{trust}})^j\right) \\ &= P_{\text{trust}} + \theta^{i-1}(1 - P_{\text{trust}})^i + P_{\text{trust}} \sum_{j=1}^{i-1} \theta^j (1 - P_{\text{trust}})^j \\ &= \theta^{i-1}(1 - P_{\text{trust}})^i + P_{\text{trust}} \left(1 + \sum_{j=1}^{i-1} \theta^j (1 - P_{\text{trust}})^j\right). \end{aligned}$$

Since $\theta^0(1 - P_{\text{trust}})^0 = 1$, we conclude:

$$E(P_i) = \theta^{i-1}(1 - P_{\text{trust}})^i + P_{\text{trust}} \sum_{j=0}^{i-1} \theta^j (1 - P_{\text{trust}})^j$$

□

DOI: 10.1002/wcm

Derivation of Equation (3):

Using Lemma 1, $E(\text{pd})$ can be rewritten into:

$$\begin{aligned} E(\text{pd}) &= \theta \sum_{i=1}^{n} \Bigg[\theta^{i-1}(1-P_{\text{trust}})^{i} \\ &\quad + \sum_{j=0}^{i-1} \theta^{j} (P_{\text{trust}} 1 - P_{\text{trust}})^{j}) \Bigg] \end{aligned}$$

$$\begin{aligned} E(\text{pd}) &= \theta P_{\text{trust}} \sum_{i=1}^{n} \sum_{j=0}^{i-1} \theta^{j}(1-P_{\text{trust}})^{j} \\ &\quad + \sum_{i=1}^{n} \theta^{i}(1-P_{\text{trust}})^{i} \end{aligned}$$

The first sum $\left(\sum_{j=0}^{i-1} \theta^{j}(1-P_{\text{trust}})^{j}\right)$ is just a finite geometric series of i terms, its first term is 1, and its ratio is $\theta(1-P_{\text{trust}})$. The second is also a geometric series of n terms, both its first term and ratio are $\theta(1-P_{\text{trust}})$. Thus:

$$\begin{aligned} E(pd) &= \theta P_{\text{trust}} \sum_{i=1}^{n} \frac{1-\theta^{i}(1-P_{\text{trust}})^{i}}{1-\theta(1-P_{\text{trust}})} + \theta(1-P_{\text{trust}}) \\ &\quad \times \frac{1-\theta^{n}(1-P_{\text{trust}})^{n}}{1-\theta(1-P_{\text{trust}})} \\ &= \frac{\theta P_{\text{trust}}}{1-\theta(1-P_{\text{trust}})} \times \left(n - \sum_{i=1}^{n} \theta^{i}(1-P_{\text{trust}})^{i} \right) \\ &\quad + \theta(1-P_{\text{trust}}) \frac{1-\theta^{n}(1-P_{\text{trust}})^{n}}{1-\theta(1-P_{\text{trust}})} \end{aligned}$$

The previous sum is a finite geometric series too, its first term is $\theta(1-P_{\text{trust}})$, its ratio is $\theta(1-P_{\text{trust}})$, and it includes n terms, so:

$$\begin{aligned} E(\text{pd}) &= \frac{\theta P_{\text{trust}}}{1-\theta(1-P_{\text{trust}})} \times \Bigg(n - \theta(1-P_{\text{trust}}) \\ &\quad \times \frac{1-\theta^{n}(1-P_{\text{trust}})^{n}}{1-\theta(1-P_{\text{trust}})} \Bigg) \\ &\quad + \theta(1-P_{\text{trust}}) \frac{1-\theta^{n}(1-P_{\text{trust}})^{n}}{1-\theta(1-P_{\text{trust}})} \end{aligned}$$

$$\begin{aligned} E(\text{pd}) &= \frac{\theta P_{\text{trust}}}{1-\theta(1-P_{\text{trust}})} \times n + \theta(1-P_{\text{trust}}) \\ &\quad \times \frac{1-\theta^{n}(1-P_{\text{trust}})^{n}}{1-\theta(1-P_{\text{trust}})} \\ &\quad \times \left(1 - \frac{\theta P_{\text{trust}}}{1-\theta(1-P_{\text{trust}})} \right) \end{aligned}$$

□

Derivation of Equation (6):

In the following we assume $\theta < 1$. Note that when $\theta = 1$, it can be easily deduced from Equation (5) that $E(Np) = n$, which fulfils formula (6)

Using Lemma 1 we get:

$$\begin{aligned} E(Np) &= \sum_{i=0}^{h-1} \sum_{j=1}^{n(1-\theta)^{i}} \Bigg(\theta^{j-1}(1-P_{\text{trust}})^{j} \\ &\quad + P_{\text{trust}} \sum_{k=0}^{j-1} \theta^{k}(1-P_{\text{trust}})^{k} \Bigg) \\ &= (1/\theta) \sum_{i=0}^{h-1} \sum_{j=1}^{n(1-\theta)^{i}} \theta^{j}(1-P_{\text{trust}})^{j} + P_{\text{trust}} \\ &\quad \times \sum_{i=0}^{h-1} \sum_{j=1}^{n(1-\theta)^{i}} \sum_{k=0}^{j-1} \theta^{k}(1-P_{\text{trust}})^{k} \\ &= \sum_{i=0}^{h-2} (1-P_{\text{trust}}) \frac{1-\theta^{n(1-\theta)^{i}}(1-P_{\text{trust}})^{n(1-\theta)^{i}}}{1-\theta(1-P_{\text{trust}})} \\ &\quad + P_{\text{trust}} \times \sum_{i=0}^{h-2} \sum_{j=1}^{n(1-\theta)^{i}} \frac{1-\theta^{j}(1-P_{\text{trust}})^{j}}{1-\theta(1-P_{\text{trust}})} \\ &= \sum_{i=0}^{h-2} (1-P_{\text{trust}}) \frac{1-\theta^{n(1-\theta)^{i}}(1-P_{\text{trust}})^{n(1-\theta)^{i}}}{1-\theta(1-P_{\text{trust}})} \\ &\quad + P_{\text{trust}} \times \sum_{i=0}^{h-2} \frac{n(1-\theta)^{i}}{1-\theta(1-P_{\text{trust}})} \\ &\quad + \frac{\theta P_{\text{trust}}}{1-\theta(1-P_{\text{trust}})} \times \sum_{i=0}^{h-2} (1-P_{\text{trust}}) \\ &\quad \times \frac{1-\theta^{n(1-\theta)^{i}}(1-P_{\text{trust}})^{n(1-\theta)^{i}}}{1-\theta(1-P_{\text{trust}})} \\ &= \frac{P_{\text{trust}}}{1-\theta(1-P_{\text{trust}})} \sum_{i=0}^{h-2} n(1-\theta)^{i} \\ &\quad + \left(1 + \frac{\theta P_{\text{trust}}}{1-\theta(1-P_{\text{trust}})} \right) \\ &\quad \times \left(\frac{1-P_{\text{trust}}}{1-\theta(1-P_{\text{trust}})} \right) \\ &\quad \times \left(h - 1 - \sum_{i=0}^{h-2} (\theta(1-P_{\text{trust}}))^{n(1-\theta)^{i}} \right) \end{aligned}$$

DOI: 10.1002/wcm

$$E(Np) = \begin{cases} \frac{P_{\text{trust}}}{1-\theta(1-P_{\text{trust}})}\frac{n}{\theta}(1-(1-\theta)^{h-1}) \\ \quad + \left(1+\frac{\theta P_{\text{trust}}}{1-\theta(1-P_{\text{trust}})}\right) \\ \left(\frac{1-P_{\text{trust}}}{1-\theta(1-P_{\text{trust}})}\right)\left(h-1-\sum_{i=0}^{h-2}\right. \\ \quad \left.\times\,(\theta(1-P_{\text{trust}}))^{n(1-\theta)^i}\right) \quad \text{when } \theta > 0 \\ (h-1)\times n \times P_{\text{trust}} + (h-1) \\ \quad \times\,(1-P_{\text{trust}}) \qquad \text{when } \theta = 0 \end{cases} \tag{A.1}$$

This is the exact value of $E(Np)$. However, since: $(h-1-\sum_{i=0}^{h-2}(\theta(1-P_{\text{trust}}))^{n(1-\theta)^i}) < h-1$, and:

$$\left(1+\frac{\theta P_{\text{trust}}}{1-\theta(1-P_{\text{trust}})}\right)\left(\frac{1-P_{\text{trust}}}{1-\theta(1-P_{\text{trust}})}\right)$$

is constant for fixed values of θ and P_{trust} and independent of n, as well as $(h-1)(1-P_{\text{trust}})$. We conclude that for high n, $E(Np)$ could be approximated by:

$$E(Np) \approx \begin{cases} O\left(\frac{P_{\text{trust}}}{1-\theta(1-P_{\text{trust}})}\frac{n}{\theta}\left(1-(1-\theta)^{h-1}\right)\right) \\ \qquad\qquad \text{when } \theta > 0 \\ O((h-1)\times n \times P_{\text{trust}}) \\ \qquad\qquad \text{when } \theta = 0 \end{cases}$$

□

References

1. Awerbuch B, Holmer D, Nita-Rotaru C, Rubens H. An on-demand secure routing protocol resilient to byzantine failures. In *ACM Workshop on Wireless Security (WiSe'02)*, Atlanta, Georgia, September 2002.
2. Marti S, Giuli T, Lai K, Baker M. Mitigating routing misbehavior in mobile ad hoc networks. In *ACM Mobile Computing and Networking, MOBICOM 2000*, Boston, MA, USA, 2000, pp. 255–265.
3. Djenouri D, Badache N. New power-aware routing for mobile ad hoc networks. *The International Journal of Ad Hoc and Ubiquitous Computing (Inderscience Publisher)* 2006; **1**(3): 126–136.
4. Doshi S, Brown T. Minimum energy routing schemes for a wireless ad hoc network. In *The 21st IEEE Annual Joint Conference on Computer Communications and Networking (INFOCOM'02)*, New York, USA, 2002.
5. Djenouri D, Ouali N, Mahmoudi A, Badache N. Random feedbacks for selfish nodes detection in mobile ad hoc networks. In *The 5th IEEE InternationalWorkshop on IP Operations and Management, IPOM'05*, ser. LNCS, no. 3751. Barcelona, Spain: Springer-Verlag GmbH, October 2005, pp. 68–75.
6. Djenouri D, Khalladi L, Badache N. A survey of security issues in mobile ad hoc and sensor networks. *IEEE Communications Surveys* 2005; **7**(4): 2–28.
7. Hu Y-C, Perrig A. A survey of secure wireless ad hoc routing. *IEEE Security and Privacy* 2004; **2**(3): 28–39.
8. David B, David A. Dynamic source routing in ad hoc wireless networks. In *Mobile Computing* (vol. 353), Imielinski T, Korth H (eds). Kluwer Academic: Norwell, MA, USA, 1996; 153–181.
9. Djenouri D, Badache N. A novel approach for selfish nodes detection in manets: proposal and petri nets based modeling. In *The 8th IEEE International Conference on Telecommunications (ConTel'05)*, Zagreb, Croatia, June 2005, pp. 569–574.
10. Capkun S, Buttyan L, Hubaux J-P. Self-organized public-key management for mobile ad hoc networks. *IEEE Transactions on Mobile Computing* 2003; **2**(1): 52–64.
11. Davison A. *Bayesian Models, Chapter 11 in Manuscript*. Springer, 2000.
12. Buchegger S, Le-Boudec J-Y. A robust reputation system for p2p and mobile ad-hoc networks. In *Second Workshop on the Economics of Peer-to-Peer Systems*, Harvard university, Cambridge, MA, USA, June 2004.
13. Zeng X, Bagrodia R, Gerla M. Glomosim: a library for the parallel simulation of large-scale wireless networks. In *The 12th Workshop on Parallel and distributed Simulation. PADS'98*, Banff, Alberta, Canada, May 1998, pp. 154–161.
14. Djenouri D, Badache N. Simulation performance evaluation of an energy efficient routing protocol for mobile ad hoc networks. In *IEEE International Conference on Pervasive Services (ICPS'04)*, American University of Beirut (AUB), Lebanon, July 2004.
15. Yang H, Meng X, Lu S. Self-organized network layer security in mobile ad hoc networks. In *ACM MOBICOM Wireless Security Workshop (WiSe'02), Georgia, Atlanta, USA*, September 2002.
16. Michiardi P, Molva R. Core: a collaborative reputation mechanism to enforce node cooperation in mobile ad hoc networks. In *The 6th IFIP Communication and Multimedia Security Conference*, Portoroz, Slovenia, September 2002.
17. Buchegger S, Boudec J-Y L. Performance analysis of the confidant, protocol cooperation of nodes fairness in dynamic ad hoc networks. In *Third ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc'02)*, Lausanne, Switzerland, June 2002, pp. 80–91.
18. Balakrishnan K, Deng J, Varshney PK. Twoack: preventing selfishness in mobile ad hoc networks. In *The IEEE Wireless Communications and Networking Conference (WCNC'05)*, New Orleans, LA, USA, March 2005, pp. 2137–2142.
19. Buttyan L, Hubaux J. Nuglets: a virtual currency to stimulate cooperation in self-organized mobile ad hoc networks. Swiss Federal Institute of Technology, Lausanne, Switzerland, Tech. Rep. DSC/2001/001, 2001.
20. Buttyan L, Hubaux J. Stimulating cooperation in self-organizing mobile ad hoc networks. *ACM/Kluwer Mobile Networks and Applications* 2003; **8**(5): 579–592.
21. Srinivasan V, Nuggehalli P, Chiasserini CF, Rao RR. Cooperation in wireless ad hoc networks. In *The 22st IEEE Annual Joint Conference on Computer Communications and Networking (INFOCOM'03)*, San Francisco, California, USA, April 2003.
22. Wang W, Li X-Y. Low-cost routing in selfish and rational wireless ad hoc networks. *IEEE Transaction on Mobile Computing* 2006; **5**(5): 596–607.
23. Zhong S, Li LE, Liu YG, Yang YR. On designing incentive-compatible routing and forwarding protocols in wireless ad-hoc networks: an integrated approach using game theoretical and cryptographic techniques. In *The 11th annual international conference on Mobile computing and networking (MobiCom'05)*. ACM Press, 2005, pp. 117–131.
24. Papadimitratos P, Haas ZJ. Secure data transmission in mobile ad hoc networks. In *ACM MOBICOM Wireless Security Workshop (WiSe'03), San Diego, California, USA*, September 2003.

DOI: 10.1002/wcm

25. Conti M, Gregori E, Maselli G. Improving the performability of data transfer in mobile ad hoc networks. In *the Second IEEE International Conference on Sensor and Ad Hoc Communications and Networks (SECON'05)*, Santa Clara, CA, USA, September 2005.
26. Kargl F, Klenk A, Weber M, Schlott S. Advanced detection of selfish or malicious nodes in ad hoc networks. In *1st European Workshop on Security in Ad-Hoc and Sensor Networks, ESAS'04*, Heidelberg, Germany, August 5–6 2004.

Authors' Biographies

Djamel Djenouri received his engineering degree in computer science and the masters degree in computer science from the University of Science and Technology USTHB (Algiers), respectively in 2001 and 2003. He is currently a research assistant at the CERIST center of research in Algiers, and is about to finalize his Ph.D. at USTHB. During his Ph.D., he visited the technical University of Compienge, France, and the John Moors University of Liverpool, U.K., as well. He also participated in many international conferences. Djamel Djenouri works mainly on ad hoc networking, especially on the following topics: security, power management, routing protocols, MAC protocols, and vehicular networks.

Nadjib Badache received his engineering degree in computer science from University of Constantine, Algeria, in 1978, and the master degree from USTHB University in 1982. In 1995, he joined The ADP research group at the IRISA institute in France, where he prepared a Ph.D. thesis on causal ordering and fault tolerance in mobile environment. He obtained his Ph.D. from USTHB in 1998. He is currently a professor at the computer science department of USTHB, where he is also the head of the LSI laboratory. Professor Badache is an author of many papers, and he supervised many thesis and research projects. His interest research areas are: distributed mobile systems, mobile ad hoc networks, and security.

DOI: 10.1002/wcm